Developer Tools > CodeCommit > Repositories > Create repository

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name

Demoapp

100 characters maximum. Other limits apply.

Description - *optional*

My demo web application

1,000 characters maximum

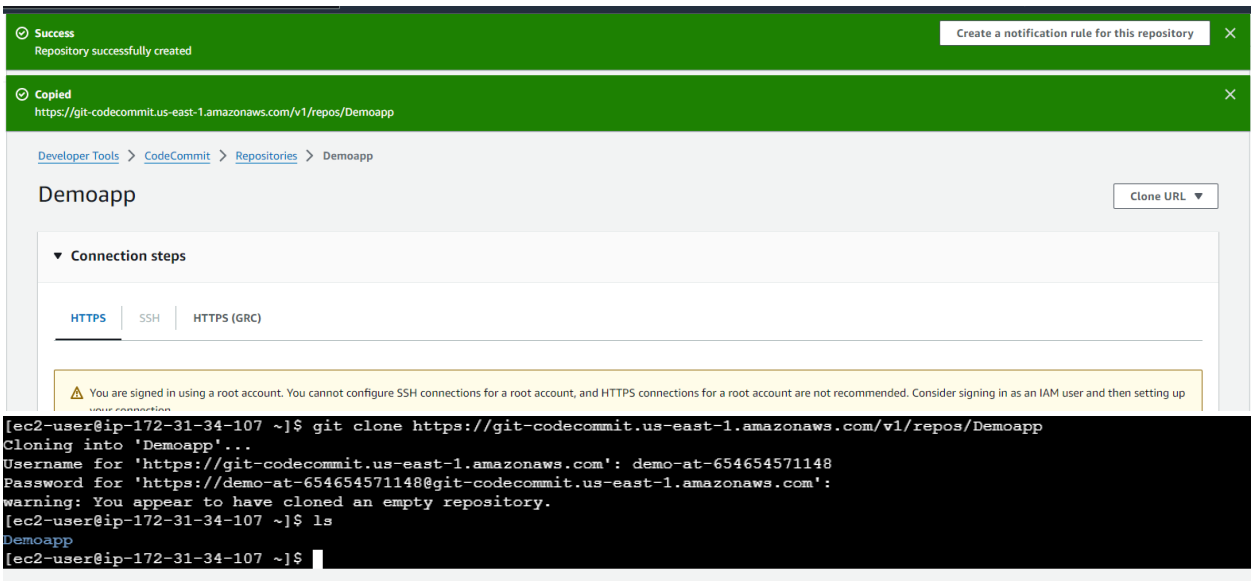
Tags

Add tag

▶ **Additional configuration**

AWS KMS key

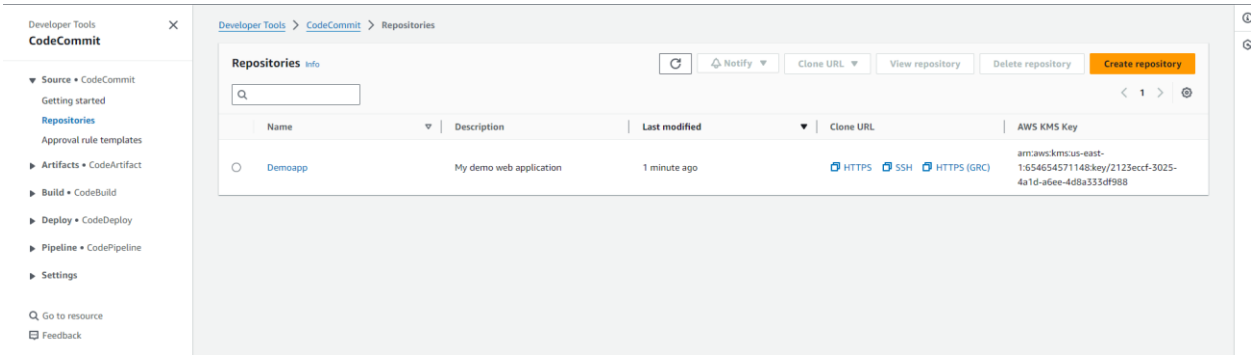
☐ Enable Amazon CodeGuru Reviewer for Java and Python - *optional*

