AWS Elastic Beanstalk – Getting Started and Exploring Deployment Options

To help you understand how AWS Elastic Beanstalk works, this tutorial walks you through creating, exploring, updating, and deleting an Elastic Beanstalk application. Additionally, you will be performing an immutable environment update.

There is no cost for using Elastic Beanstalk, but the AWS resources that it creates for this tutorial are live (and don't run in a sandbox). You incur the standard usage fees for these resources until you terminate them at the end of this tutorial. The total charges are typically less than a dollar.

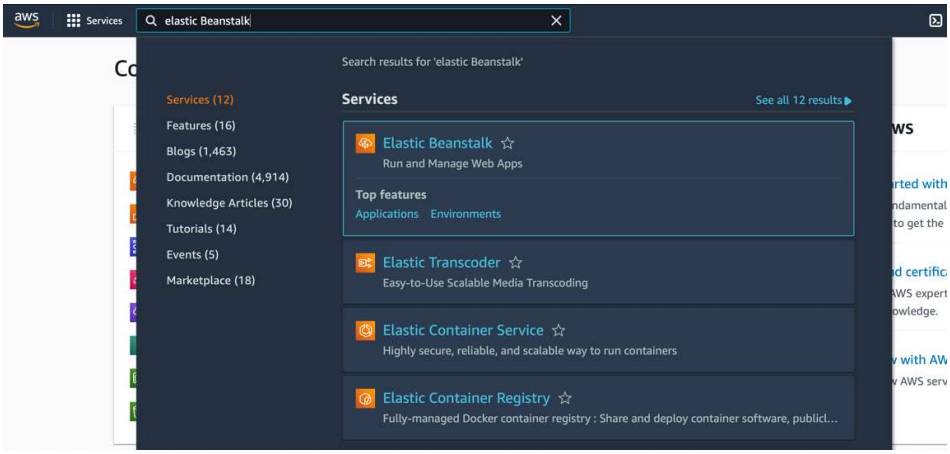
Following are the series of steps:

- Step 1: Create an application and an environment (test)
- Step 2: Explore your environment
- Step 3: Create a new environment (prod) within the same application
- Step 4: Configure your environment (prod)
- Step 5: Perform an immutable environment update
- Step 6: Clean up

Step 1: Create an application and an environment (test)

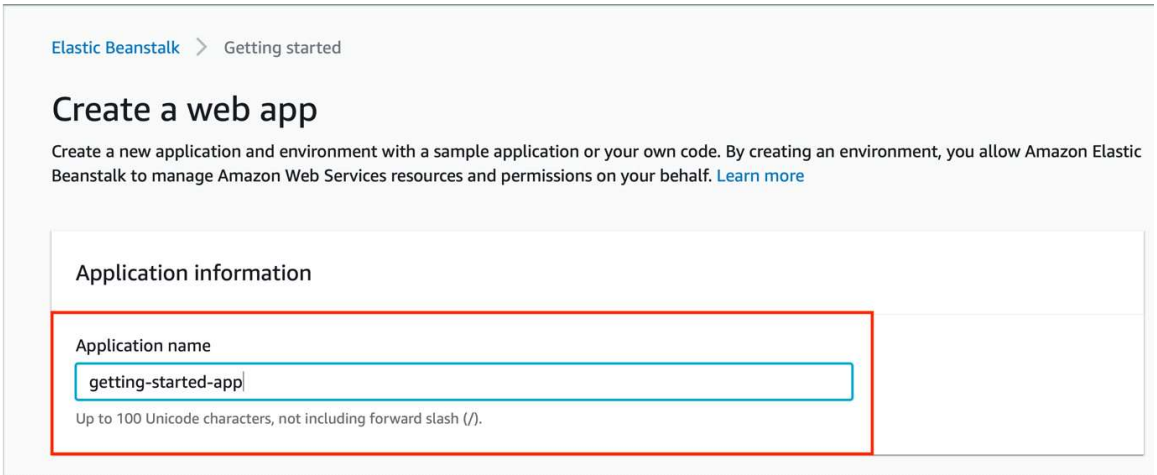
In this step, you create a new application starting from a preexisting sample code. Elastic Beanstalk supports platforms for different programming languages, application servers, and Docker containers. You choose a platform when you create the application.

- After logging in to AWS management console, search for Elastic Beanstalk and click it to go to Elastic Beanstalk dashboard.

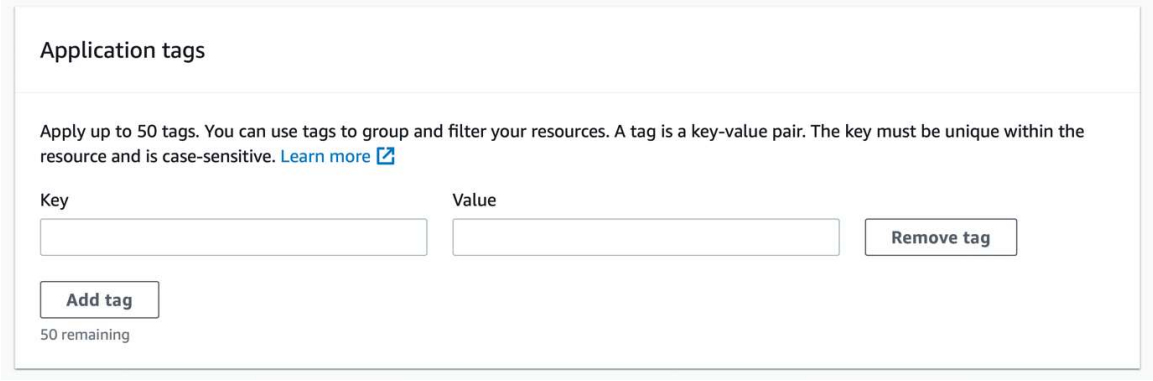


- Choose **Create Application**.

- Within **Application name**, type a name you want to allocate to this sample application.

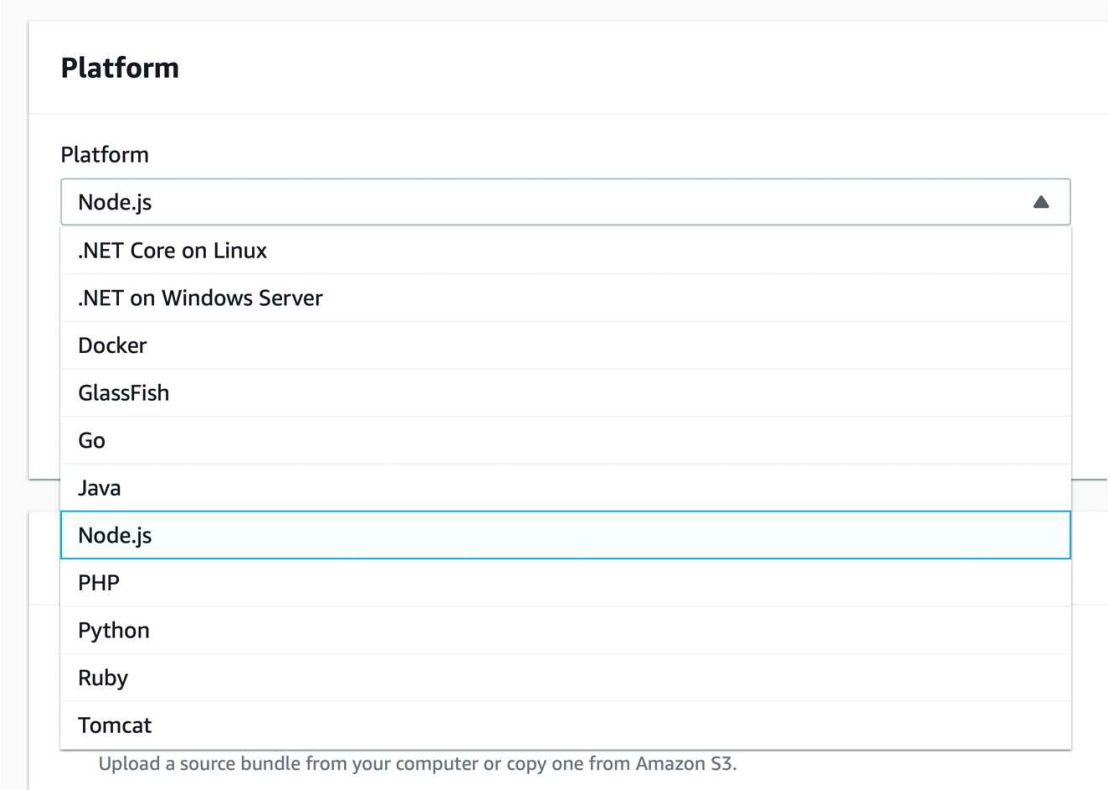


- Optionally, you may add application tags. Keep it blank for this hands-on.

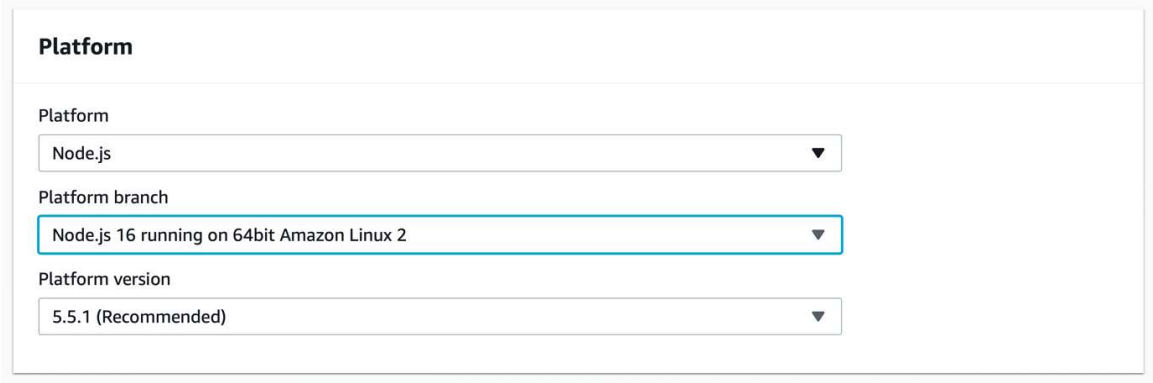


- For **platform**, select Node.js.

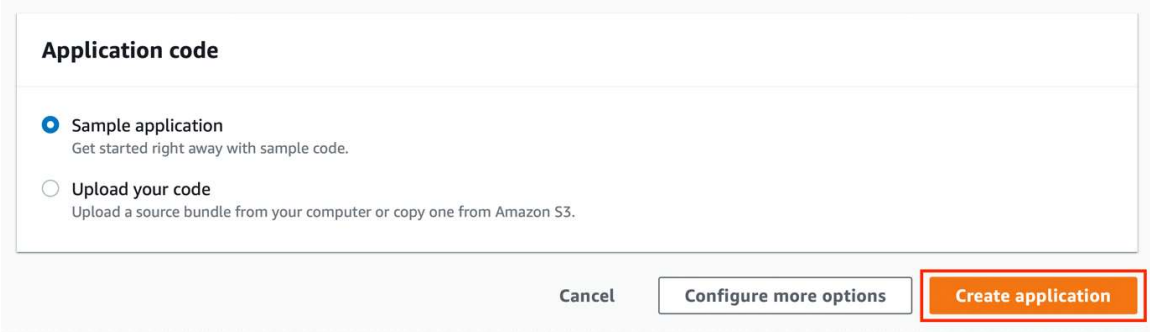
A platform is a combination of an operating system, programming language runtime, web server, application server, and Elastic Beanstalk components. You design and target your web application to a platform. Elastic Beanstalk provides a variety of platforms on which you can build your applications.



- The options like **Platform branch** and **Platform version** will be selected automatically. Keep these parameters to the default values without changing them.



- For **Application code**, choose **Sample application** and click **Create Application**.



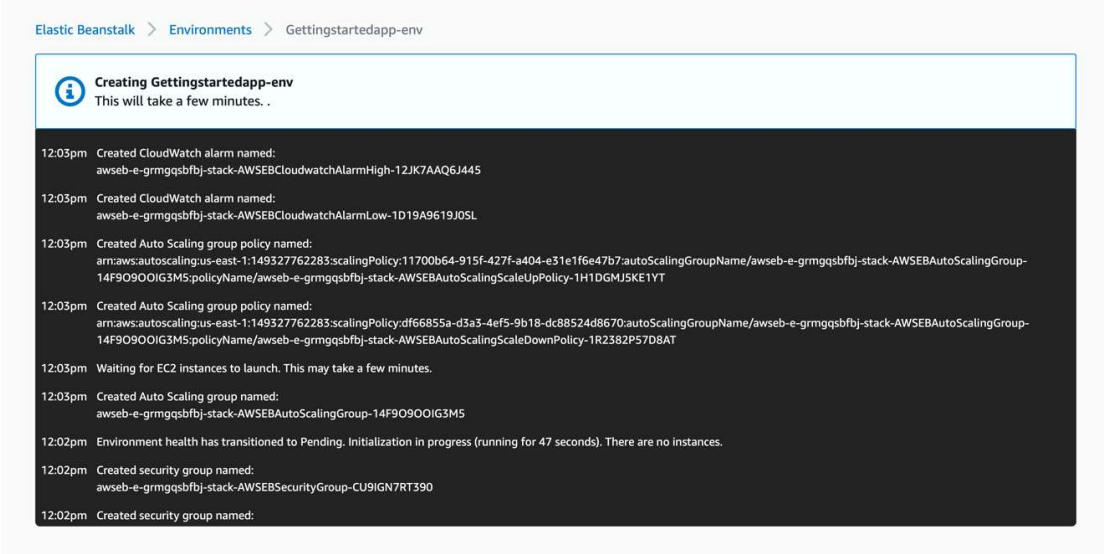
- This will initiate the process to deploy your sample application.

