Immersion Day

Getting Started with Elastic Beanstalk

April 2020

Table of Contents

Overview

AWS Elastic Beanstalk is an even easier way for you to quickly deploy and manage applications in the AWS cloud. You simply upload your application and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, auto-scaling and application health monitoring.

At the same time, with Elastic Beanstalk, you retain full control over the AWS resources powering your application, and you can access the underlying resources at any time. Elastic Beanstalk leverages AWS services such as Amazon Elastic Cloud Compute (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Simple Notification Service (Amazon SNS), Elastic Load Balancing, and Auto Scaling to deliver the same highly reliable, scalable, and cost-effective infrastructure that hundreds of thousands of businesses depend on today.

For this lab you will build a simple, scalable web-based customer signup form that is deployed to AWS Elastic Beanstalk. The application stores data in Amazon DynamoDB and publishes notifications to the Amazon Simple Notification Service (SNS) when a customer fills out the form.

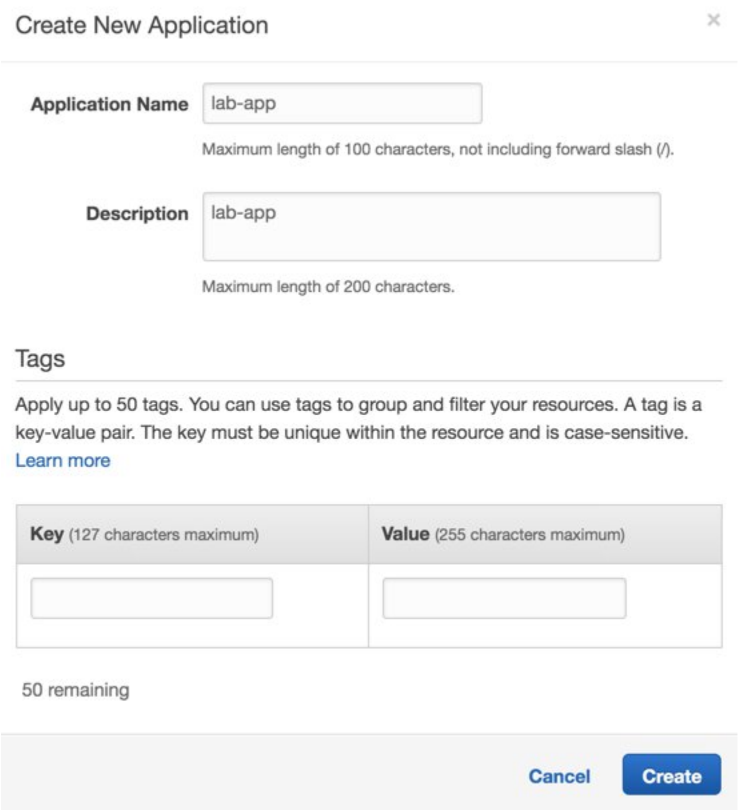
This lab will walk you through the following:

- Launch an Elastic Beanstalk Environment
- Add Permissions to Your Environment's Instances
- Deploy the Sample Application
- Update the Application's Configuration Files
- Configure Your Environment for High Availability
- Clean Up

Launch an Elastic Beanstalk Environment

AWS Elastic Beanstalk makes it easy to create new environments for your application. You can create and manage separate environments for development, testing, and production use, and you can deploy any version of your application to any environment.

1. Download the sample application source bundle from GitHub: eb-node-express-sample-v1.1.zip (https://bit.ly/2pubq2c)
2. Sign in to the AWS Management Console and open the AWS Elastic Beanstalk console at https://console.aws.amazon.com/elasticbeanstalk
3. Click **Create New Application** on the top right**.** The **Create New Application** dialog appears.



Create New Application

Application Name: lab-app

Maximum length of 100 characters, not including forward slash (/).

Description: lab-app

Maximum length of 200 characters.

