

This Lab covers:

- How to build an Elastic Beanstalk Application container and environments
- The efficient way to perform a rolling deployment on underlying EC2 resources
- Blue-Green deployment with Elastic Beanstalk for zero-downtime launches
- How to mitigate cost by validating cleanup after immutable or blue-green deploys

During this lab, you will walk deployment of a new load balanced, auto scaling Todo application into an AWS VPC, upgrade the code of your app to version 2 using low-friction and rapid rolling deployment, construct a secondary deployment stack, upgrade the secondary for use as the primary, swap the environments with no downtime, and clean up the older versions of code to save money on resources.

Step 1. Create an Elastic Beanstalk App & Environment

Since we will be performing both a rolling and blue-green deploy in this lab, it makes sense to use the best tool AWS makes available for the job. For controlled deployments and efficient deployment services of code on EC2 instances, Elastic Beanstalks provides a superior interaction model and developer tools experience.

First, we navigate to the AWS Elastic Beanstalk Console, by clicking on the **Services** tab in the top-right of the Console, hovering over the **Compute** Section in the tray, then clicking on the **Elastic Beanstalk** section...



Once inside the Elastic Beanstalk Console, because the account the Lab uses has never used Elastic Beanstalk before, you will be presented with a welcome message. The most obvious button in the center of the screen is actually **NOT** the one we want. That one launches a simplified demo application immediately. Click on the **Create Application** link in the top-right of the Console view.

Welcome to AWS Elastic Beanstalk

With Elastic Beanstalk, you can **deploy**, **monitor**, and **scale** an application quickly and easily. Let us do the heavy lifting so you can focus on your business.

To deploy your **existing web application**, create an [application source bundle](#) and then [create a new application](#). If you're using Git and would prefer to use it with our command line tool, please see [Getting Started with the EB CLI](#).

To deploy a **sample application** with just one click, select a platform and click **Launch Now**.

By launching the sample application, you allow AWS Elastic Beanstalk to administer AWS resources and necessary permissions on your behalf. [Learn more](#).

Select a platform Looking for a different platform? [Let us know](#).

Launch Now

You will be immediately presented with options. The first of which are two text blanks for **Application Name** and **Description**. These are vanity names, and do not really affect anything, so you can pick whatever you like that AWS accepts. Spaces are allowed in both the name and description. Click the blue **Next** button in the bottom right of the view.

Application Info

New Environment

Application Information

To create a new application, enter the details of your application.

Application name: Must be less than 100 characters and cannot contain a /

Description: Optional.

Cancel **Next**

Next, you are presented with some options for the Environment Type you want to launch. We will be launching a **Web Server Environment**, so click on the large blue button that says **Create Web Server**.

Elastic Beanstalk

Create New Environment

Application Info

New Environment

New Environment

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web servers are standard applications that listen for and then process HTTP requests, typically over port 80. Workers are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

Web Server Environment

Provides resources for an AWS Elastic Beanstalk web server in either a single instance or load-balancing, auto scaling environment. [Learn more.](#)

Create web server

Worker Environment*

Provides resources for an AWS Elastic Beanstalk worker application in either a single instance or load-balancing, auto scaling environment. [Learn more.](#)

Create worker

* Worker environments require additional permissions to access other AWS services. [Learn more.](#)

Cancel Done

Then, you will be presented with two drop-downs, the top of which allows you to choose the platform or programming language in which your application will be authored. The code this Lab provides you is all **node.js**, so in the top dropdown, you should select this value. The second dropdown allows you to select either **Load Balancing, Auto Scaling** or **Single Instance** for the **Environment Type**. Fortunately for us, once you select **node.js** in the top dropdown, **Load Balancing, Auto Scaling** will be automatically selected - this is the value we want. Once you have configured these values and confirmed they match the image below, hit the blue **Next** button in the bottom right of the view.

Elastic Beanstalk

Create New Environment

Application Info

New Environment

Environment Type

Choose the platform and type of environment to launch.

Predefined configuration: Node.js

Looking for a different platform? [Let us know.](#)

AWS Elastic Beanstalk will create an environment running Node.js on 64bit Amazon Linux 2015.09 v2.0.8. [Change platform version.](#)

Environment type: Load balancing, auto scaling

Learn more

Cancel Previous Next

In the next screen, you are presented with a number of options relating to code deployment. The only one we will configure is the **Source** field. Please download [this zip file](#), which contains the

application we will use, then select it for upload using the **Choose File** button associated with the **Upload Your Own** option in the **Source** field under the **Application Version** heading.

We will ignore the **Deployment Preferences** for now - these controls do not matter very much for the lab. These controls configure how quickly and in what size batches the later **Rolling Updates** we'll configure will execute.

Once you have uploaded the **Source** bundle, please click the **Next** button in the bottom-right hand corner of the view to continue.