- To run the example application on AWS resources, Elastic Beanstalk takes the following actions. They take about five minutes to complete.

- Creates an Elastic Beanstalk application named getting-started-app.
- Launches an environment named GettingStartedApp-env with these AWS resources:
    - An Amazon Elastic Compute Cloud (Amazon EC2) instance (virtual machine)
    - An Amazon EC2 security group
    - An Auto Scaling Group
    - An Application Load Balancer

    - An Amazon Simple Storage Service (Amazon S3) bucket
    - Amazon CloudWatch alarms
    - An AWS CloudFormation stack
    - A domain name
- Creates a new application version named Sample Application. This is the default Elastic Beanstalk example application file.
- Deploys the code for the sample application to the GettingStartedApp-env environment.

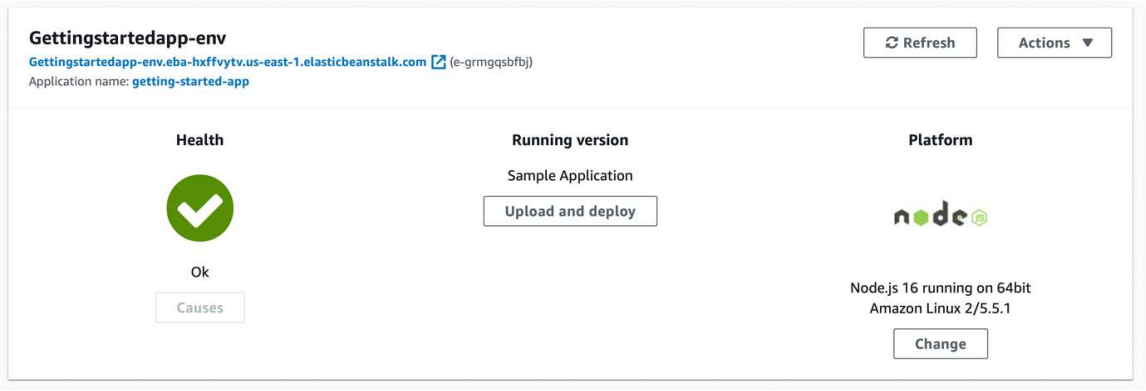
During the environment creation process, the console tracks progress and displays events.



- When all the resources are launched and the EC2 instances running the

application pass health checks, the environment's health changes to Ok.

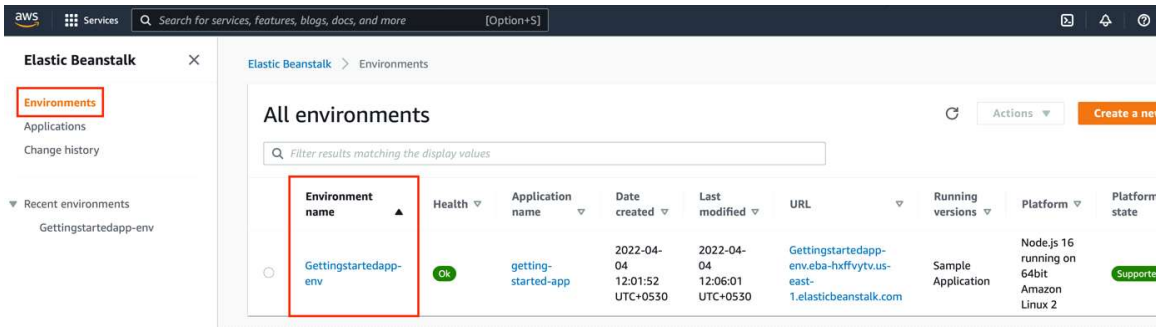
You can now use your web application's website.



Step 2: Explore your environment

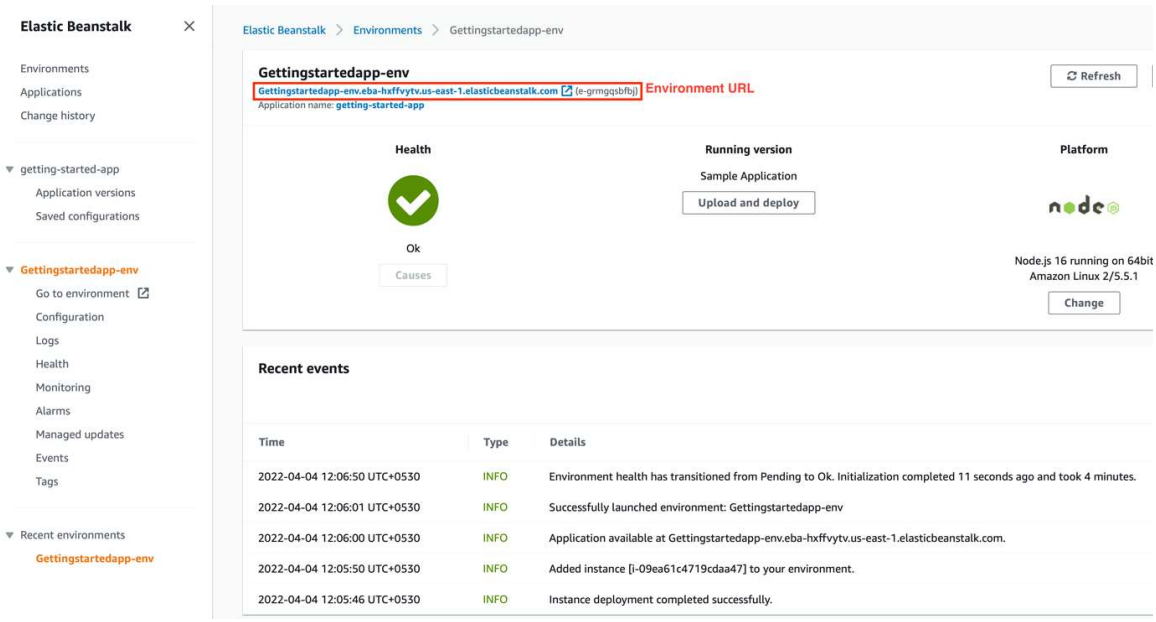
To see an overview of your Elastic Beanstalk application's environment, use the environment page in the Elastic Beanstalk console.

- In the navigation pane, choose Environments, and then choose the name of your environment from the list.



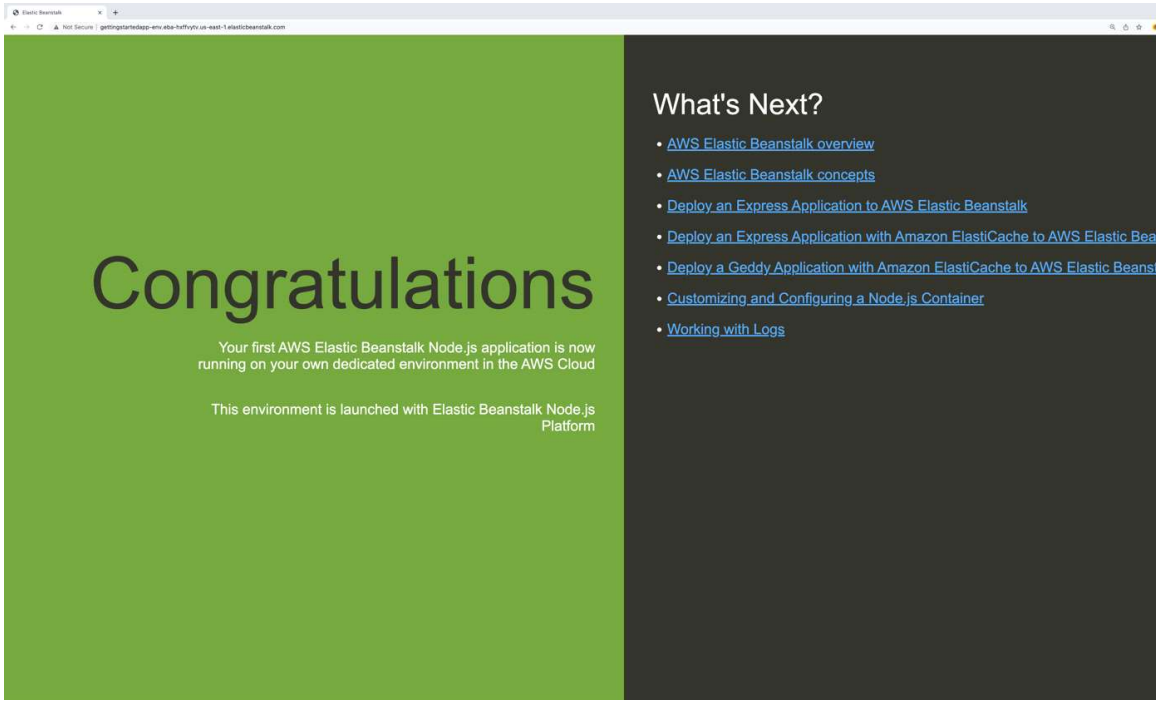
The environment overview pane shows top level information about your environment. This includes its name, its URL, its current health status, the name of the currently deployed application version, and the platform version that the application is running on.

Below the overview pane you can see the five most recent environment events.



- The environment's URL is located at the top of the overview, below the environment name. This is the URL of the web application that the environment is running.

Choose this URL to get to the sample application's Congratulations page.



The navigation page on the left side of the console links to other pages that contain more detailed information about your environment and provide access to additional features:

- **Configuration** – Shows the resources provisioned for this environment, such as the Amazon Elastic Compute Cloud (Amazon EC2) instances that host your application. You can configure some of the provisioned resources on this page.

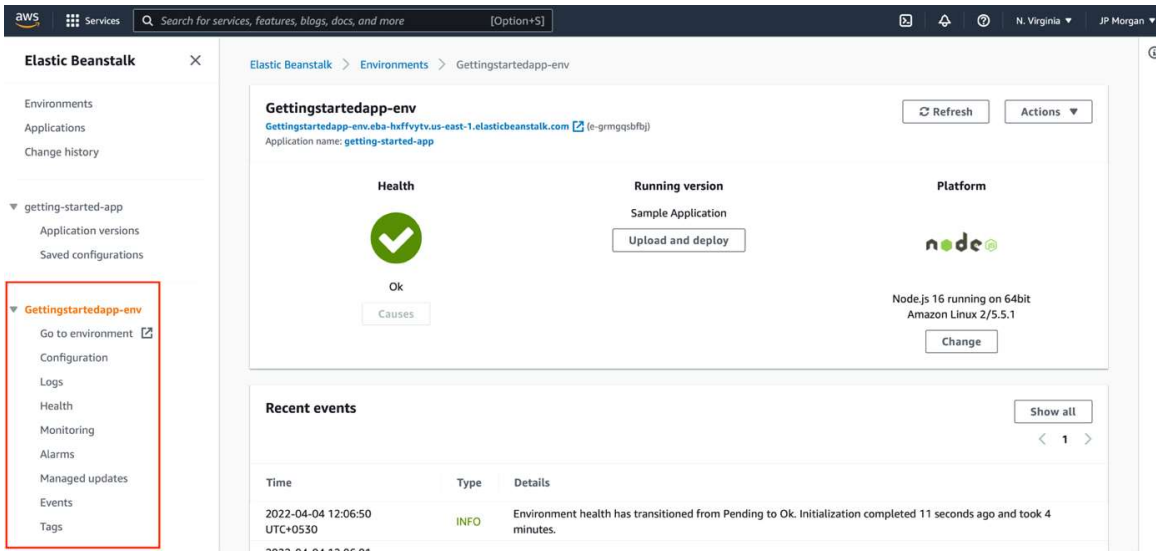
- **Health** – Shows the status of and detailed health information about the Amazon EC2 instances running your application.

- **Monitoring** – Shows statistics for the environment, such as average latency and CPU utilization. You can use this page to create alarms for the metrics

that you are monitoring.

- **Events** – Shows information or error messages from the Elastic Beanstalk service and from other services whose resources this environment uses.