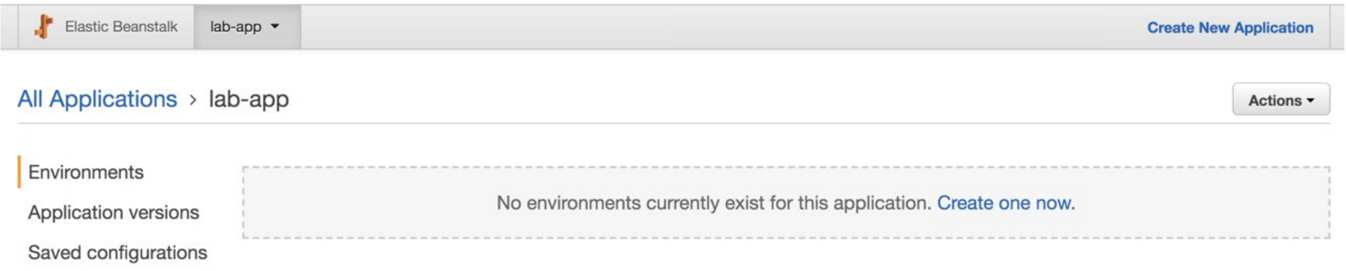
Tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. Learn more

Key (127 characters maximum)	Value (255 characters maximum)

50 remaining

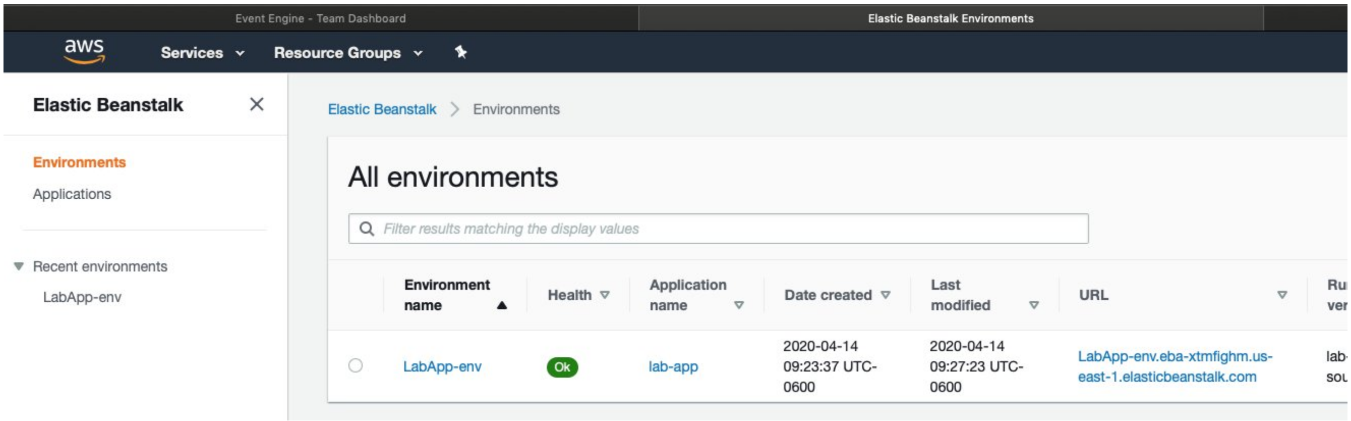
4. In the **Application name** field, type a name (e.g. **lab-app**).
5. Leave the Tags fields empty and click **Create**.



No environments currently exist for this application. Create one now.

6. For **Platform**, Choose Node.js and leave **Branch** and **Version** as the defaults
7. For **Application Code**, choose **Upload** your code and select the .zip file you downloaded in step 1 above
8. Click the orange **Create application** button.

9. It will take a few minutes for the application environment to be created. You can monitor progress under “Environments” by clicking on the name of your environment e.g. LabApp-Env
10. Once deployment has completed, the Health column will show a green OK as shown below:


Event Engine - Team Dashboard
Elastic Beanstalk Environments
aws
Services
Resource Groups
Elastic Beanstalk
Environments
Applications
Recent environments
LabApp-env
Elastic Beanstalk > Environments
Filter results matching the display values

All environments

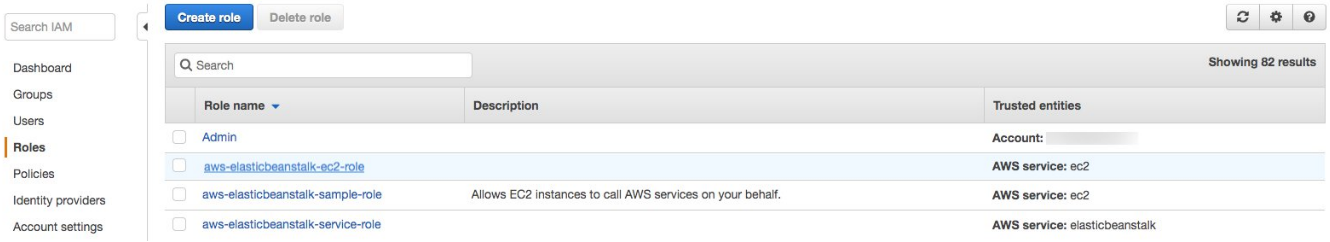
	Environment name ▲	Health ▽	Application name ▽	Date created ▽	Last modified ▽	URL ▽	Ru ver
○	LabApp-env	Ok	lab-app	2020-04-14 09:23:37 UTC-0600	2020-04-14 09:27:23 UTC-0600	LabApp-env.eba-xtmfighm.us-east-1.elasticbeanstalk.com	lab- sou