The screenshot shows the AWS Elastic Beanstalk console. The top navigation bar includes 'Services', 'Resource Groups', and 'student @ 0143-0645-5691 Oregon Support'. A 'Create New Environment' button is on the right. The main area has a left sidebar with links: Application Info, New Environment, Environment Type, Application Version (which is selected), Application Info, Additional Resources, Configuration Details, Environment Tags, Permissions, and Review Information. The 'Application Version' section title is 'Application Version'. It contains a sub-section 'Select a source for your application version' with three options: 'Sample application' (radio button), 'Upload your own (Learn more)' (radio button, selected), and 'S3 URL' (radio button). Below this is a file input field with the value 'Stogla... v1.zip'. A note '(e.g. https://s3.amazonaws.com/s3Bucket/s3Key)' is shown next to the S3 URL field. The 'Deployment Preferences' section follows, with a note: 'Elastic Beanstalk will update your application in batches so as to avoid downtime when deploying.' It includes fields for 'Deployment policy' (set to 'Rolling'), 'Healthy threshold' (set to 'Ok'), 'Ignore health check' (set to 'False'), 'Batch size' (set to 'Percentage' with '30 % of the fleet at a time'), and 'Fixed' (set to '1 instances at a time'). At the bottom right of the form are 'Cancel', 'Previous', and a blue 'Next' button.

You should now be on a view with the heading **Environment Information**. This setting will play a key role later. Note that AWS attempts to auto-populate your **Environment Name** and **Environment URL** fields for you. This will sometimes not work, because the URLs operate on a subdomain basis - so all users on all AWS regions share from the same pool of URL prefix names. Pick something memorable, as you will be using this value later. Use the **Check Availability** button to make sure the name you settle on is available for your use, then when you are ready, click on the blue **Next** button in the bottom-right of the view.

Elastic Beanstalk

Create New Environment

Application Info Environment Information

New Environment
Environment Type
Application Version
Environment Info
Additional Resources
Configuration Details
Environment Tags
Permissions
Review Information

Enter your environment information.

Environment name:

Environment URL:

Description: Optional: 200 character maximum

Check the box for **Create this environment inside a VPC**, as this Lab generated a VPC for your use, and we need to learn the extra step associated with configuring Elastic Beanstalk for use in a VPC. Do not try to **Create an RDS DB Instance with this environment**, simply click on the blue **Next** button in the bottom-right of the view.

Elastic Beanstalk

Create New Environment

Application Info Additional Resources

New Environment
Environment Type
Application Version
Additional Resources
Configuration Details
Environment Tags
VPC Configuration

Select additional resources for this environment.

Create an RDS DB Instance with this environment [Learn more](#)

Create this environment inside a VPC [Learn more](#)

You will land on a large configuration page. The only value you should set for this lab is to changing the **Application Health Check URL** value from an empty text field to a single slash, to represent the root of the Elastic Beanstalk domain's structure (/). After making sure to set this value to /, scroll down to the bottom and hit the blue **Next** button in the bottom right of the view.

Elastic Beanstalk

[Create New Environment](#)

Application Info

New Environment
Environment Type
Application Version
Environment Info
Additional Resources

Configuration Details

Modify the following settings or click **Next** to accept the default configuration. [Learn more.](#)

Instance type: **t1.micro**

Determines the processing power of the servers in your environment.

EC2 key pair: Refresh

Optional: Enables remote login to your instances.

Email address: Optional: Get notified about any major changes to your environment.

Application health check URL: Enter the relative URL that ELB continually monitors to ensure your application is available.

Enable rolling updates: Lets you control how changes to the environment's instances are propagated. [Learn more.](#)

Health

Specifies whether to wait to deploy updates and deployments according to a set period of time or instance health.

Cross zone load balancing: Enables load balancing across multiple Availability Zones. [Learn more.](#)

Connection draining: Enables the load balancer to maintain connections to an Amazon EC2 instance to complete in-progress requests while also stopping

Connection draining timeout: Maximum time that the load balancer maintains connections to an Amazon EC2 instance before forcibly closing connections.

Health Reporting

System type: Determines the health reporting type.

Root Volume (Boot Device)

Root volume type: Determines the type of storage volume to attach to instances.

Root volume size: Enables you to specify the size of the root volume.

Number of gibibytes of the root volume attached to each instance. Must be between 10 and 16384 for Provisioned IOPS (SSD) and General Purpose (SSD) root volumes and between 8 and 1024 for other root volumes.

[Cancel](#) [Previous](#) [Next](#)

In the Environment Tags section click on the **Next** button, you should now have landed on the **VPC Configuration** view. Here you must select the proper subnet values for the EC2 and ELB portions of the Elastic Beanstalk deployment. The Lab will have auto-generated subnets with the same exact same CIDR blocks, so you should put the ELB in the Subnet with a CIDR block covering the lower range of the two, as this is the public subnet's block. When you're done evaluating your work, please proceed to the next view by clicking the blue **Next** button in the bottom right.