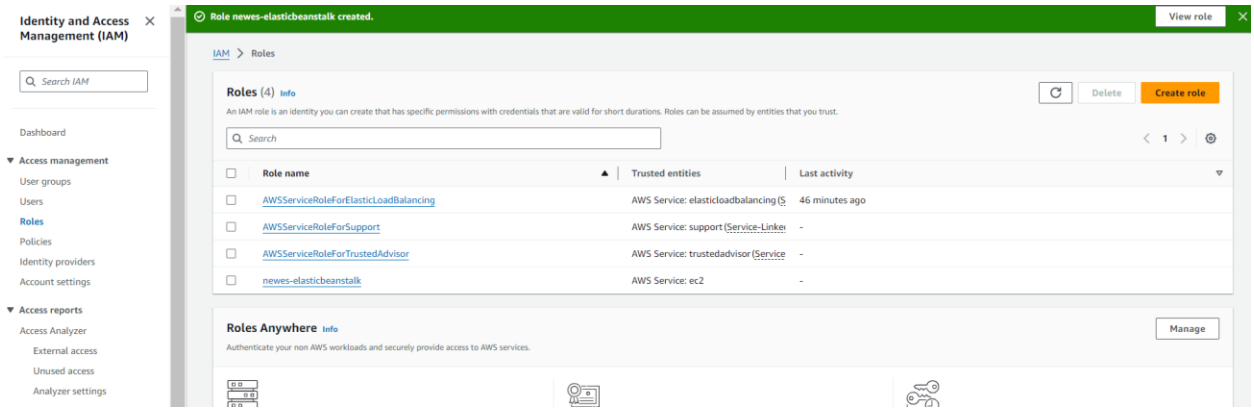


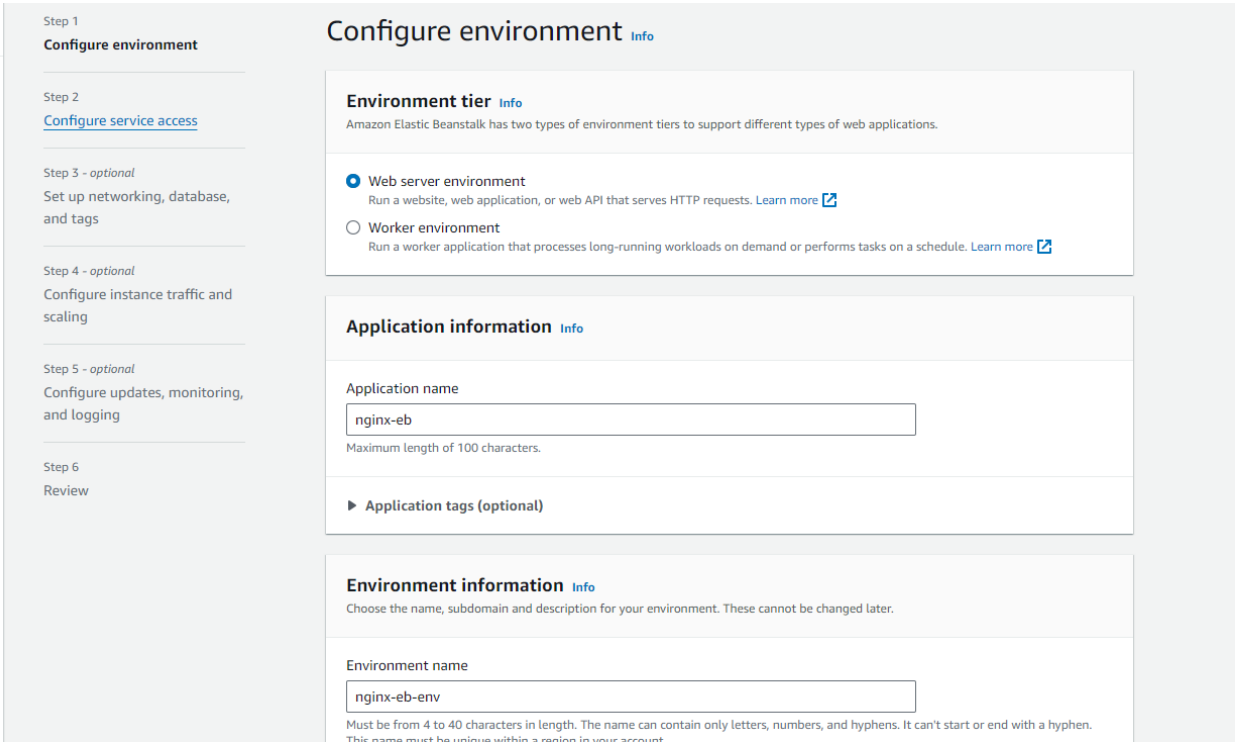
```
[ec2-user@ip-172-31-34-107 ~]$ ls
Demoapp
[ec2-user@ip-172-31-34-107 ~]$ cd Demoapp
[ec2-user@ip-172-31-34-107 Demoapp]$ ls
[ec2-user@ip-172-31-34-107 Demoapp]$ vi index.html
[ec2-user@ip-172-31-34-107 Demoapp]$ git add .
[ec2-user@ip-172-31-34-107 Demoapp]$ git commit -m "index file added"
[master (root-commit) d57cccb] index file added
 Committer: EC2 Default User <ec2-user@ip-172-31-34-107.ec2.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 10 insertions(+)
 create mode 100644 index.html
[ec2-user@ip-172-31-34-107 Demoapp]$
```