11. You can visit your deployed application by clicking on the URL in the URL column (recommend opening in a new tab).

Add Permissions to Your Environment’s Instances

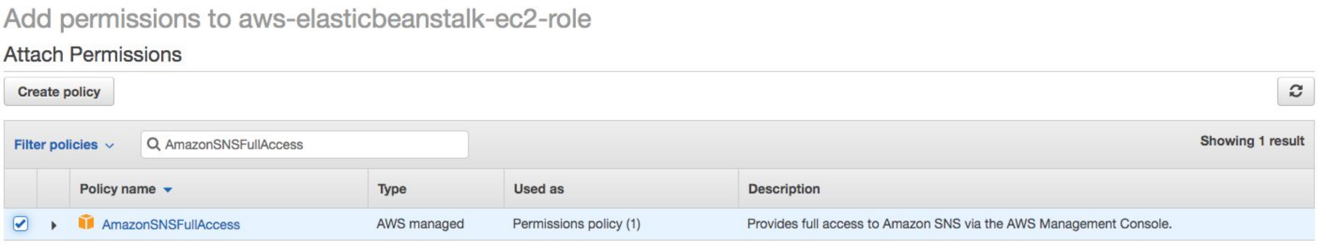
The sample application uses instance permissions to write data to a DynamoDB table, and to send notifications to an Amazon SNS topic with the SDK for JavaScript in Node.js. Add the following managed policies to the default instance profile to grant the EC2 instances in your environment permission to access DynamoDB and Amazon SNS:

1. In the **AWS Management Console**, on the **Services** menu, under **Security, Identity & Compliance** headline, click **IAM.**
2. Click **Roles** in the left navigation.
3. Click **aws-elasticbeanstalk-ec2-role** in the list



	Role name	Description	Trusted entities
□	Admin		Account: [redacted]
□	aws-elasticbeanstalk-ec2-role		AWS service: ec2
□	aws-elasticbeanstalk-sample-role	Allows EC2 instances to call AWS services on your behalf.	AWS service: ec2
□	aws-elasticbeanstalk-service-role		AWS service: elasticbeanstalk

4. On the Permissions tab, click **Attach policies**.
5. Select the managed policies for the additional services that your application uses: **AmazonSNSFullAccess** and **AmazonDynamoDBFullAccess**. You can search for these policies by typing their names into **Search** field. After finding them, click the checkbox.

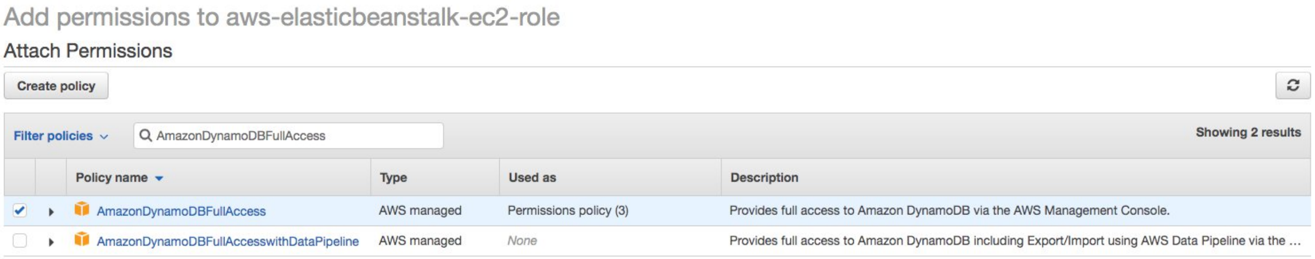


Add permissions to aws-elasticbeanstalk-ec2-role

Attach Permissions

Search: AmazonSNSFullAccess

		Policy name	Type	Used as	Description
☑	▸	AmazonSNSFullAccess	AWS managed	Permissions policy (1)	Provides full access to Amazon SNS via the AWS Management Console.