Note: You might have to select a different VPC in order to match the one showed in the laboratory.

Application Info
New Environment
Environment Type
Application Version
Environment Info
Additional Resources
Configuration Details
Environment Tags

VPC Configuration

Permissions
Review Information

VPC Configuration

Select the VPC to use when creating your environment. [Learn more.](#)

VPC: vpc-0be3ec6e (10.0.0.0/16)

Associate Public IP Address

Select different subnets for ELB and EC2 instances in your Availability Zone.

AZ	Subnet	ELB	EC2
us-west-2a	subnet-e3aab886 (10.0.1.0/24)	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	subnet-edaab888 (10.0.0.0/24)	<input checked="" type="checkbox"/>	<input type="checkbox"/>
us-west-2b			
us-west-2c			

VPC security group: AllowAnyHTTP--sg-e27e9185

ELB visibility: External

The next screen covers Roles and Identity on AWS. Use the standard suggested Elastic Beanstalk roles for both the individual instances and the service role.

Elastic Beanstalk

Application Info
New Environment
Environment Type
Application Version
Environment Info
Additional Resources
Configuration Details
Environment Tags
VPC Configuration
Permissions
Review Information

Permissions

Select an instance profile and service role for your AWS Elastic Beanstalk environment.

An instance profile is an IAM role configured for use with EC2 instances. The instances in your Elastic Beanstalk use the credentials provided by the instance profile to communicate with AWS.

A service role allows the Elastic Beanstalk service to monitor environment resources on your behalf. See [Roles and Instance Profiles](#) in the Elastic Beanstalk developer guide for details.

Instance profile: aws-elasticbeanstalk-ec2-role

Service role: aws-elasticbeanstalk-service-role

Scroll all the way down to the bottom of the view, then hit the blue **Launch** button.

Permissions

Service role aws-elasticbeanstalk-service-role

Instance profile aws-elasticbeanstalk-ec2-role

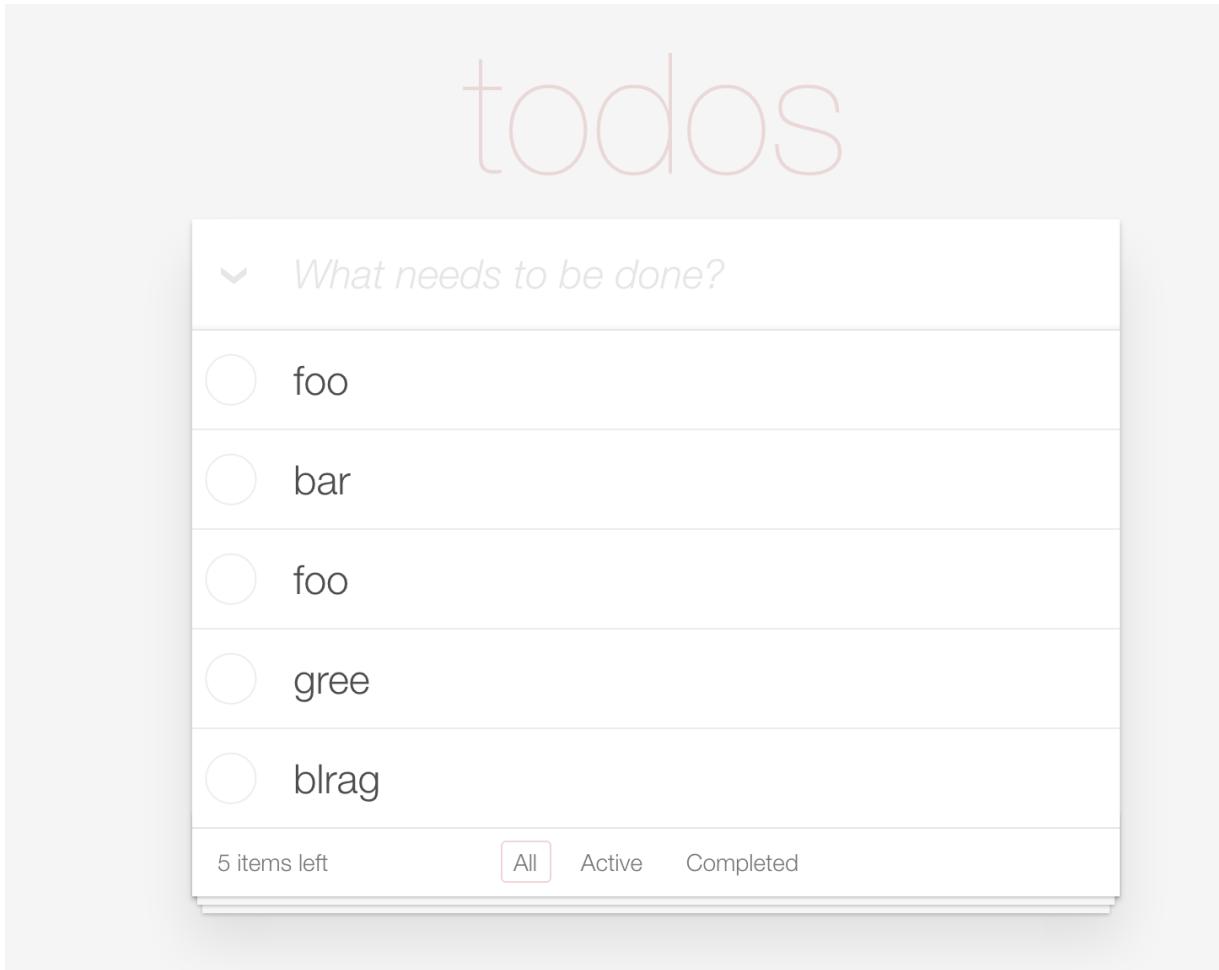
After hitting **Launch**, you should see something like the image below, substituting your own names for a couple of the fields...