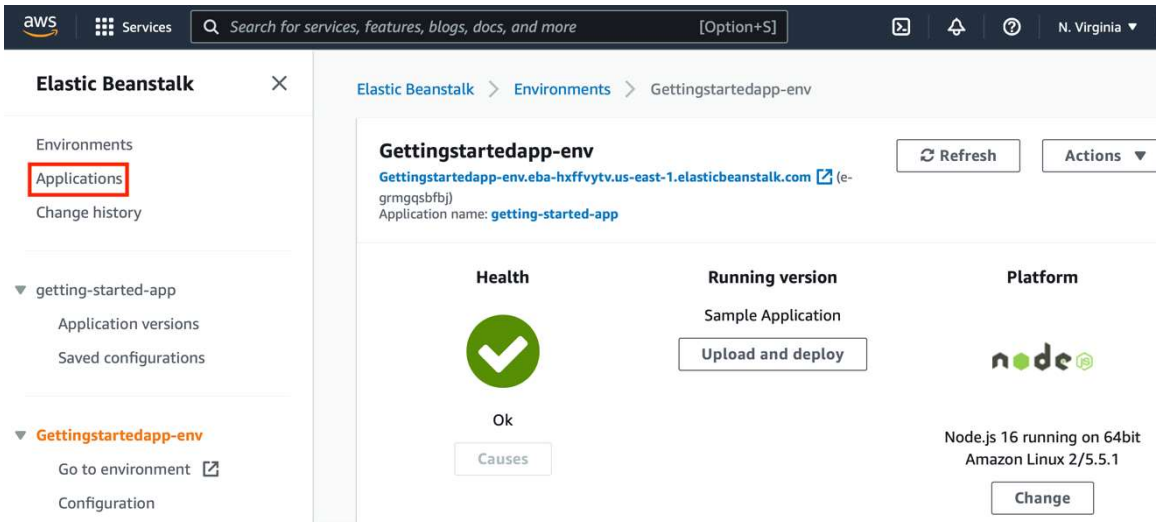
- **Tags** – Shows environment tags and allows you to manage them. Tags are key-value pairs that are applied to your environment.

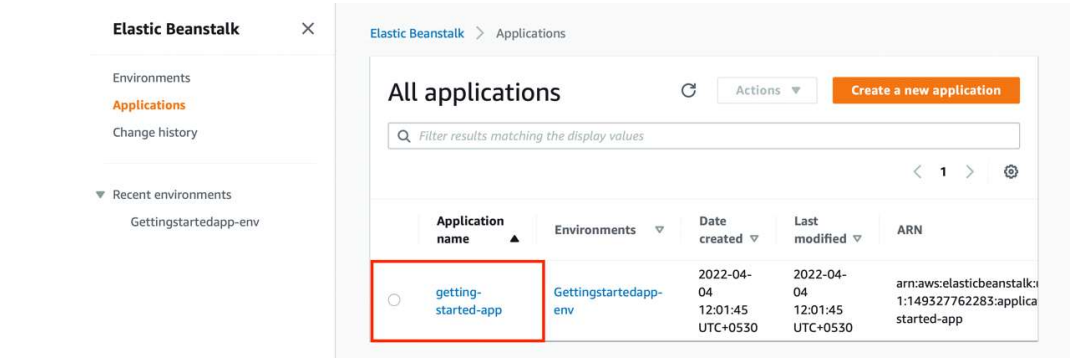


Step 3: Create a new environment (prod) within the same application

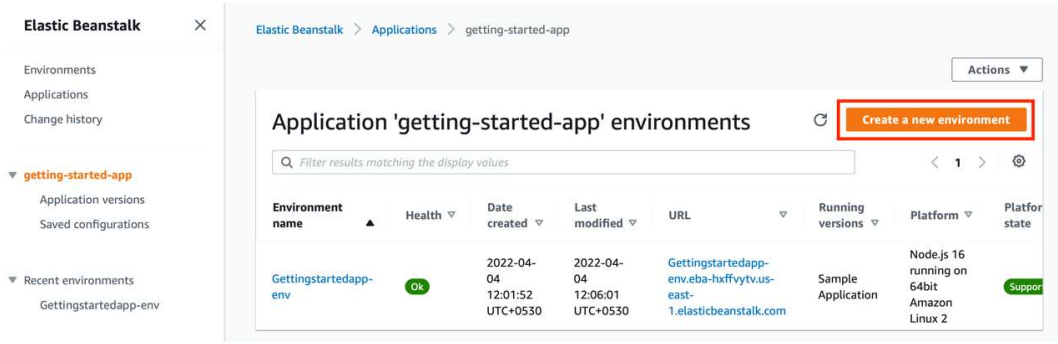
In this step, you will be creating a new environment intended for production use, and additionally explore and configure some advanced options.

- In the navigation pane, choose Applications, and then choose your application's name from the list.

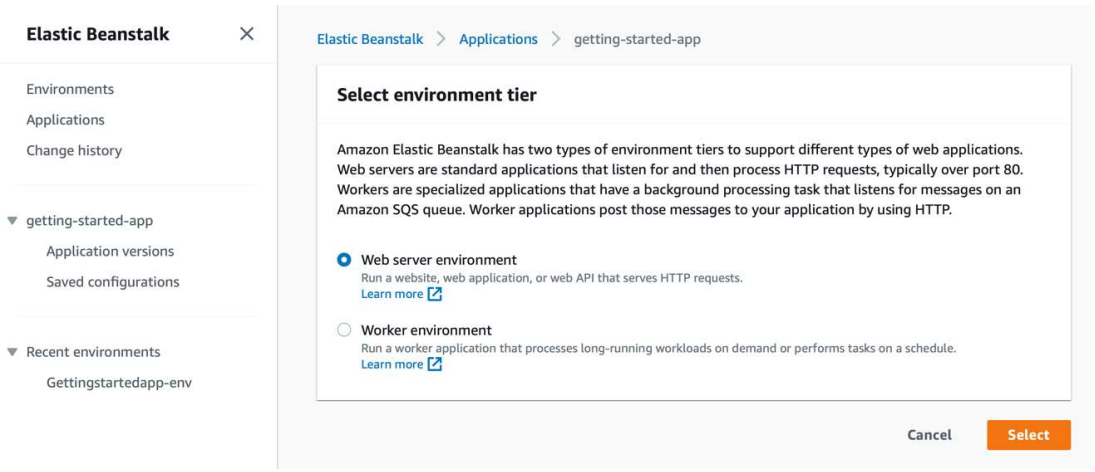




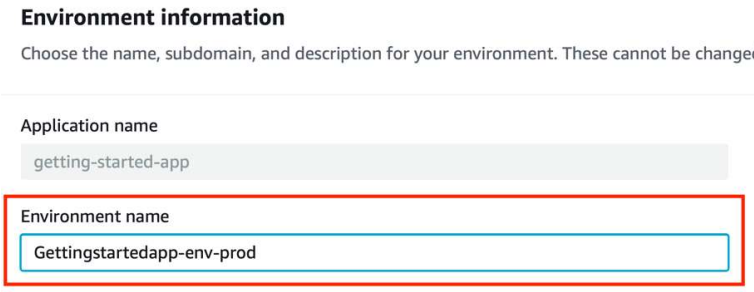
- On the application overview page, choose **Create a new environment**.



- Next, for environment tier, choose the **Web server environment** or **Worker environment** tier. You can't change an environment's tier after creation.

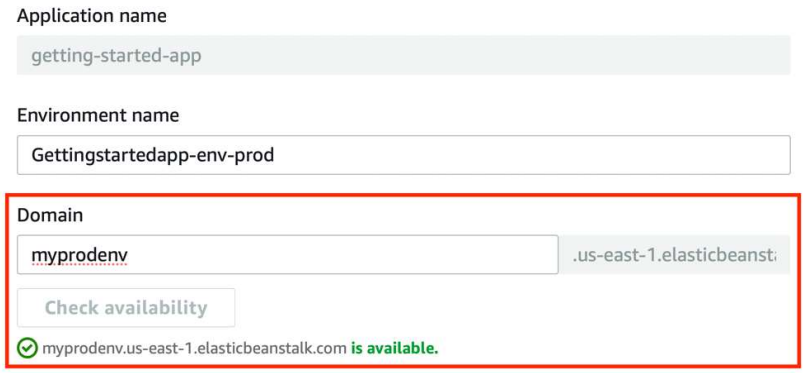


- Enter a name for the environment. The form provides a generated name. Add keyword *prod* at the end so that this environment can be easily identified as a production environment.



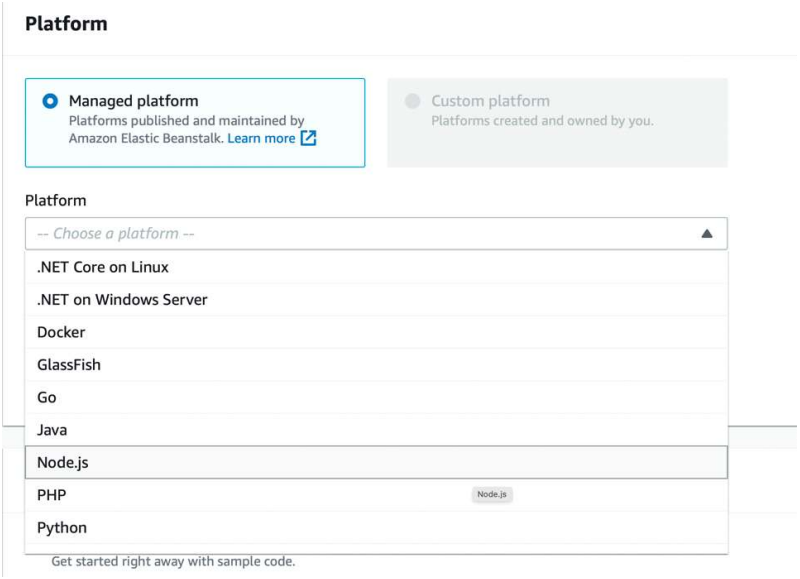
- Enter a unique domain name for your environment. The default name is the environment's name. You can enter a different domain name. Elastic Beanstalk uses this name to create a unique CNAME for the environment. To check whether the domain name you want is available, choose **Check**

Availability.



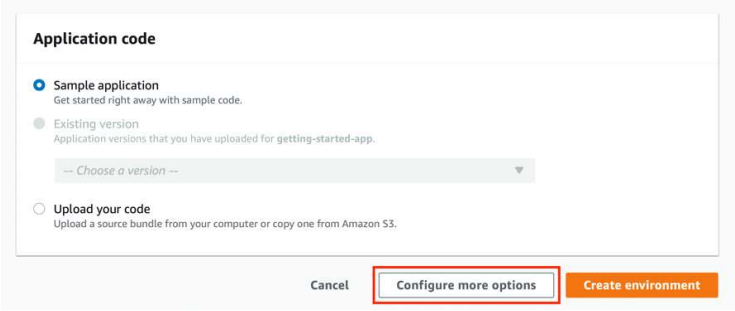
- Enter a description for this environment.

- For **platform**, select Node.js.



- The options like **Platform branch** and **Platform version** will be selected automatically. Keep these parameters to the default values without changing them.

- For **Application code**, choose **Sample application** and click **Configure more options**.



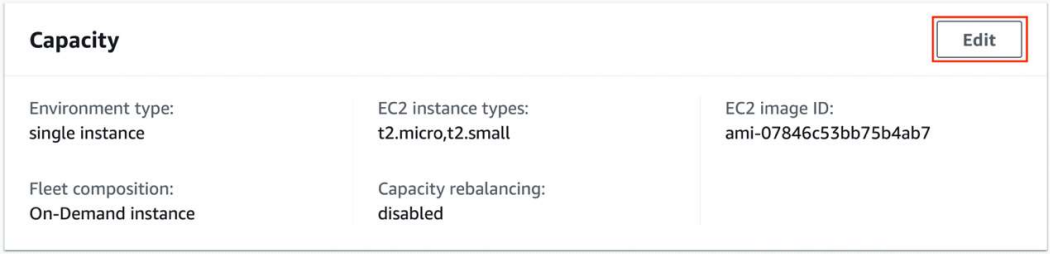
Step 4: Configure your environment (prod)

You can configure your environment to better suit your application. For example, if you have a compute-intensive application, you can change the type of Amazon Elastic Compute Cloud (Amazon EC2) instance that is running your application. To apply configuration changes, Elastic Beanstalk performs an environment update.

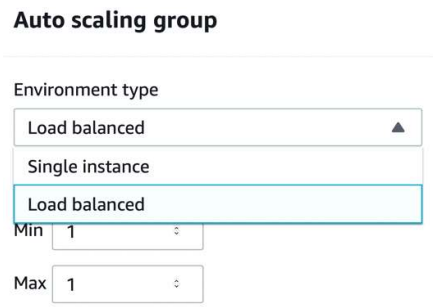
Some configuration changes are simple and happen quickly. Some changes require deleting and recreating AWS resources, which can take several minutes. When you change configuration settings, Elastic Beanstalk warns you about potential application downtime.

In this step of a configuration change, you edit your environment's capacity settings. You configure a load-balanced, scalable environment that has between two and four Amazon EC2 instances in its Auto Scaling group, and then you verify that the change occurred. Elastic Beanstalk creates two Amazon EC2 instances. Then, Elastic Beanstalk associates both instances with the environment's load balancer. As a result, your application's responsiveness is improved and its availability is increased.

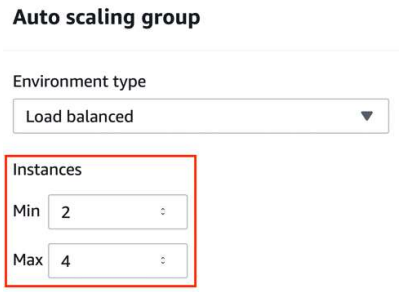
- In the **Capacity** configuration category, choose **Edit**.