Add permissions to aws-elasticbeanstalk-ec2-role

Attach Permissions

Search: AmazonDynamoDBFullAccess

		Policy name	Type	Used as	Description
☑	▸	AmazonDynamoDBFullAccess	AWS managed	Permissions policy (3)	Provides full access to Amazon DynamoDB via the AWS Management Console.
□	▸	AmazonDynamoDBFullAccesswithDataPipeline	AWS managed	None	Provides full access to Amazon DynamoDB including Export/Import using AWS Data Pipeline via the ...

6. Click **Attach policy**.

Update the Application's Configuration File

In this step, we will update the configuration file in the application source to define our email address.

Linux/Mac:

1. Extract the project files from the source bundle:

```
~$ mkdir nodejs-tutorial
~$ cd nodejs-tutorial
~/nodejs-tutorial$ unzip ~/Downloads/eb-node-express-sample-v1.1.zip
```

2. Open **.ebextensions/options.config** and change the value of the following setting:

    - **NewSignupEmail** – Your email address.

    This configures the email address that the Amazon SNS topic uses for notifications.

3. Create a source bundle from the modified code.

```
~/nodejs-tutorial$ zip nodejs-tutorial.zip -r * .[^.]*
```

Windows PowerShell:

1. Extract the project files from the source bundle:

```
PS D:\Users\workshop\Downloads> mkdir nodejs-tutorial
PS D:\Users\workshop\Downloads> cd nodejs-tutorial
PS D:\Users\workshop\Downloads\nodejs-tutorial> Expand-Archive -Path
D:\Users\workshop\Downloads\eb-node-express-sample-v1.1.zip -DestinationPath .
```

2. Open **.ebextensions/options.config** and change the value of the following setting:

```
PS D:\Users\workshop\Downloads\nodejs-tutorial> powershell_ise.exe .\.ebextensions\options.config
```

   - **NewSignupEmail** – Your email address.

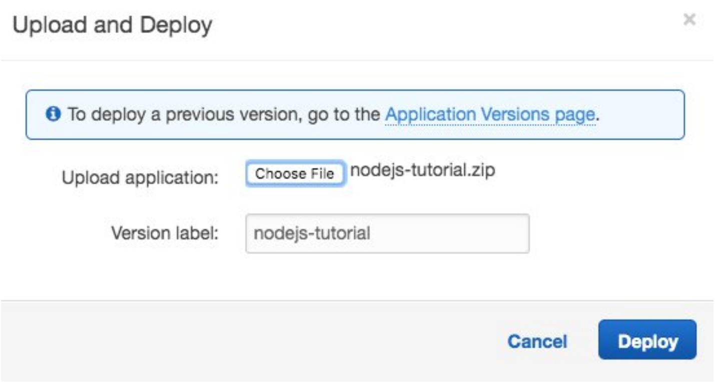
   This configures the email address that the Amazon SNS topic uses for notifications.

3. Create a source bundle from the modified code.

```
PS D:\Users\workshop\Downloads\nodejs-tutorial> Compress-Archive -Path
D:\Users\workshop\Downloads\nodejs-tutorial\* -DestinationPath
D:\Users\workshop\Downloads\nodejs-tutorial\nodejs-tutorial
```

Deploy the nodejs-tutorial.zip Source Bundle to Your Environment

1. On the AWS **Services** menu, under **History** headline, click **Elastic Beanstalk.**
2. Click on the environment (LabApp-env) created for lab-app.
3. Click **Upload and Deploy.**
4. Click **Choose File.**



Upload and Deploy

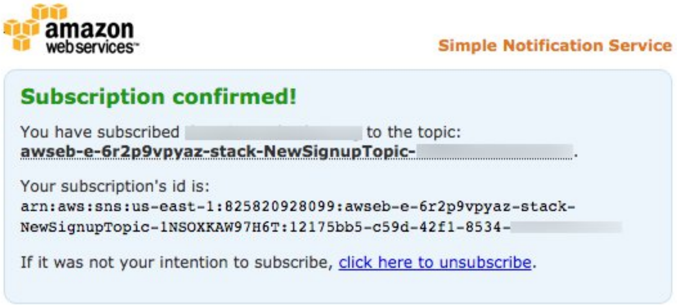
To deploy a previous version, go to the Application Versions page.