The screenshot shows the Elastic Beanstalk console for the environment "reallyCoolApp-env". At the top, there is an "Info" message: "Elastic Beanstalk is now creating your environment. When it has finished it will be running First Release." On the left, a sidebar lists navigation options: Dashboard, Configuration, Logs, Health (NEW), Monitoring, Alarms, Events, and Tags. The main area is titled "Overview" and shows a large circular "refresh" icon. To the right of the icon, the "Health" status is listed as "Pending" with a "Causes" link. Below the health status, there is a "Running Version" section with a "Upload and Deploy" button. To the right of the version section, there is a "node.js" logo and a "Configuration" section showing "64bit Amazon Linux 2015.09 v2.0.8 running Node.js" with a "Change" button. A "Actions" dropdown menu is located at the top right.

... After waiting for a while, your system will transition into code green, like below.

The screenshot shows the Elastic Beanstalk console for the environment "reallyCoolApp-env". The interface is identical to the previous screenshot, but the "Health" status has changed to "Ok" with a large green checkmark icon. The rest of the interface remains the same, including the "Running Version" section, the "node.js" logo, and the "Configuration" section. The "Actions" dropdown menu is still present at the top right.

Once your system has signaled it's done launching, you should be able to enjoy the app itself. Go check it out!



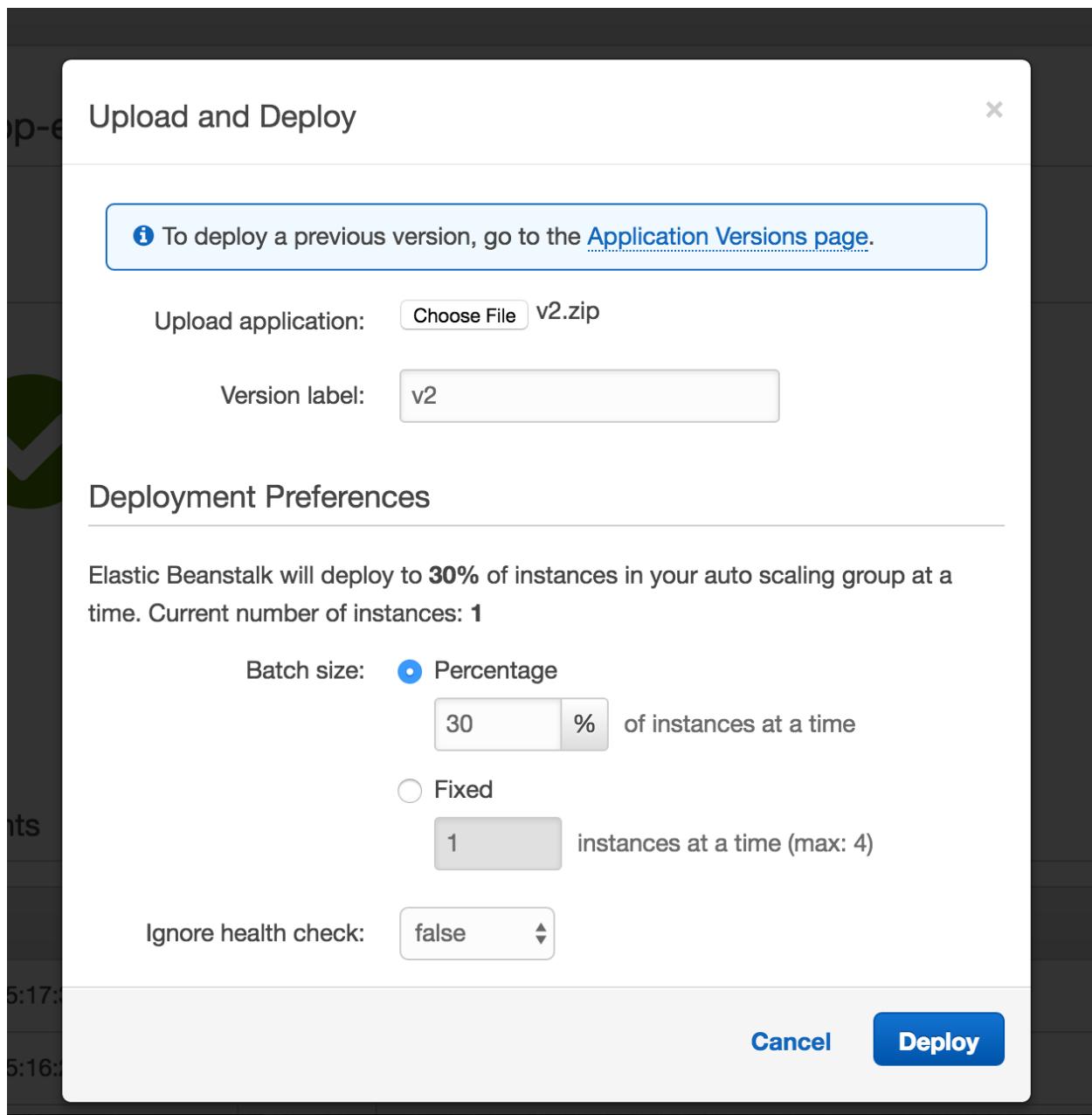
The application is a simple todo app. This Lab uses the Angular.js [Todomvc.com](#) mini app.