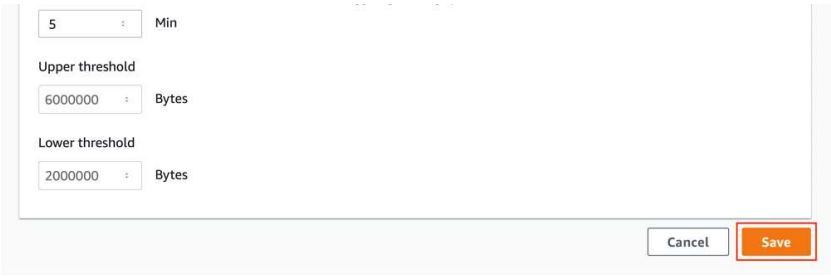
- In the **Auto Scaling group** section, change **Environment type** to **Load balanced**.



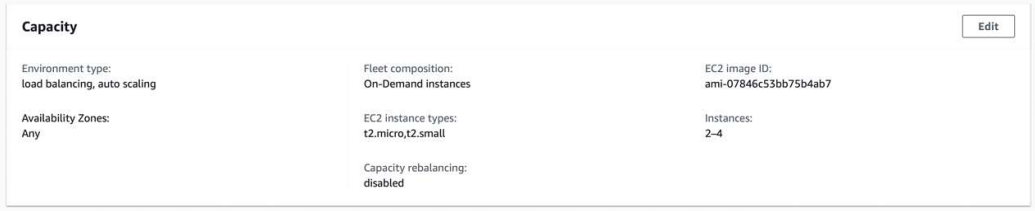
- In the **Instances** row, change **Max** to 4, and then change **Min** to 2.



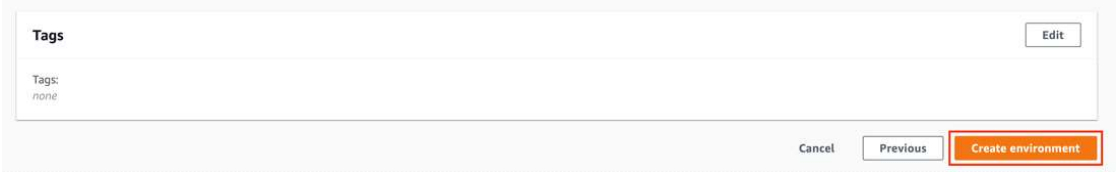
- Scroll down while accepting other parameters set to their default values and click **Save**.



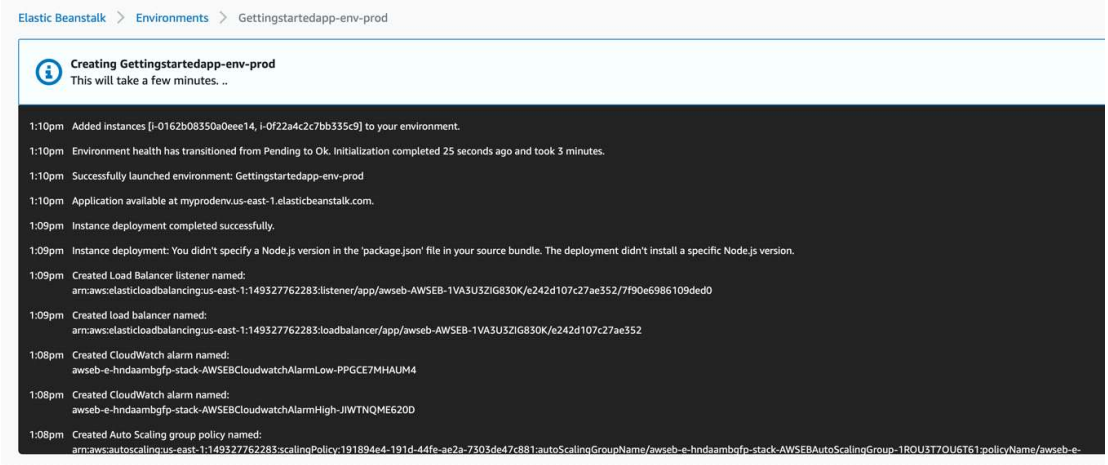
- Once you land back to configuration page, you can easily review the changes made by you.



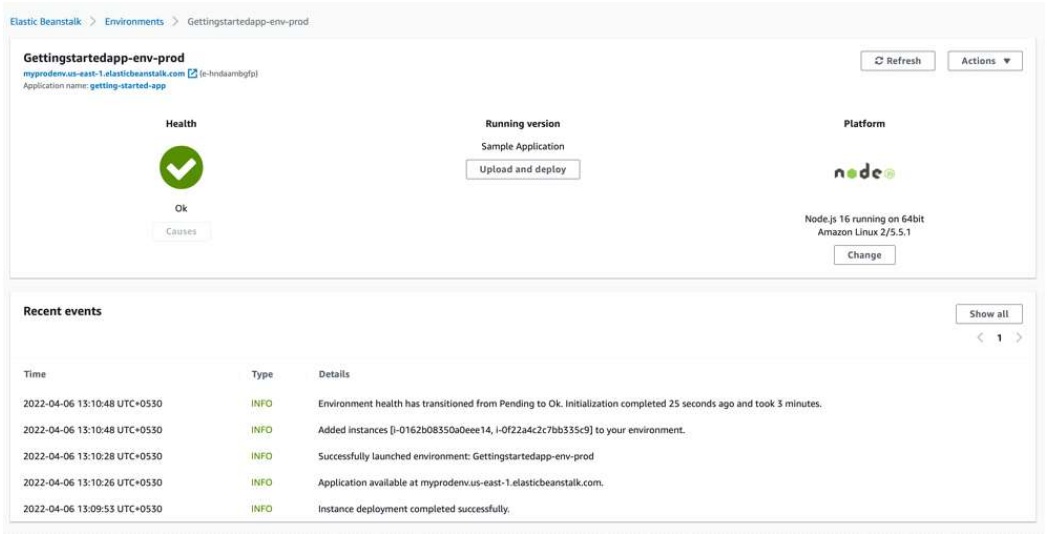
- Scroll down and click **Create Environment**.



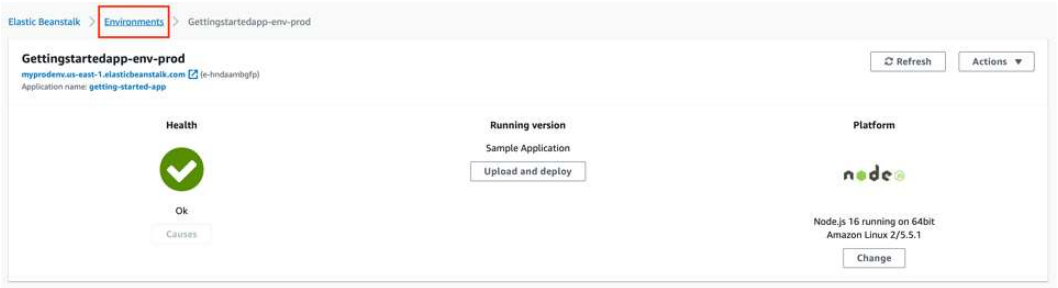
- While your production environment is getting created the console tracks progress and displays events. This will take a few minutes.



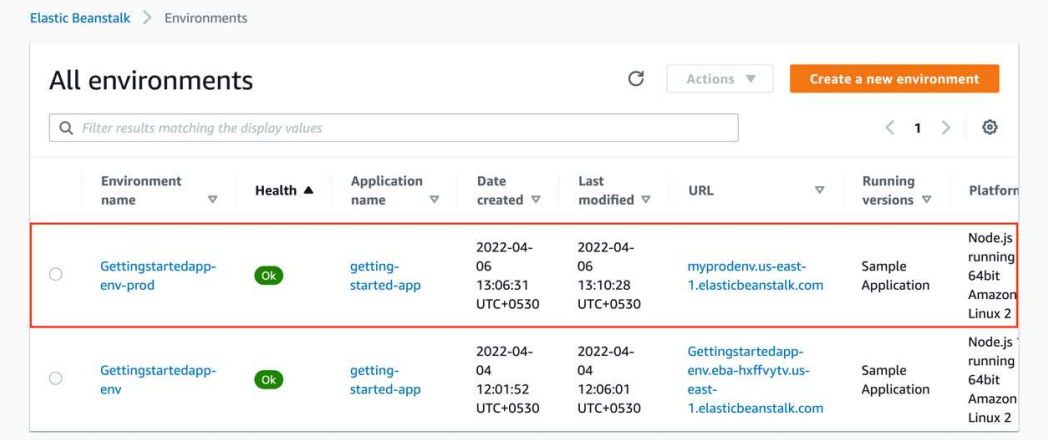
- Finally, your production environment is created.



- Click **Environments** to navigate back to environments' web page.



- Now you can see two environments out of which one is the production one that you just configured and created.



Step 5: Perform an immutable environment update

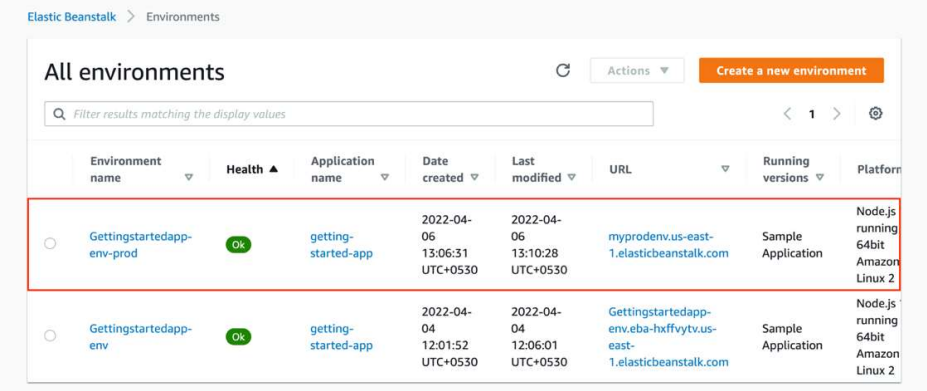
Immutable environment updates are an alternative to rolling updates. Immutable environment updates ensure that configuration changes that require replacing instances are applied efficiently and safely. If an immutable environment update fails, the rollback process requires only terminating an Auto Scaling group. A failed rolling update, on the other hand, requires performing an additional rolling update to roll back the changes.

To perform an immutable environment update, Elastic Beanstalk creates a second, temporary Auto Scaling group behind your environment's load balancer to contain the new instances. First, Elastic Beanstalk launches a single instance with the new configuration in the new group. This instance serves traffic alongside all of the instances in the original Auto Scaling group that are running the previous configuration.

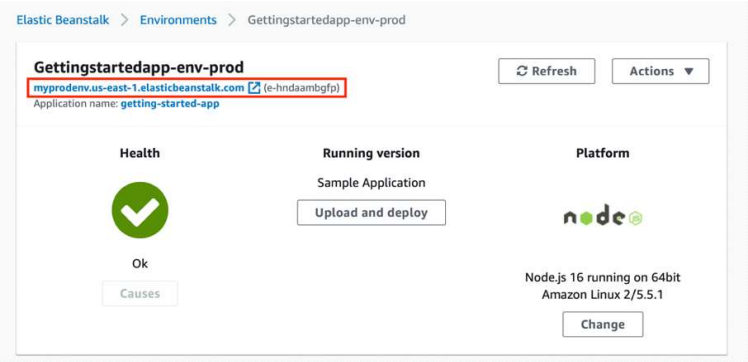
When the first instance passes health checks, Elastic Beanstalk launches additional instances with the new configuration, matching the number of instances running in the original Auto Scaling group. When all the new instances pass health checks, Elastic Beanstalk transfers them to the original Auto Scaling group and terminates the temporary Auto Scaling group and old instances.

In this step, you will be pushing the immutable update to your production environment.

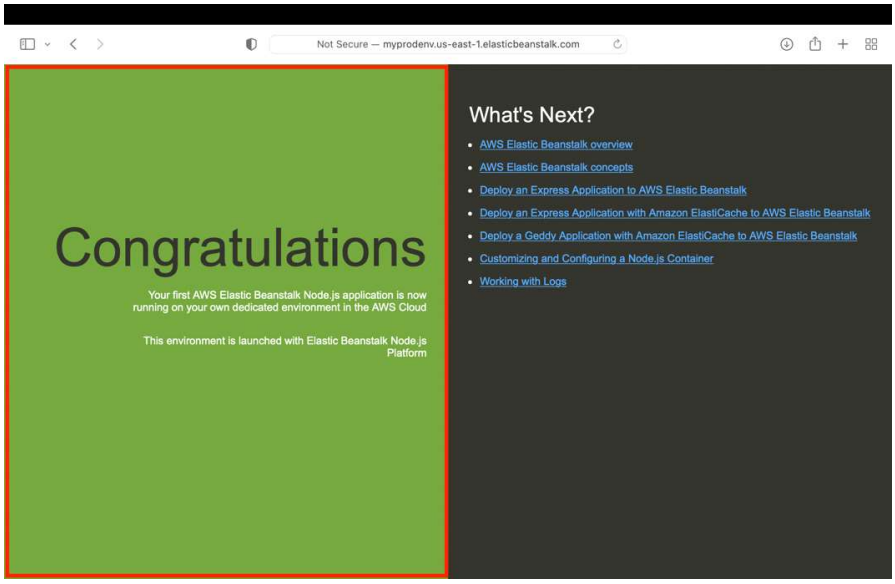
- Choose your production environment.