Upload application: Choose File nodejs-tutorial.zip

Version label: nodejs-tutorial

5. Browse for the application zip file you just created (**nodejs-tutorial.zip**) and click **Deploy.** The deployment will take a few minutes to complete. You can monitor the progress on the dashboard page.

6. You will receive an email with subject **AWS Notification - Subscription Confirmation.** Click the **Confirm subscription** link in the email body. When you deploy the application, Elastic Beanstalk updates the configuration of the Amazon SNS topic.



Subscription confirmed!

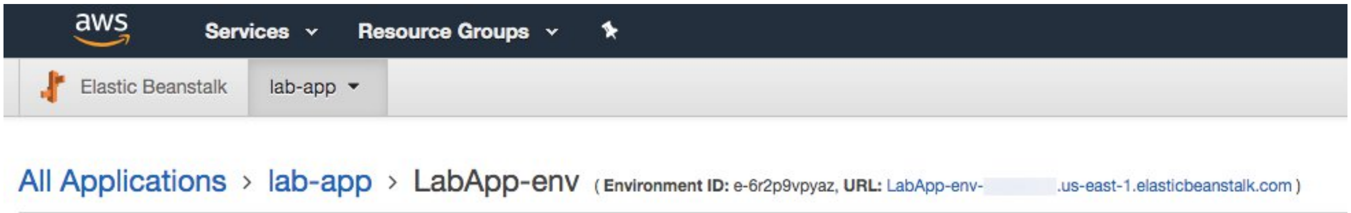
You have subscribed [redacted] to the topic:
awseb-e-6r2p9vpyaz-stack-NewSignupTopic-[redacted].

Your subscription's id is:
`arn:aws:sns:us-east-1:825820928099:awseb-e-6r2p9vpyaz-stack-NewSignupTopic-1NSOXKAW97H6T:12175bb5-c59d-42f1-8534-`[redacted]

If it was not your intention to subscribe, click here to unsubscribe.

View & Test the Application

1. On the AWS **Services** menu, under **History** headline, click **Elastic Beanstalk.**
2. Click on the environment (LabApp-env) created for lab-app.
3. You will see the URL of your application at the top. Click on the URL to open the application.



All Applications › lab-app › LabApp-env (**Environment ID:** e-6r2p9vpyaz, **URL:** LabApp-env-[redacted].us-east-1.elasticbeanstalk.com)

4. Click the **Sign up today** button and fill out the form. You should receive an email message for each entry you submit.

View the DynamoDB Table

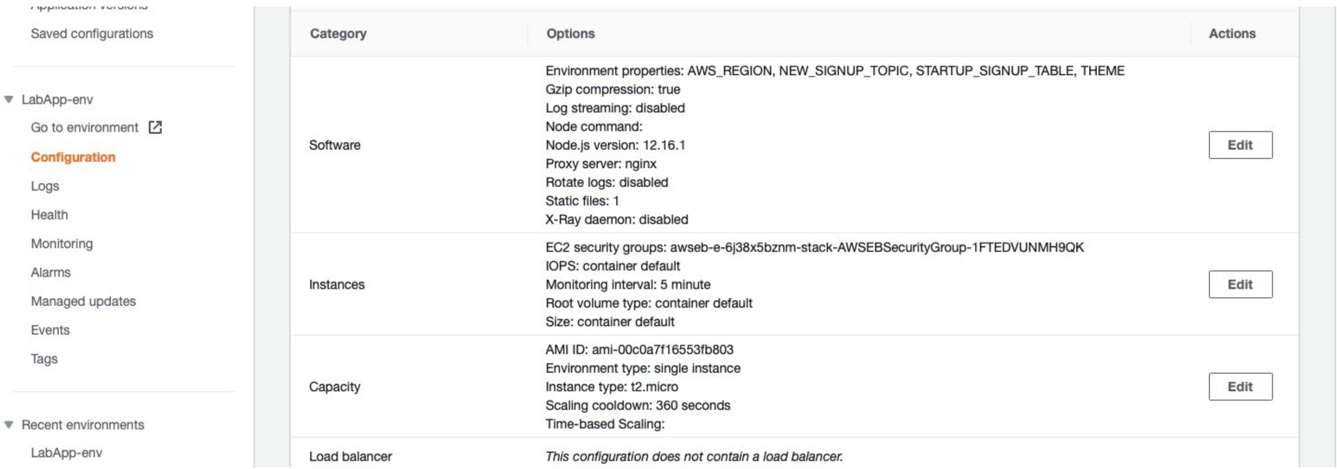
You can add AWS Elastic Beanstalk configuration files (.ebextensions) to your web application's source code to configure your environment and customize the AWS resources that it contains. The sample application includes configuration files **(.ebextensions/create-dynamodb-table)** that create the DynamoDB table used by the application.