Step 2. Run a Rolling Deploy

Once you have had a bit of time to play with the new environment we created, return to the status screen.

The screenshot shows the AWS Elastic Beanstalk Overview page for the 'Really Cool App' environment ('reallyCoolApp-env'). The top navigation bar includes the Elastic Beanstalk logo, a 'Create New Environment' button, and an 'Actions' dropdown. The main content area displays the application name, environment ID, and URL. On the left, a sidebar lists navigation links: Dashboard, Configuration, Logs, Health (NEW), Monitoring, Alarms, Events, and Tags. The central 'Overview' section features a large green circular icon with a white checkmark, labeled 'Health' and 'OK'. Below it is a 'Causes' button. To the right, the 'Running Version' is listed as 'v1' with a 'Upload and Deploy' button. Further right is a 'node.js' logo and a 'Configuration' section showing '64bit Amazon Linux 2015.09 v2.0.8 running Node.js' with a 'Change' button. A 'Refresh' button is located at the top right of the overview section.

When the dialog opens, select [this version two code](#) and click on the **Upload and Deploy** button in the bottom-right corner of the screen.



After hitting deploy, Elastic Beanstalk will begin working on a Mutable Rolling Deploy of your code bundle. The environment will turn Gray, and will have a number of messages which amount to "please wait".

Elastic Beanstalk

Create New Environment

Really Cool App > reallyCoolApp-env (Environment ID: e-ptnvfpvys, URL: mighthavetochange.us-west-2.elasticbeanstalk.com)

Actions ▾

Dashboard Configuration Logs Health NEW Monitoring Alarms Events Tags

Elastic Beanstalk is updating your environment.
To cancel this operation select Abort Current Operation from the Actions dropdown.
[View Events](#)

Overview Refresh

Health OK Causes Running Version v1 Upload and Deploy

node Configuration
64bit Amazon Linux 2015.09
v2.0.8 running Node.js Change

While you are waiting, think about how a rolling deploy works. If you watched the associated course for Advanced Deployment Techniques on AWS, you should know that a Mutable Rolling Deploy will update the environment by briefly shutting down portions of it during deploy. If you go back to the "todo" front-end, you may be able to refresh during this Rolling Deploy and find that the application is unavailable.



While the Rolling Deploy in Elastic Beanstalk is very simple and the most convenient one, it may not fit your use case.

After a while, your environment should stabilize. The gray spinner on the left of center will stop spinning and turn back into a green check mark.

Elastic Beanstalk

Create New Environment

Really Cool App > reallyCoolApp-env (Environment ID: e-ptnvfxpys, URL: mightHaveToChange.us-west-2.elasticbeanstalk.com)

Actions ▾

Dashboard Overview Refresh

Configuration

Logs

Health NEW

Monitoring

Alarms

Events

Tags

Recent Events Show All

Health: Ok

Running Version: v2

Upload and Deploy

node.js

Configuration: 64bit Amazon Linux 2015.09
v2.0.8 running Node.js

Change

Your environment should report that it is running V2 now. The only change we made in V2 was altering some text in the todo view from **todos** to **Cool V2**. Are you able to see the updated test when you re-visit the application environment URL?