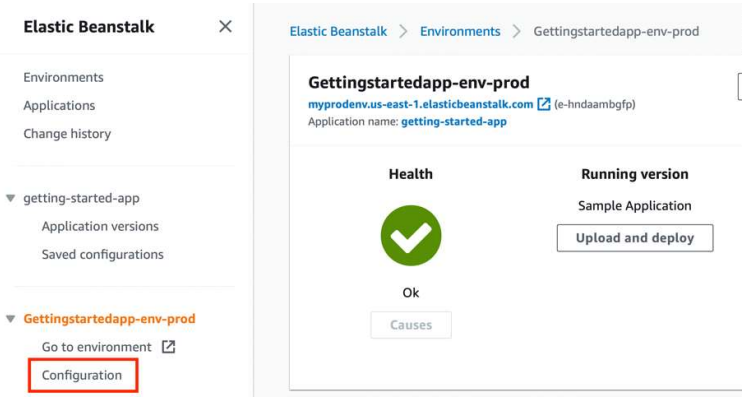
- Click on the production environment URL to access the sample website.



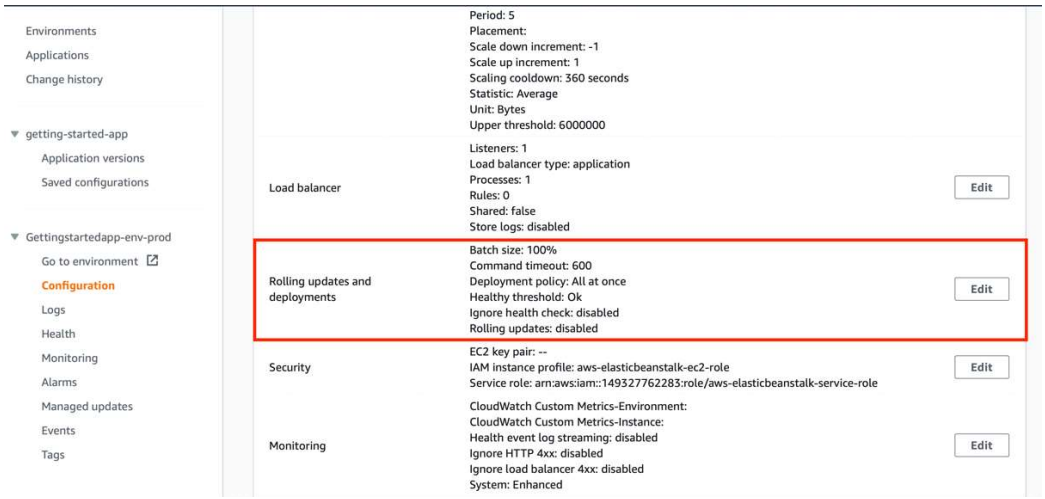
- You will be able to see the following sample website with green coloured column on the left. After pushing the immutable update this will be changed to blue.



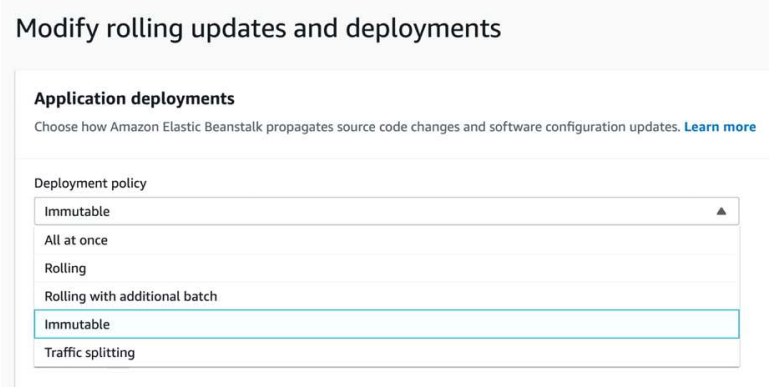
- In the navigation pane, choose **Configuration** under your production environment.



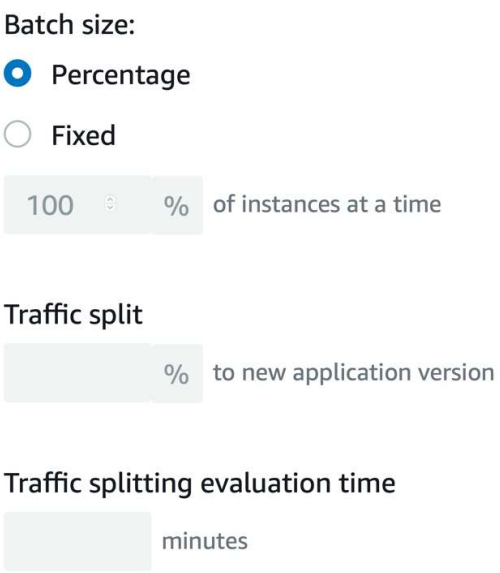
- In the **Rolling updates and deployments** configuration category, choose **Edit**.



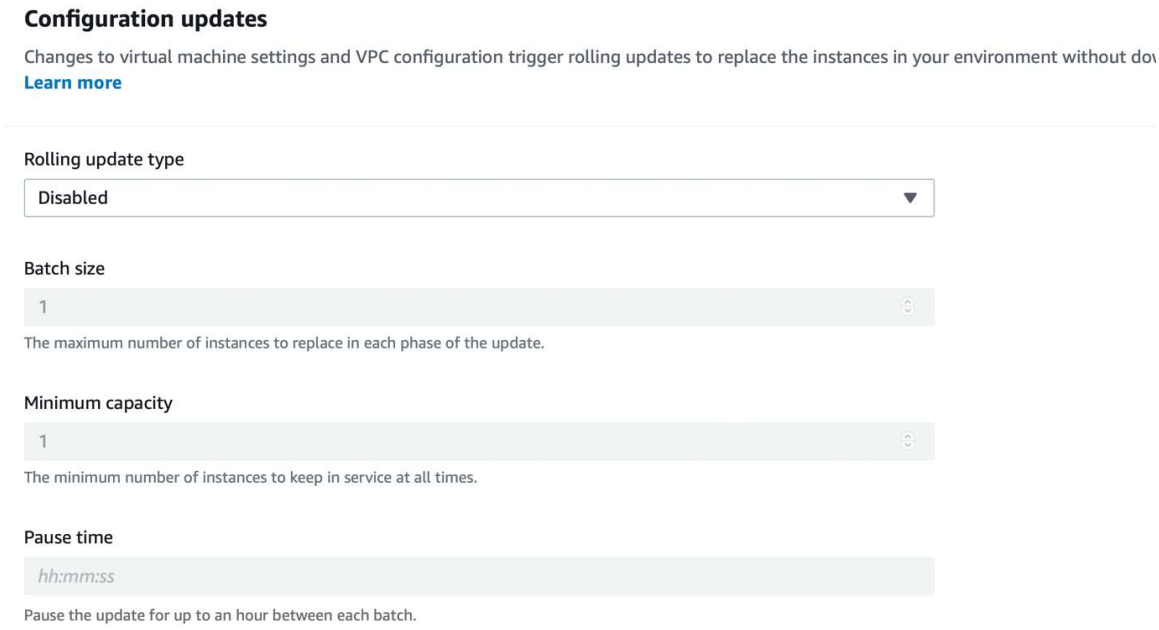
- In the **Configuration Updates** section, set **Rolling update** type to **Immutable**.



- As you have chosen Immutable deployment option, defining the batch size is irrelevant in this case. Skip this step and scroll down the page.



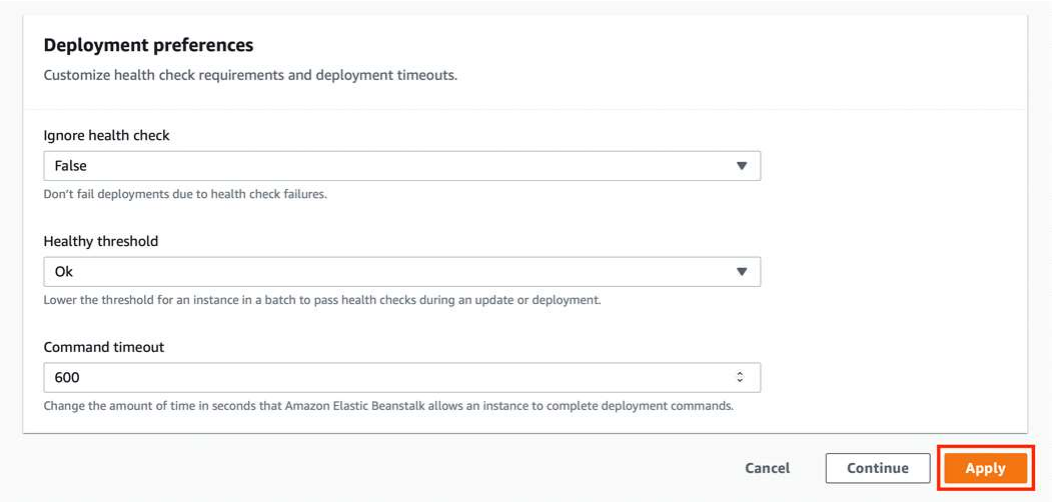
- For this deployment scenario there is no requirement to change and apply **Configuration updates** settings. Skip this step and scroll down the page.



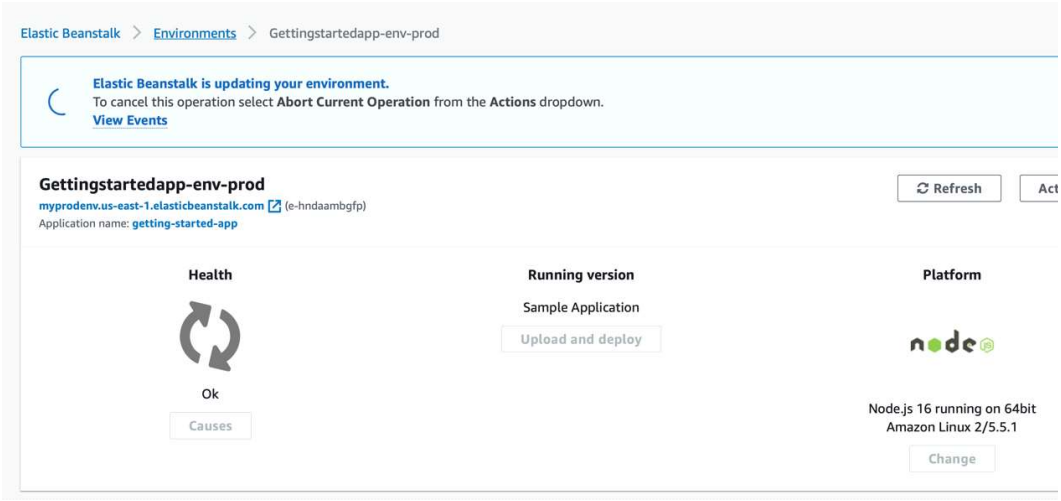
- The **Deployment preferences** section contains options related to health checks.

- **Ignore health check** – Prevents a deployment from rolling back when a batch fails to become healthy within the **Command timeout**.
- **Healthy threshold** – Lowers the threshold at which an instance is considered healthy during rolling deployments, rolling updates, and immutable updates.
- **Command timeout** – The number of seconds to wait for an instance to become healthy before canceling the deployment or, if **Ignore health check** is set, to continue to the next batch.

Accept the default values here and click **Apply**.



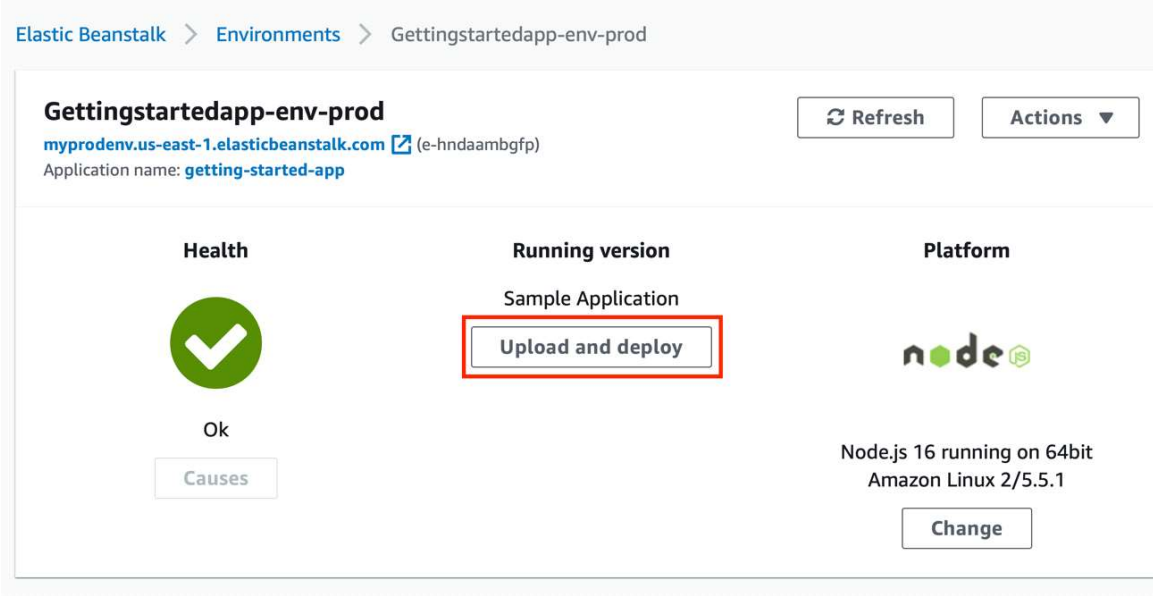
- Thereafter, Elastic Beanstalk will start updating your environment.