1. Open the Tables page in the DynamoDB console.
2. Find the table that contains **StartupSignupsTable** text in its name.
3. Select the table, choose the **Items** tab, and then click **Start search** to view all items in the table.
4. To get more items in the table, go back to the web application and fill out the form again. Then come back to the DynamoDB table and refresh the page.

Configure Your Environment for High Availability

Finally, configure your environment's Auto Scaling group with a higher minimum instance count. Run at least two instances at all times to prevent the web servers in your environment from being a single point of failure and to allow you to deploy changes without taking your site out of service.

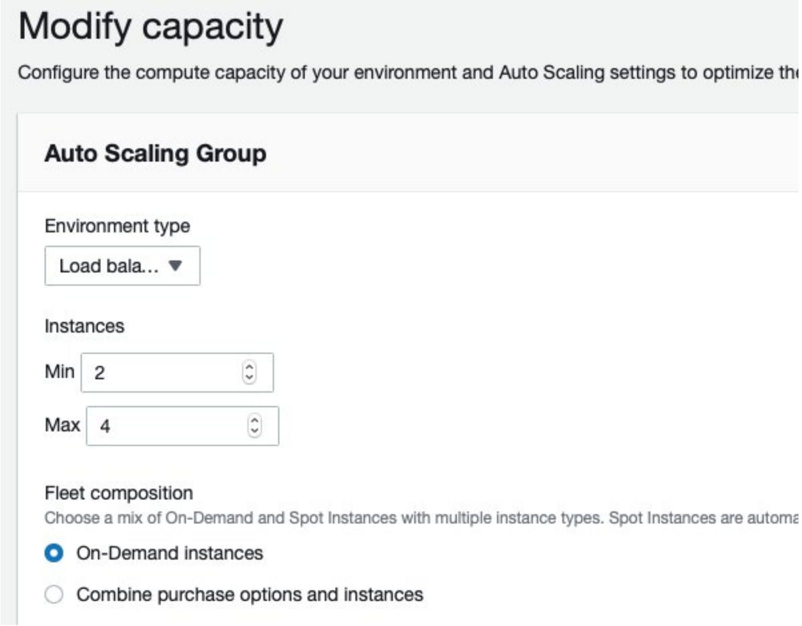
1. On the AWS **Services** menu, under **History** headline, click **Elastic Beanstalk.**
2. Click on the environment (**LabApp-env**) created for lab-app.
3. Choose **Configuration** from the menu on the left.
4. In the **Capacity** section, click the **Edit** link.



Category	Options	Actions
Software	Environment properties: AWS_REGION, NEW_SIGNUP_TOPIC, STARTUP_SIGNUP_TABLE, THEME Gzip compression: true Log streaming: disabled Node command: Node.js version: 12.16.1 Proxy server: nginx Rotate logs: disabled Static files: 1 X-Ray daemon: disabled	Edit
Instances	EC2 security groups: awseb-e-6j38x5bznm-stack-AWSEBSecurityGroup-1FTEDVUNMH9QK IOPS: container default Monitoring interval: 5 minute Root volume type: container default Size: container default	Edit
Capacity	AMI ID: ami-00c0a7f16553fb803 Environment type: single instance Instance type: t2.micro Scaling cooldown: 360 seconds Time-based Scaling:	Edit
Load balancer	*This configuration does not contain a load balancer.*	

5. In the Auto Scaling Group section, configure the following settings.

Environment type – Select **Load balanced**.
Min instances – 2



Modify capacity

Configure the compute capacity of your environment and Auto Scaling settings to optimize th