▼ What needs to be done?

- foo
- bar
- foo
- gree
- blrag

5 items left All Active Completed

Once you confirm that you can see the changes, proceed to the **next step** in this lab.

Step 3. Prepare a Blue-Green Deploy

Now that we have run a Rolling Deploy, it's time to try our hand at the other major kind of deploy, the Blue-Green deploy.

If you recall, the first thing we need with this kind of deploy is a complete second environment that mirrors the one currently being used.

Fortunately, Elastic Beanstalk has a **Clone Environment** feature. From within the environment overview area, click on the **Actions** dropdown in the top right, then click on **Clone Environment**.

The screenshot shows the AWS Elastic Beanstalk console for the 'Really Cool App' environment. The 'Actions' menu is open, with 'Clone Environment' selected. Other options in the menu include Load Configuration, Save Configuration, Swap Environment URLs, Clone with Latest Platform, Abort Current Operation, Restart App Server(s), Rebuild Environment, and Terminate Environment. Below the menu, system information is displayed: 64bit Amazon Linux 2015.09 and v2.0.8 running Node.js. A 'Change' button is also present.

You will be presented with a form and a view that describes the existing environment at a high level, then also shows the settings of the to-be-created environment. We are going to use this as the new environment to try blue-green on, so pick a name and Environment URL which is self-explanatory.

Original Environment

Environment name: reallyCoolApp-env
 Environment URL: mightHaveToChange.us-west-2.elasticbeanstalk.com
 Description: This is the first environment for our app.
 Platform: 64bit Amazon Linux 2015.09 v2.0.8 running Node.js

New Environment

Environment name: Environment URL:
 Description:
 Platform:
 Service role:

Once you click the blue **Clone** button, the same process and view that we experienced earlier during environment creation appears. Wait it out again.

Once you have the second environment open, we need to prepare that environment to be the upgraded version, or version number 3. Open the code deployment dialog again, use [this version three code](#), then execute a Rolling Deployment on this environment - because it is unused right now, we can tolerate the availability and performance concerns surrounding Rolling Deployments.

Upload and Deploy

To deploy a previous version, go to the [Application Versions page](#).

Upload application: v3.zip

Version label:

Deployment Preferences

Elastic Beanstalk will deploy to **30%** of instances in your auto scaling group at a time. Current number of instances: **0**

Batch size: Percentage
 % of instances at a time

Fixed
 instances at a time (max: 4)

Ignore health check:

Cancel **Deploy**

Created CloudWatch alarm named: awseb-e-a3p9arwmwm-sta

Wait again while this deploy completes.

Screenshot of the Elastic Beanstalk console showing the deployment process for environment "reallyCoolApp-round-2".

The "Health" status is currently "Ok". The "Running Version" is v2. A "Upload and Deploy" button is available.

The configuration details show:

- Configuration: 64bit Amazon Linux 2015.09, v2.0.8 running Node.js
- Health: Ok
- Running Version: v2
- Recent Events: None (Show All)

Screenshot of the Elastic Beanstalk console showing the deployment process for environment "reallyCoolApp-round-2".

The "Health" status is currently "Ok". The "Running Version" is v2. A "Upload and Deploy" button is available.

The configuration details show:

- Configuration: 64bit Amazon Linux 2015.09, v2.0.8 running Node.js
- Health: Ok
- Running Version: v2
- Recent Events: None (Show All)

Screenshot of the Elastic Beanstalk console showing the deployment process for environment "reallyCoolApp-round-2".

The "Health" status is currently "Ok". The "Running Version" is v3. A "Upload and Deploy" button is available.

The configuration details show:

- Configuration: 64bit Amazon Linux 2015.09, v2.0.8 running Node.js
- Health: Ok
- Running Version: v3
- Recent Events: None (Show All)

Now that the application's third version is deployed, manually verify that the code was deployed correctly to this environment by navigating to the URL associated to the environment and view the functioning Todo application. Note that this time, we changed the header text again, to **B-G Deploy**.

B-G Deploy

▼ What needs to be done?

- foo
- bar
- foo
- gree
- blrag

5 items left

All

Active

Completed

Once you're satisfied that a third version of the same application has been deployed, proceed to the **next step** in this Lab.

Step 4. Do the Deploy Swap

When you are ready to run the rest of the Blue-Green deploy, navigate back to the page with the environment dashboard.

Blue-Green deployments rely on having entire systems or subsystems deployed all at once using a DNS or reverse proxy cutover. We currently have two environments, one of which is running V3 and the other of which is running V2. The one running V2 is using the original DNS that we need to continue using, so we should quickly and invisibly trade DNS on the V2 and V3 versions, so V3 can get the original and more valuable URL.