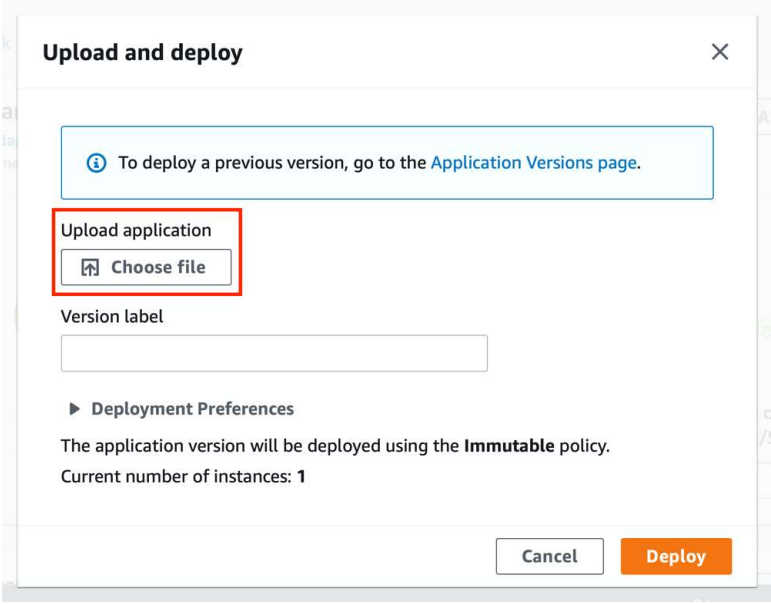
- After the update is completed, you will be pushing the new version of your web application. You can deploy a new version at any time, as long as no other update operations are in progress on your environment.

Click the following link to download the application source bundle:
https://tinyurl.com/4x5rfam8

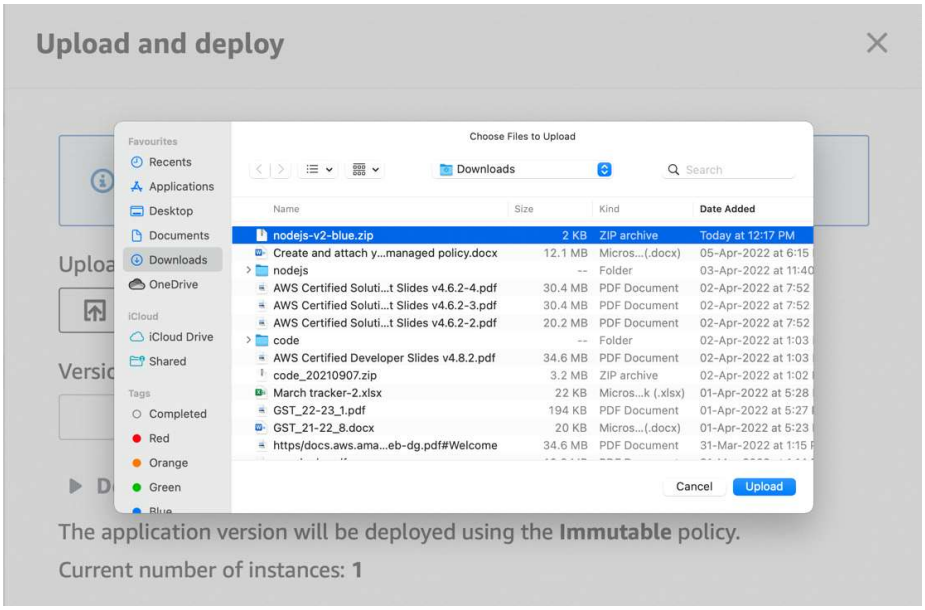
- On the environment overview page, choose **Upload and deploy**.



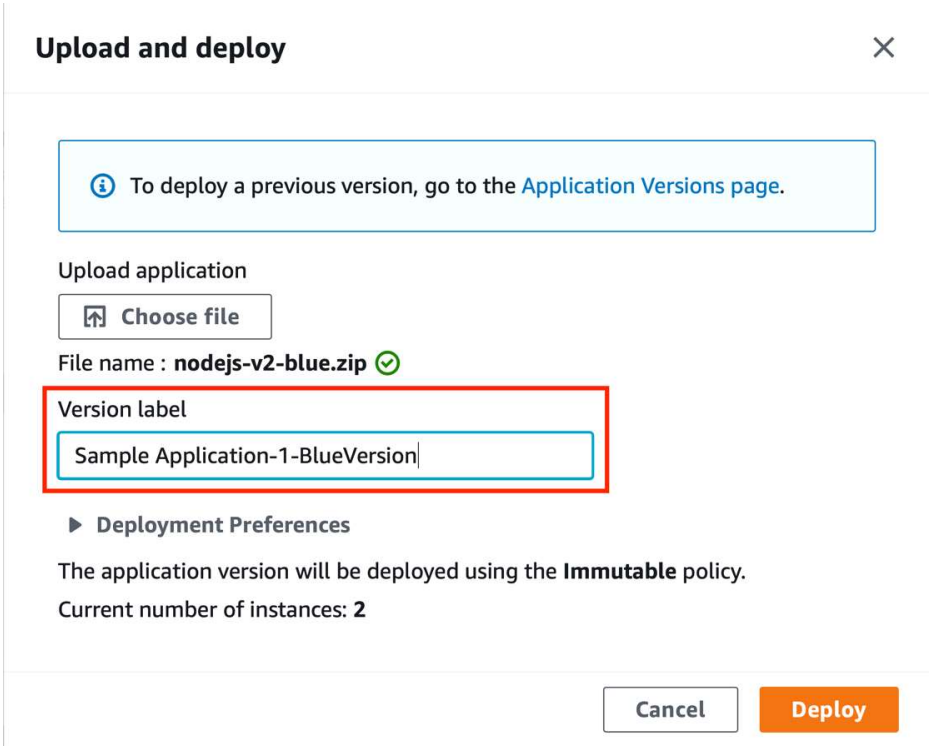
- Choose **Choose file**, and then upload the application source bundle that you downloaded.



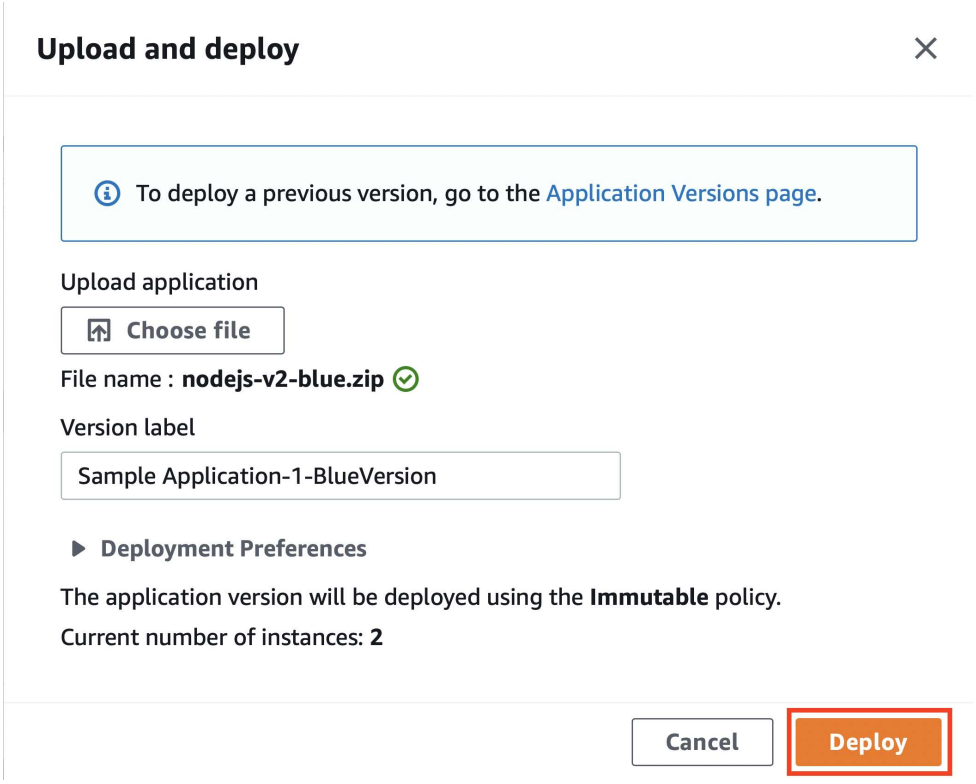
Ensure that the entire ZIP folder is selected for uploading.



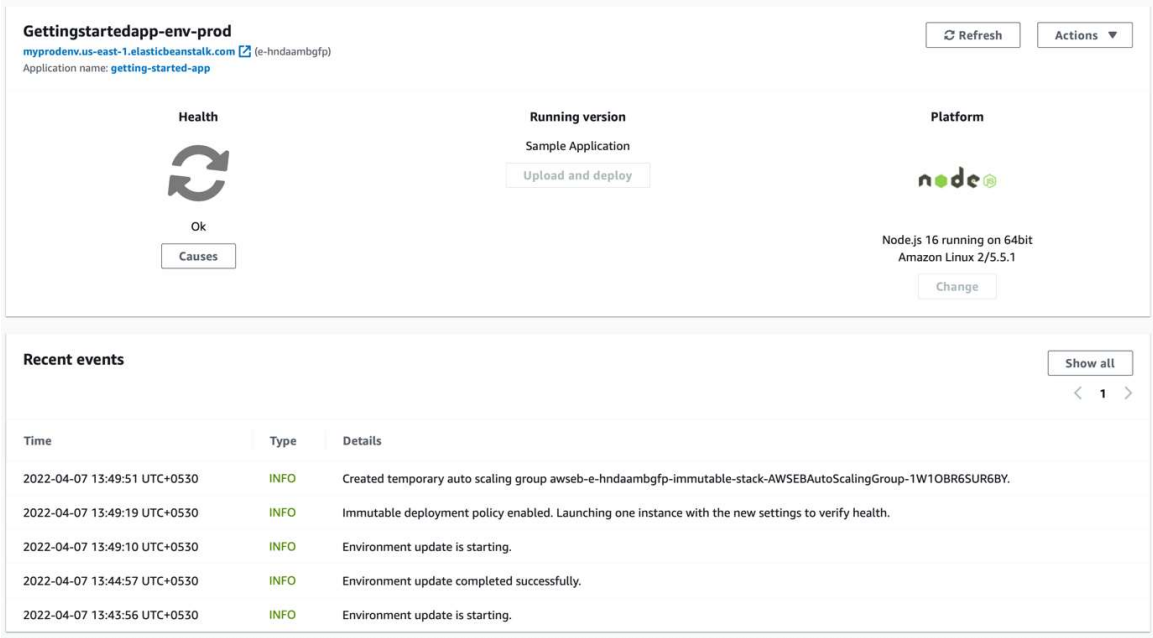
- The console automatically fills in the **Version label** with a new unique label. If you type in your own version label, ensure that it's unique.



- Choose **Deploy**.

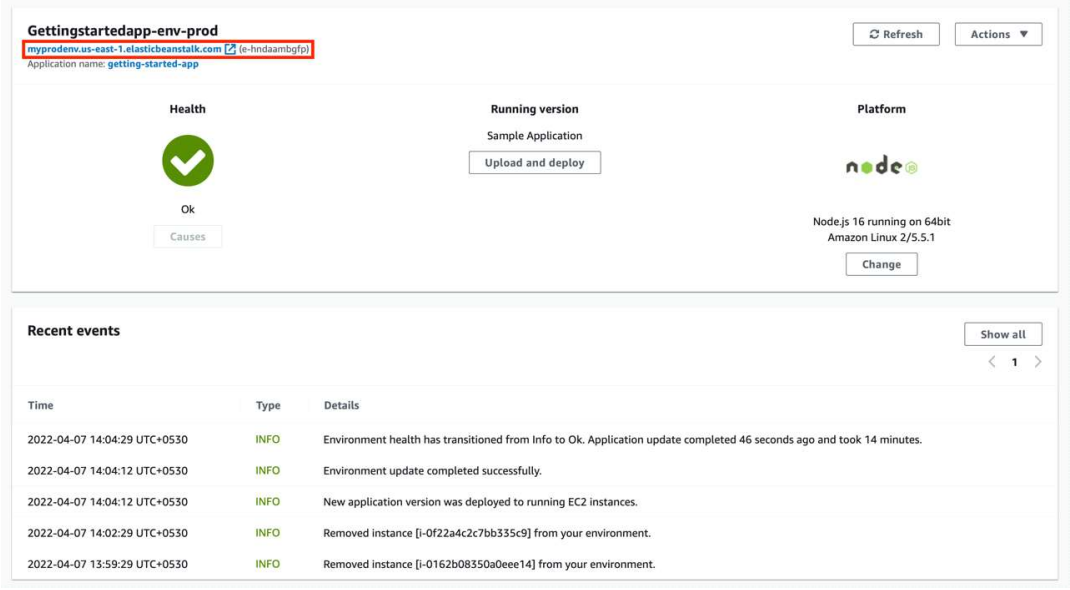


- While Elastic Beanstalk deploys your file to your Amazon EC2 instances, you can view the deployment status on the environment's overview.