Auto Scaling Group

Environment type: Load bala... ▼

Instances

Min: 2

Max: 4

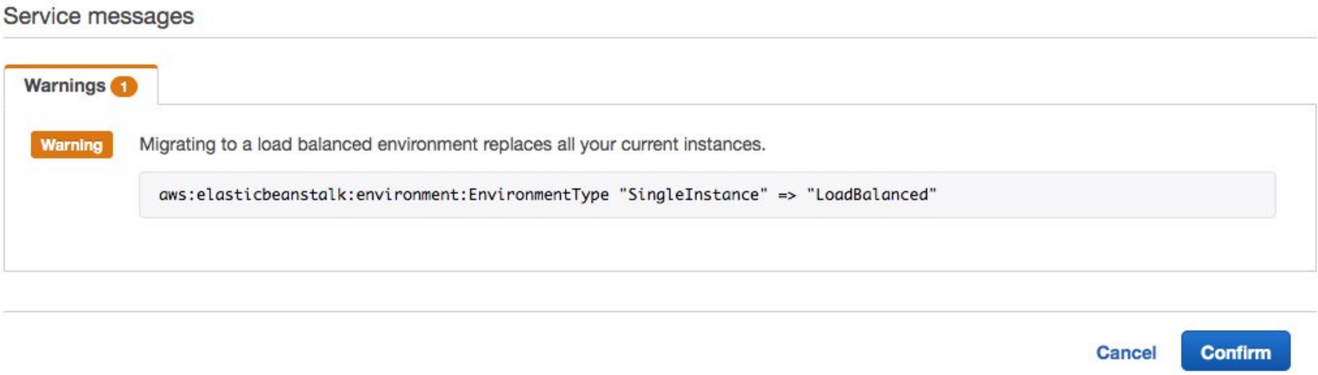
Fleet composition

Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automa

● On-Demand instances

○ Combine purchase options and instances

6. Scroll to the bottom, click **Apply**, then click **Confirm**.



Service messages

Warnings 1

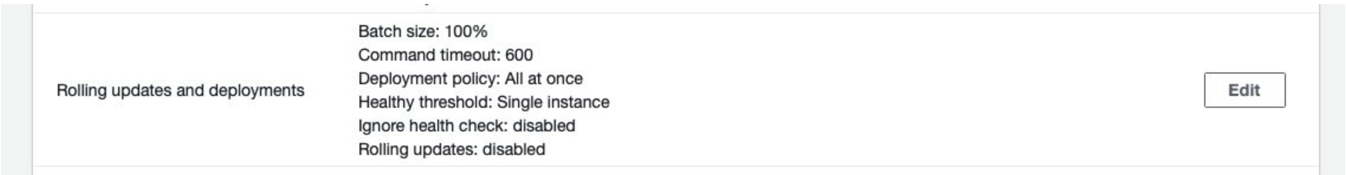
Warning Migrating to a load balanced environment replaces all your current instances.

```
aws:elasticbeanstalk:environment:EnvironmentType "SingleInstance" => "LoadBalanced"
```

7. Wait until Elastic Beanstalk is done with updating your environment.
8. Choose **Configuration** from the left navigation and note the **Capacity** section has changed.

Enable rolling deployments

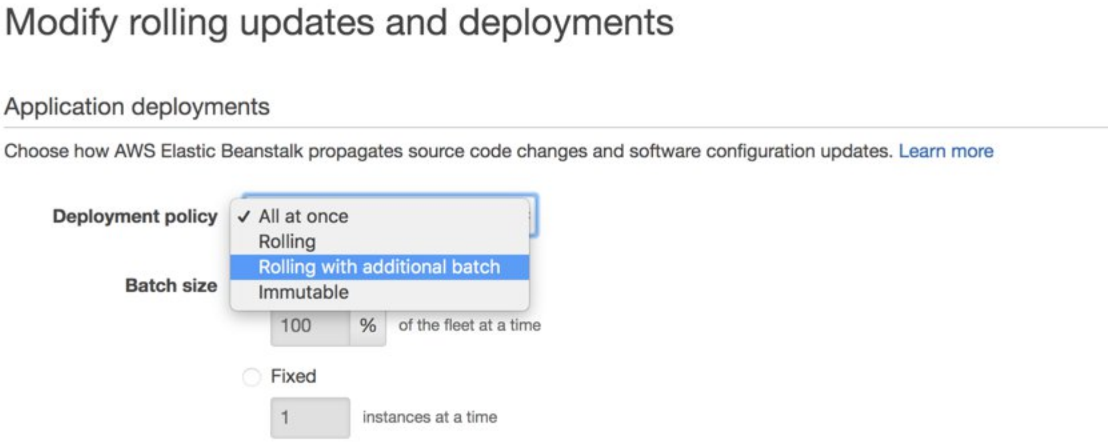
1. While on the Configuration page, click **Edit** in the **Rolling updates and deployments** section.