Click on the **Actions** dropdown, then click on **Swap Environment URLs**.

Elastic Beanstalk Create New Environment

Really Cool App > reallyCoolApp-round-2 (Environment ID: e-q3p9qrwmwm, URL: reallycoolapp-round-2.us-west-2.elasticbeanstalk.com)

Actions ▾

- Load Configuration
- Save Configuration
- Swap Environment URLs**
- Clone Environment
- Clone with Latest Platform
- Abort Current Operation
- Restart App Server(s)
- Rebuild Environment
- Terminate Environment

64bit Amazon Linux 2015.09
v2.0.8 running Node.js

Dashboard Overview

Logs Health **NEW** Monitoring Alarms Events Tags

Recent Events Show All

Health: **OK** Causes

Running Version: v3 Upload and Deploy

Change

You will be taken to a page which allows you to select the environment you would like to swap with. Pick the V2 environment.

Elastic Beanstalk Create New Environment

Swap Environment URLs

When you swap an environment's URL with another environment's URL, you can deploy versions with no downtime. [Learn more.](#)

⚠ Swapping the environment URL will modify the Route 53 DNS configuration, which may take a few minutes. Your application will continue to run while the changes are propagated.

Environment Details

Environment name: reallyCoolApp-round-2

Environment URL: reallycoolapp-round-2.us-west-2.elasticbeanstalk.com

Select an Environment to Swap

Environment name:

Environment URL: mightHaveToChange.us-west-2.elasticbeanstalk.com

Cancel **Swap**

While the system warns you that the change could take a number of minutes, it's effectively instant in nearly all cases. The environment will likely be finished swapping as soon as you return to the dashboard for either environment.

Really Cool App > reallyCoolApp-round-2 (Environment ID: e-q3p9qrwmwm, URL: mighthavetochange.us-west-2.elasticbeanstalk.com)

Actions ▾

Dashboard Overview Refresh

Health OK Causes

Running Version v3 Upload and Deploy

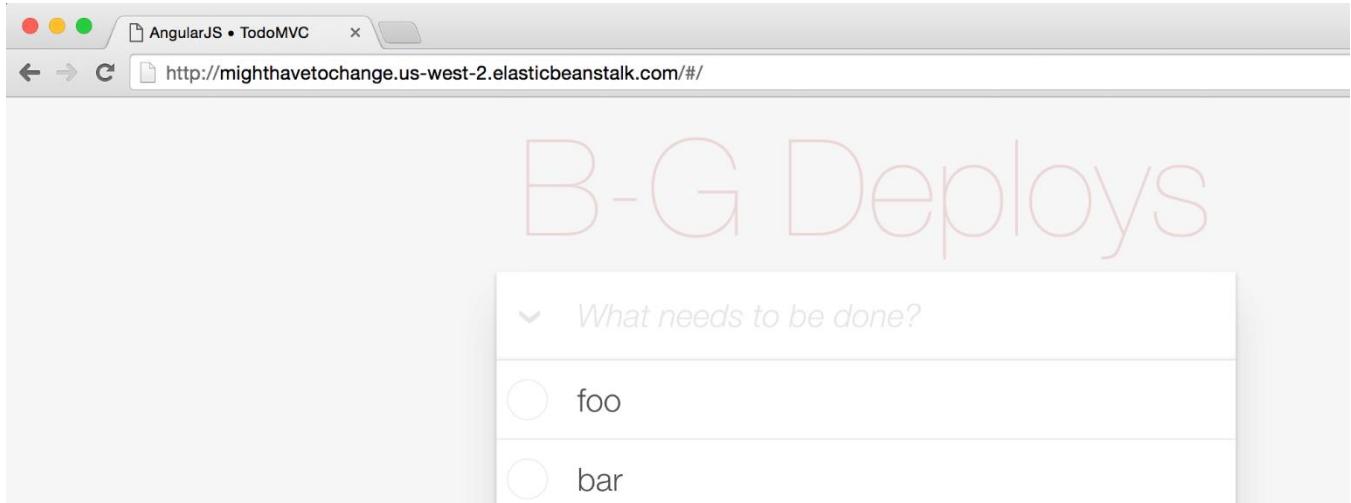
node.js

Configuration
64bit Amazon Linux 2015.09
v2.0.8 running Node.js Change

Logs Health NEW Monitoring Alarms Events Tags

This swap is simple to perform and fast. There is no downtime during the cutover since the DNS had both environments working correctly. All traffic to the original Elastic Beanstalk Environment is now routed to the new one running V3, as the V3 environment has taken over all the traffic via DNS.

Try it out - visit the original URL you made for the **first** environment, and you should see the V3 version running there now.