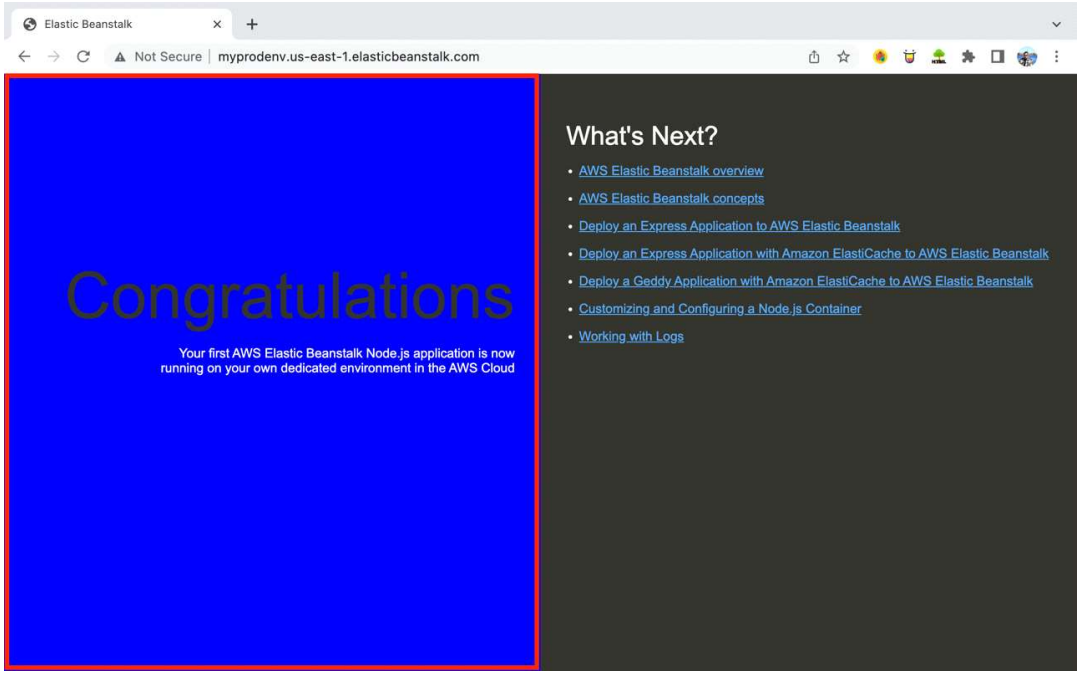
While the application version is updated, the Environment Health status is grey. When the deployment is complete, Elastic Beanstalk performs an application health check. When the application responds to the health check, it's considered healthy and the status returns to green. The environment overview shows the new Running Version—the name you provided as the Version label.

- Click on the environment URL to access the new version of your website.



- This is the expected outcome of the immutable update. The green colored

column of your web page is now changed to blue.

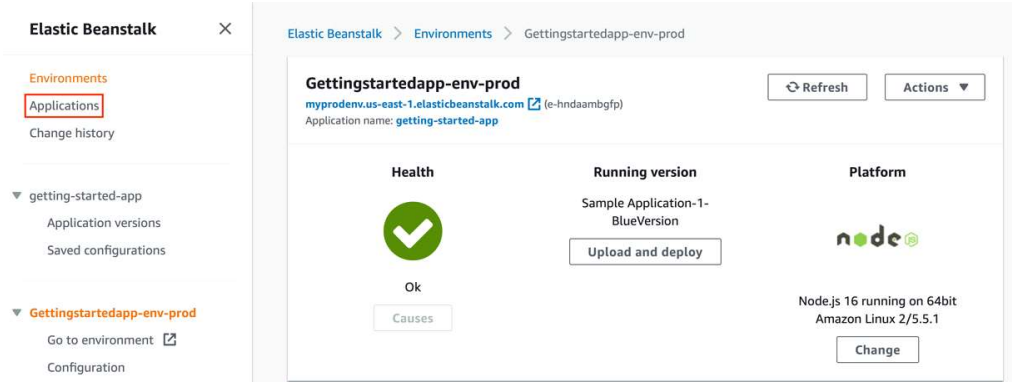


Step 6: Clean up

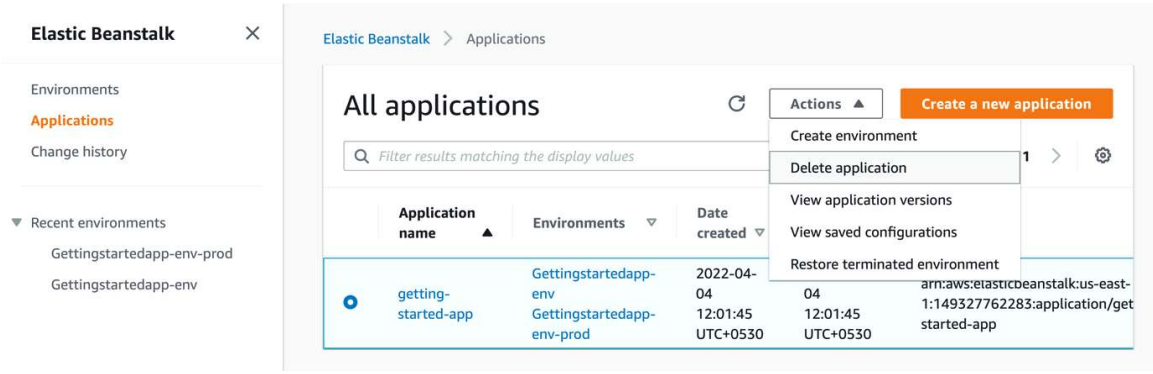
Congratulations! You have successfully deployed a sample application to the AWS Cloud, created a new environment while modifying its configuration, and uploaded a new version by pushing an immutable environment update.

To ensure that you're not charged for any services you aren't using, delete all application versions and terminate the environment. This also deletes the AWS resources that the environment created for you.

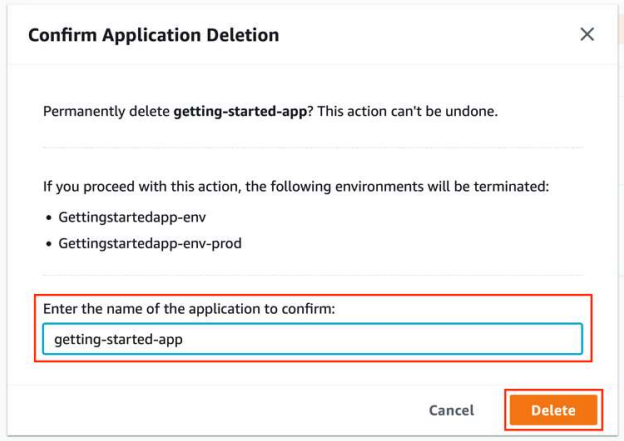
- In the navigation pane, choose **Applications**.



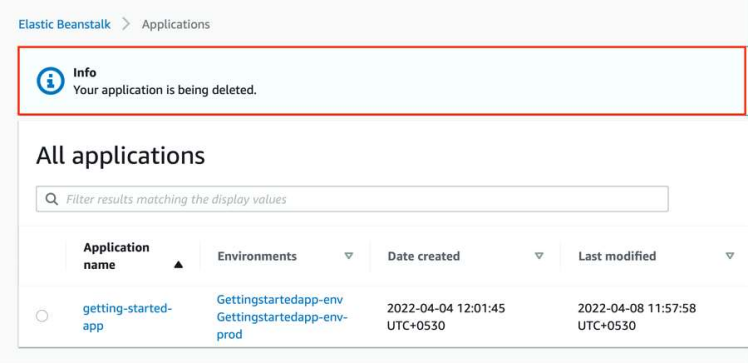
- Select the application you've deployed, go to **Actions,** and click **Delete Application**.



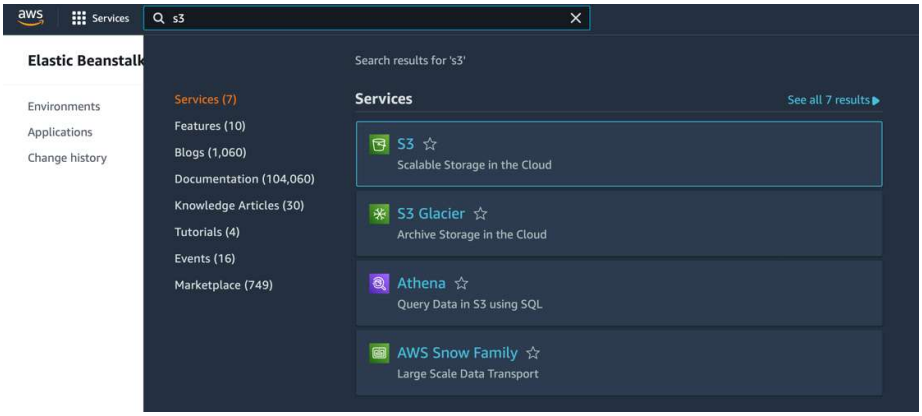
- Once prompted, enter the name of the application, and click **Delete**.



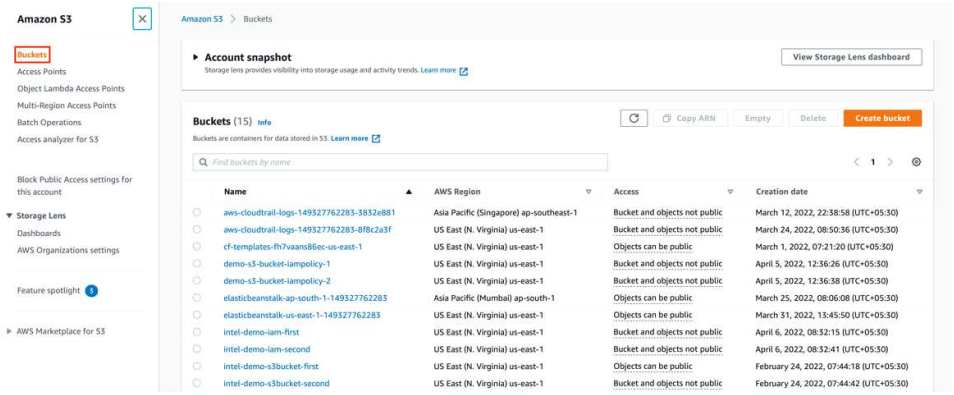
- This will initiate the application deletion process. Consequently, both environments and resources line EC2 instances, Load Balancers and Auto Scaling groups etc. (excluding S3 bucket) will get terminated or removed permanently.



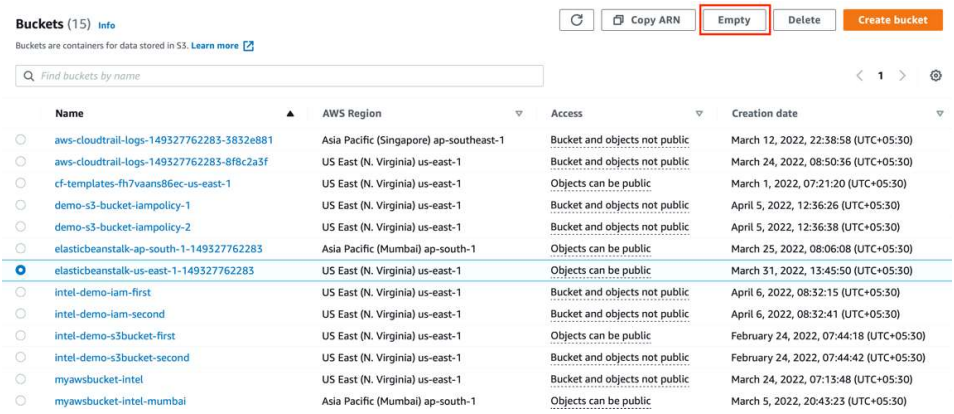
- To delete Amazon S3 bucket, enter keyword **S3** in search menu bar.