Rolling updates and deployments

Batch size: 100%
Command timeout: 600
Deployment policy: All at once
Healthy threshold: Single instance
Ignore health check: disabled
Rolling updates: disabled

2. For Deployment policy, select **Rolling with additional batch**.
3. Scroll to the bottom and click **Apply**.



Modify rolling updates and deployments

Application deployments

Choose how AWS Elastic Beanstalk propagates source code changes and software configuration updates. Learn more

Deployment policy: ✓ All at once

Batch size: 100 % of the fleet at a time

○ Fixed

1 instances at a time

4. Click **Apply configuration**

Perform another deployment

(Refer to Section: [Deploy the nodejs-tutorial.zip Source Bundle to Your Environment]) to verify your deployment setup is as expected. Click on: EC2, load balancer, and auto scaling dashboards to see this reflected in the EC2 Services menu. Depending on your timing, you'll see the Auto Scaling group move to a desired number of instances of 4, then you'll see 4 instances running. After the software deployment completes the Auto Scaling group will return to 2 instances and the 2 older instances will show as terminated. During this deployment, your application will remain online and continue accepting new registrations.

Note:

Elastic Beanstalk creates an Amazon S3 bucket named **elasticbeanstalk-***region-account-id* for each region in which you create environments. Elastic Beanstalk uses this bucket to store objects, including deployment packages and temporary configuration files, that are required for the proper operation of your application.

You can see these objects by going to the S3 console and inspecting the contents of the bucket beginning with the name **elasticbeanstalk-**.

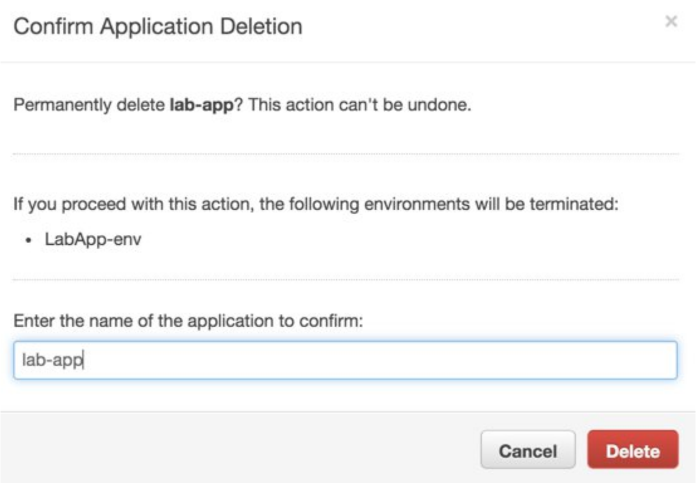
Clean Up

To terminate your Elastic Beanstalk environment:

When you finish working with Elastic Beanstalk, you can terminate your environment. Elastic Beanstalk terminates all AWS resources associated with your environment, such as Amazon EC2 instances, database instances, load balancers, security groups, and alarms.

1. On the AWS **Services** menu, under **History** headline, click **Elastic Beanstalk**.

2. Choose **Actions**, and then choose **Delete application**.
3. In the **Confirm Application Deletion** dialog, enter **lab-app** in the application field and click **Delete**.



Confirm Application Deletion

Permanently delete **lab-app**? This action can't be undone.

If you proceed with this action, the following environments will be terminated:

- LabApp-env

Enter the name of the application to confirm:

lab-app

4. Click on the environment (LabApp-env) to view progress of the deletion.

To remove instance profile permissions:

When you create resources outside of Elastic Beanstalk, it is completely independent of Elastic Beanstalk and your Elastic Beanstalk environments, and will not be terminated by Elastic Beanstalk.

1. Open the **Roles** page in the IAM console.
2. Choose **aws-elasticbeanstalk-ec2-role.**

3. Remove the following AWS managed policies you attached earlier (Refer to Section: **Add Permissions to Your Environment's Instances**).

    - AmazonSNSFullAccess
    - AmazonDynamoDBFullAccess

Conclusion

In this lab you have walked through the process of deploying a sample Node.js application that uses the AWS SDK for JavaScript in Node.js to interact with Amazon DynamoDB. You learned basic operations to deploy and update an application using AWS Elastic Beanstalk. Finally, you learned how to configure your environment for high availability.