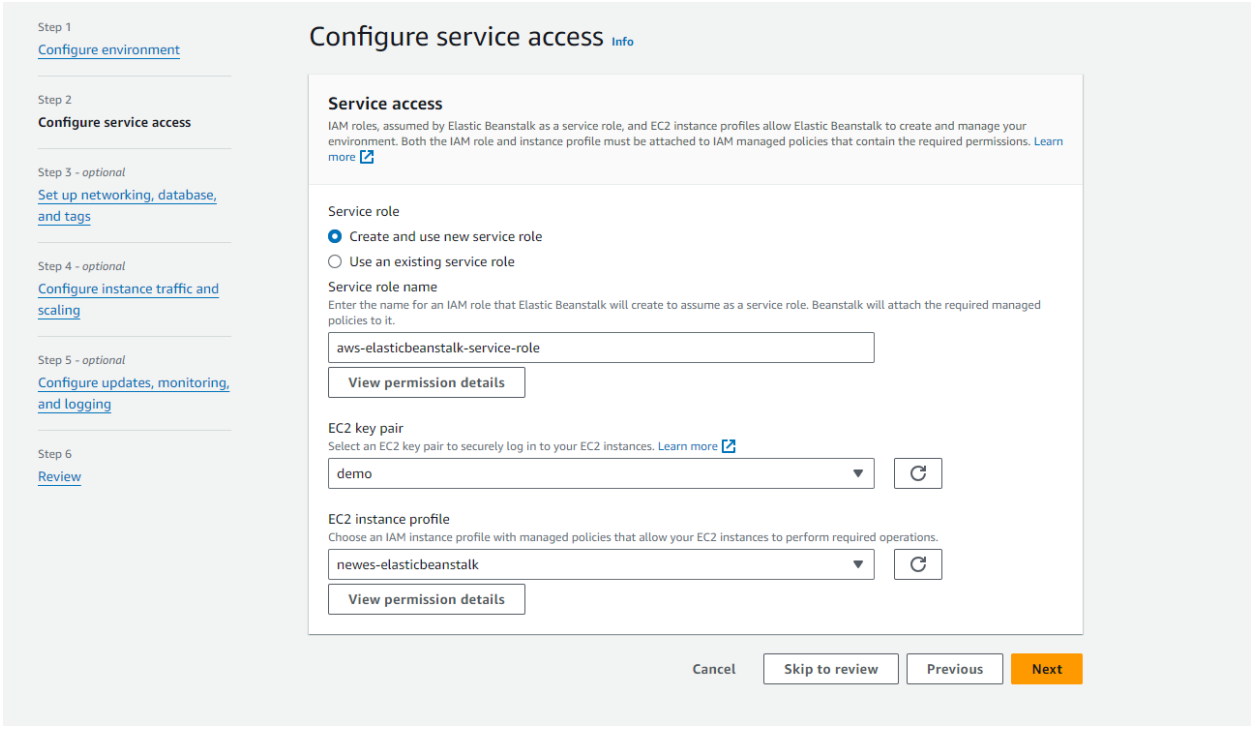
```
Username for 'https://git-codecommit.us-east-1.amazonaws.com': demo-at-654654571148
Password for 'https://demo-at-654654571148@git-codecommit.us-east-1.amazonaws.com':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 352 bytes | 352.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/Demoapp
 * [new branch]      master -> master
[ec2-user@ip-172-31-34-107 Demoapp]$
```