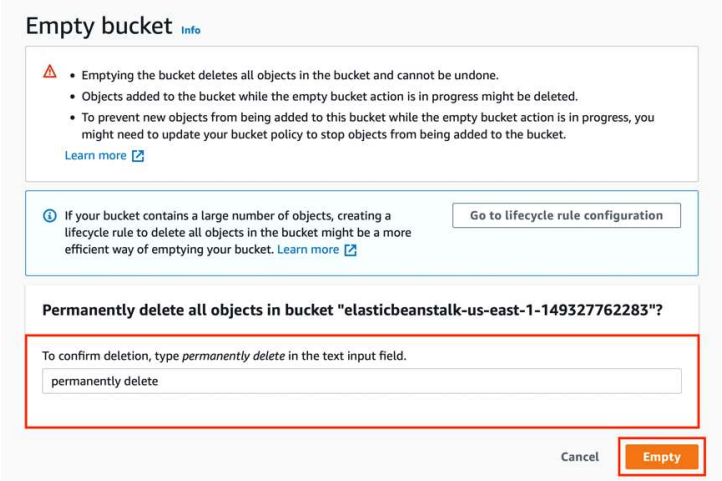
- Click **Buckets** to retrieve the list of Amazon S3 buckets.



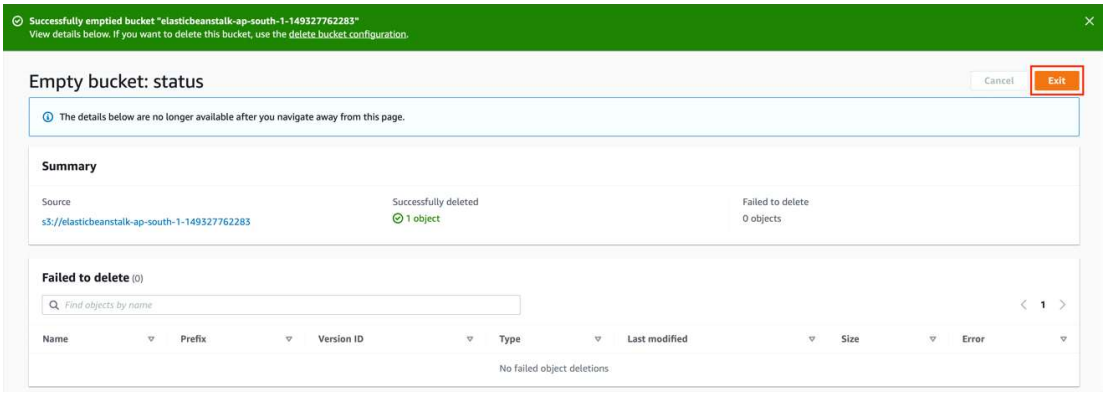
- Select the S3 bucket linked with your Elastic Beanstalk application and click **Empty** to remove all the environments' data.



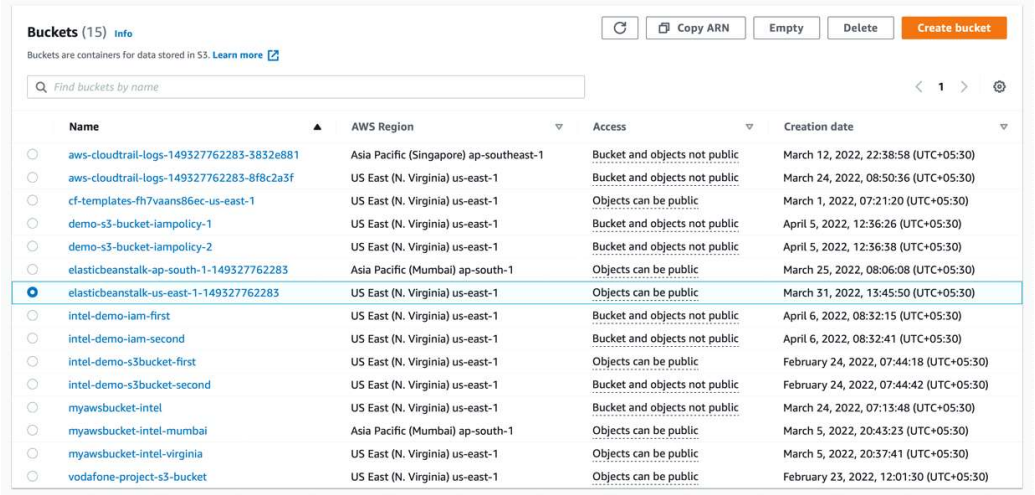
- To confirm deletion, type *permanently delete* in the text input field, and click **Empty**.



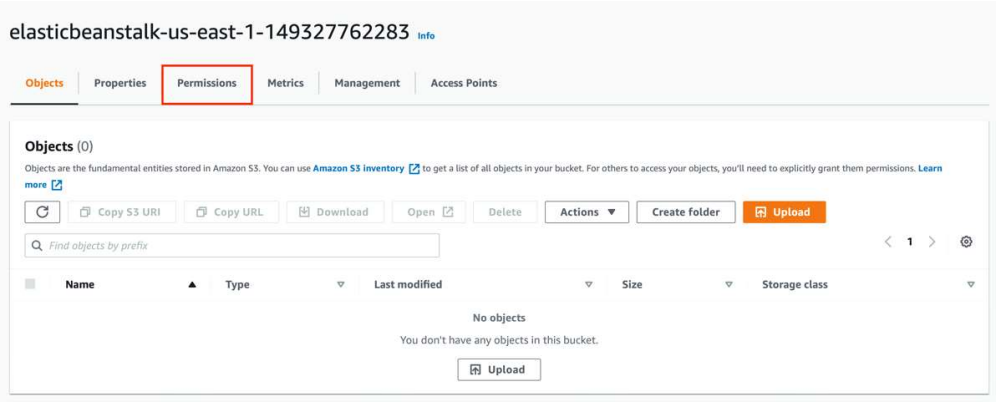
- Once the bucket is emptied, click **Exit**.



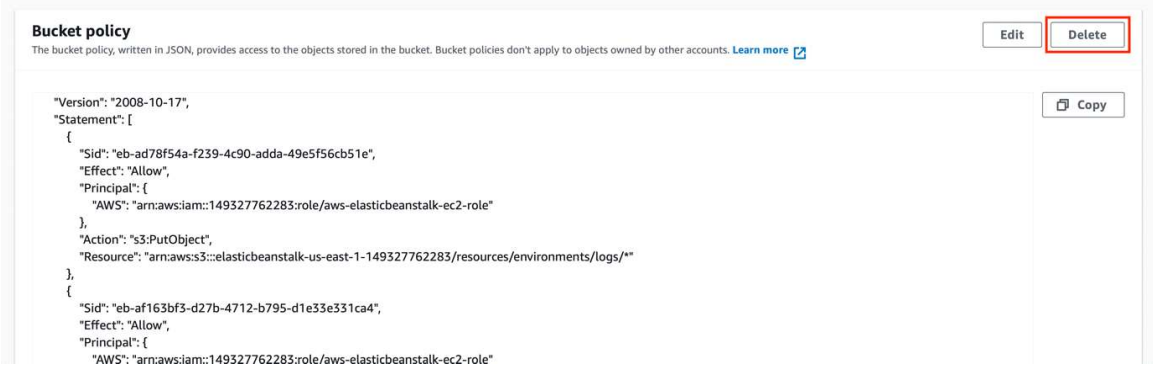
- You will be landed back to the S3 buckets list. Click on the same S3 bucket we just emptied.



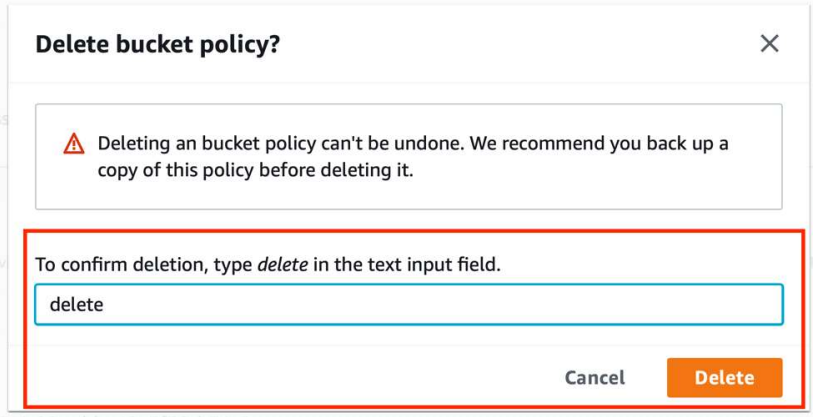
- Go to **Permissions**.



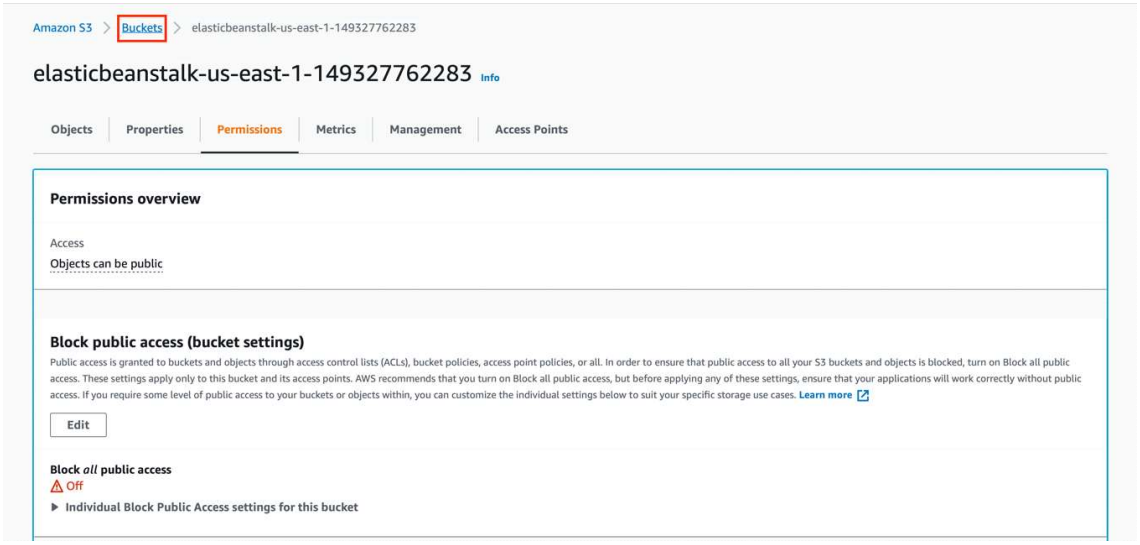
- Scroll down while looking for **Bucket policy** and click **Delete**.



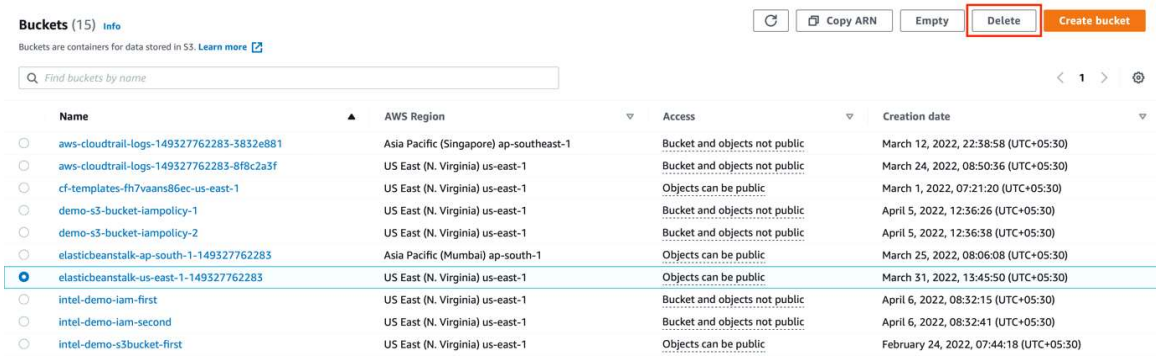
- To confirm deletion, type *delete* in the text input field and click **Delete**. This will ensure that you can remove your S3 bucket without any restriction.



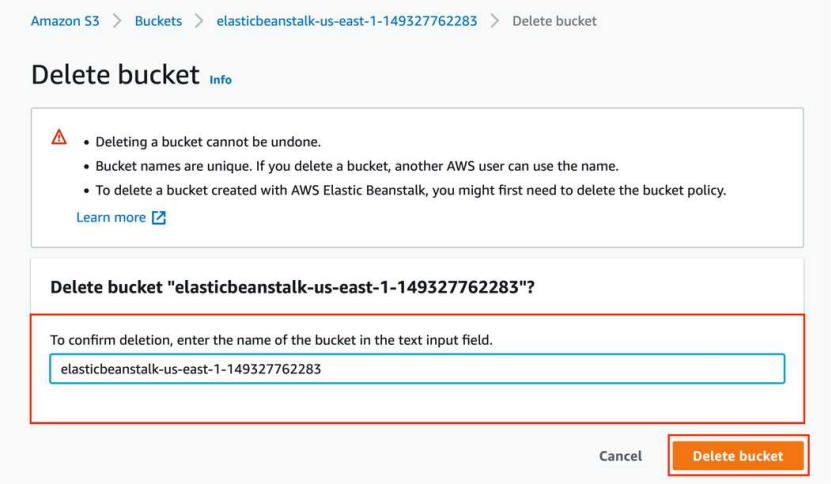
- Click **Buckets** to go back and access the buckets' list.



- Select the S3 bucket now and click **Delete**.



- To confirm deletion, enter the name of the bucket in the text input field and click **Delete Bucket**.



- Finally, the bucket will be deleted successfully.