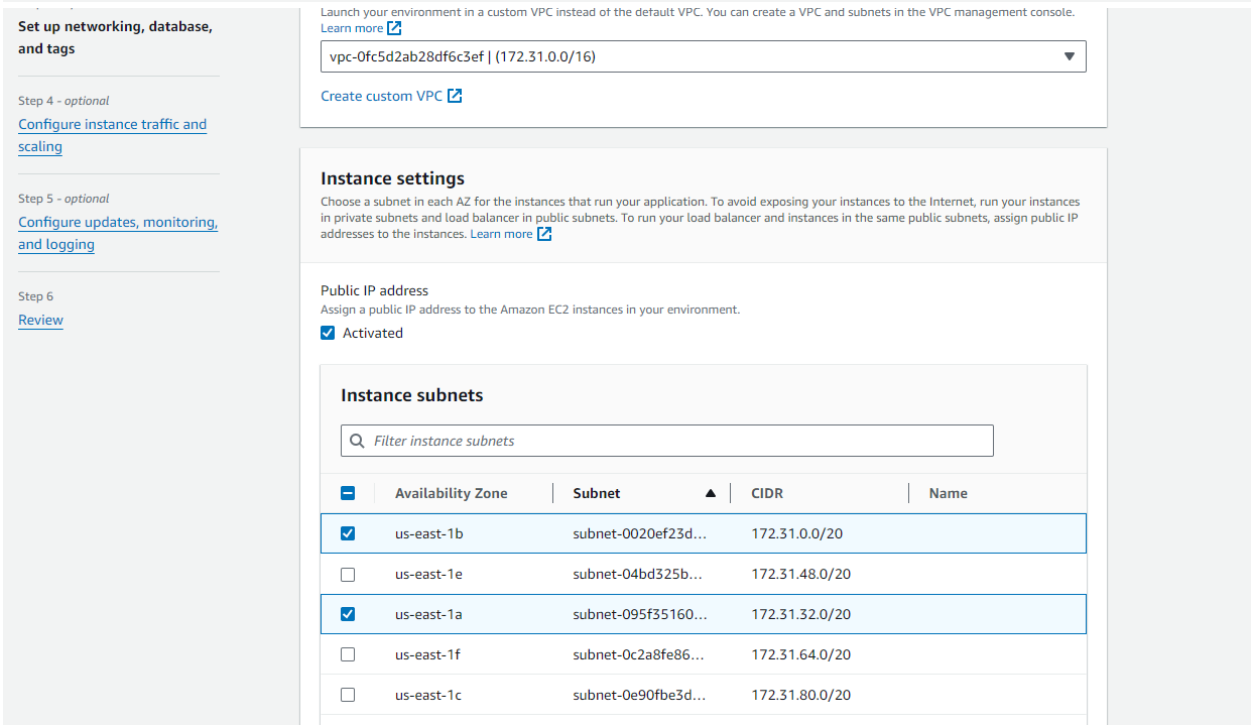
```
[ec2-user@ip-172-31-34-107 Demoapp]$ git push origin master
Username for 'https://git-codecommit.us-east-1.amazonaws.com': demo-at-654654571148
Password for 'https://demo-at-654654571148@git-codecommit.us-east-1.amazonaws.com':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 352 bytes | 352.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Validating objects: 100%
To https://git-codecommit.us-east-1.amazonaws.com/v1/repos/Demoapp
 * [new branch]      master -> master
[ec2-user@ip-172-31-34-107 Demoapp]$
```

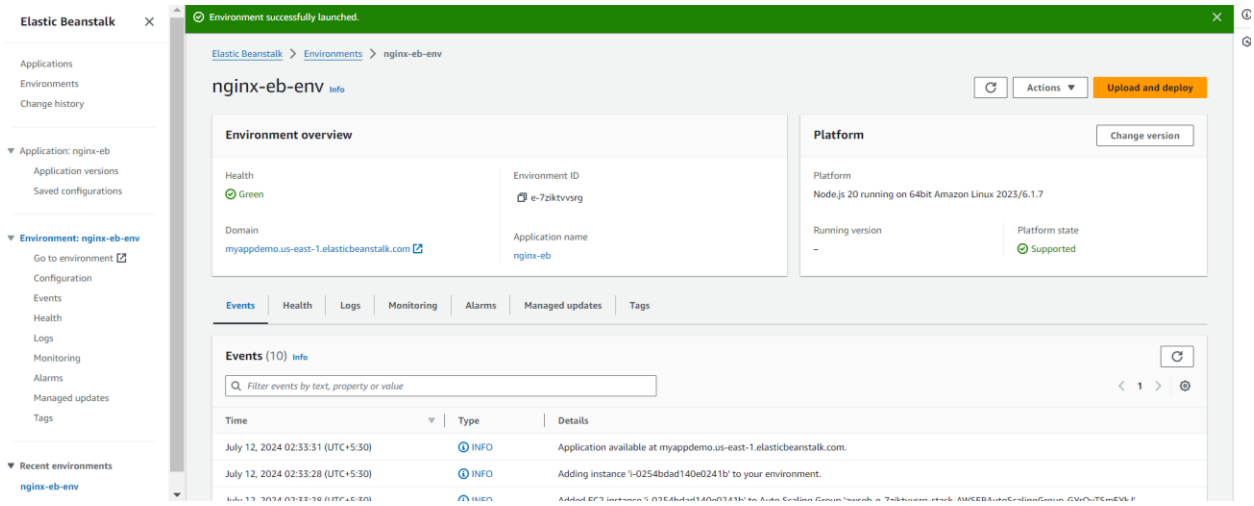


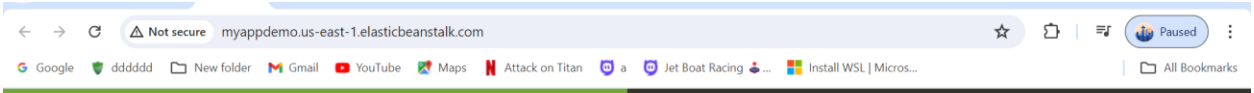

Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Role newes-elasticbeanstalk created.
View role
IAM > Roles
Roles (4) Info
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.
Delete
Create role
Search
Role name
Trusted entities
Last activity
AWSServiceRoleForElasticLoadBalancing
AWS Service: elasticloadbalancing (S
46 minutes ago
AWSServiceRoleForSupport
AWS Service: support (Service-Linke
-
AWSServiceRoleForTrustedAdvisor
AWS Service: trustedadvisor (Service
-
newes-elasticbeanstalk
AWS Service: ec2
-
Roles Anywhere Info
Authenticate your non AWS workloads and securely provide access to AWS services.
Manage


Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure environment Info
Environment tier Info
Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.
Web server environment
Run a website, web application, or web API that serves HTTP requests. Learn more
Worker environment
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. Learn more
Application information Info
Application name
nginx-eb
Maximum length of 100 characters.
Application tags (optional)
Environment information Info
Choose the name, subdomain and description for your environment. These cannot be changed later.
Environment name
nginx-eb-env
Must be from 4 to 40 characters in length. The name can contain only letters, numbers, and hyphens. It can't start or end with a hyphen.


Step 1
Configure environment
Step 2
Configure service access
Step 3 - optional
Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Configure service access Info
Service access
IAM roles, assumed by Elastic Beanstalk as a service role, and EC2 instance profiles allow Elastic Beanstalk to create and manage your environment. Both the IAM role and instance profile must be attached to IAM managed policies that contain the required permissions. Learn more
Service role
Create and use new service role
Use an existing service role
Service role name
Enter the name for an IAM role that Elastic Beanstalk will create to assume as a service role. Beanstalk will attach the required managed policies to it.
aws-elasticbeanstalk-service-role
View permission details
EC2 key pair
Select an EC2 key pair to securely log in to your EC2 instances. Learn more
demo
EC2 instance profile
Choose an IAM instance profile with managed policies that allow your EC2 instances to perform required operations.
newes-elasticbeanstalk
View permission details
Cancel
Skip to review
Previous
Next


Set up networking, database, and tags
Step 4 - optional
Configure instance traffic and scaling
Step 5 - optional
Configure updates, monitoring, and logging
Step 6
Review
Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console. Learn more
vpc-0fc5d2ab28df6c3ef | (172.31.0.0/16)
Create custom VPC
Instance settings
Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances in private subnets and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances. Learn more
Public IP address
Assign a public IP address to the Amazon EC2 instances in your environment.
Activated
Instance subnets
Filter instance subnets
Availability Zone
Subnet
CIDR
Name
us-east-1b subnet-0020ef23d... 172.31.0.0/20
us-east-1e subnet-04bd325b... 172.31.48.0/20
us-east-1a subnet-095f35160... 172.31.32.0/20
us-east-1f subnet-0c2a8fe86... 172.31.64.0/20
us-east-1c subnet-0e90fbe3d... 172.31.80.0/20





Congratulations

Your first AWS Elastic Beanstalk Node.js application is now running on your own dedicated environment in the AWS Cloud

This environment is launched with Elastic Beanstalk Node.js Platform

What's Next?

- AWS Elastic Beanstalk overview
- AWS Elastic Beanstalk concepts
- Deploying an Express Application to AWS Elastic Beanstalk
- Deploying an Express application with clustering to Elastic Beanstalk
- Customizing and Configuring a Node.js Container
- Working with Logs