Once you are satisfied that the deployment worked as intended, proceed to the **next step**.

Step 5. Clean Up Old Resources

Now that we have successfully tried two kinds of low-downtime cloud deployment techniques on Elastic Beanstalk, we might be concerned about the costs associated with these techniques. The deploys are inexpensive, but they may get a little more expensive if we simply leave the environment we swapped away from online all the time. We should clean up the environment to save costs.

First, navigate back to the dash for the environment we just swapped to - the one running V3.

The screenshot shows the Elastic Beanstalk application dashboard for the environment 'reallyCoolApp-round-2'. At the top, there's a navigation bar with the Elastic Beanstalk logo and a 'Create New Environment' button. Below the navigation, the application name 'Really Cool App' and environment ID 'e-q3p9qrwmwm' are displayed, along with the URL 'mightHaveToChange.us-west-2.elasticbeanstalk.com'. On the right, there's an 'Actions' dropdown and a 'Refresh' button. The main area is divided into sections: 'Dashboard' (selected), 'Overview' (disabled), 'Logs', 'Health NEW' (disabled), 'Monitoring', 'Alarms', 'Events', and 'Tags'. In the center, there's a large green circle with a checkmark indicating 'Health Ok'. Below it is a 'Causes' button. To the right, the 'Running Version' is listed as 'v3' with a 'Upload and Deploy' button. Further right is a 'node.js' icon and a 'Configuration' section showing '64bit Amazon Linux 2015.09 v2.0.8 running Node.js' with a 'Change' button.

Click on the Elastic Beanstalk icon in the top-left of this view to navigate to the **All Applications** view. Your application will show up, along with two green boxes, representing two live environments associated with the application.

All Applications

Filter by Application

Really Cool App

<u>reallyCoolApp-env</u>	<u>reallyCoolApp-round-2</u>
Environment tier: Web Server	Environment tier: Web Server
Running versions: v2	Running versions: v3

Click the one running **v2**. You will be taken to the dashboard for this now-obsolete environment. Click on the **Actions** drop-down, then click on the **Terminate Environment** option, so we can destroy this environment and save some money.

Elastic Beanstalk

Create New Environment

Really Cool App > reallyCoolApp-env (Environment ID: e-ptnvfpvys, URL: reallycoolapp-round-2.us-west-2.elasticbeanstalk.com)

Actions ▾

- Load Configuration
- Save Configuration
- Swap Environment URLs
- Clone Environment
- Clone with Latest Platform
- Abort Current Operation
- Restart App Server(s)
- Rebuild Environment
- Terminate Environment

64bit Amazon Linux 2015.09
v2.0.8 running Node.js

Dashboard Overview

Health **OK**

Running Version v2

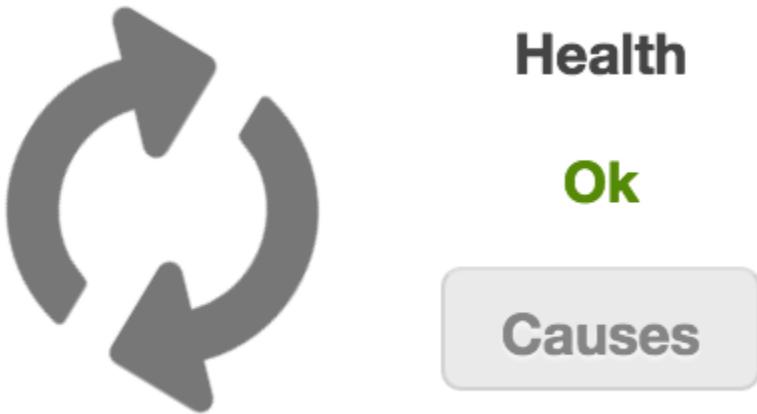
Upload and Deploy

Causes

Recent Events Show All

Change

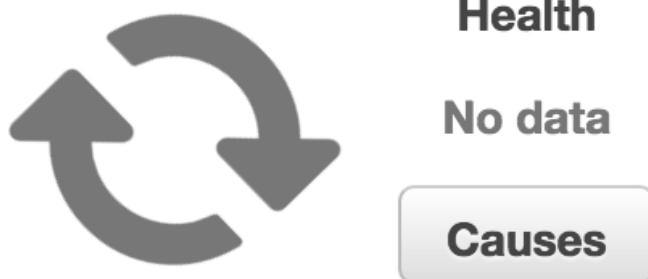
You should see the environment spin into a gray state, then see the **Health** spin into a death spiral - it might swing between **OK**, **Severe**, and **No Data**. Wild!



Elastic Beanstalk Create New Environment

Really Cool App > reallyCoolApp-round-2 (Environment ID: e-q3p9qrwmwm, URL: mightHaveToChange.us-west-2.elasticbeanstalk.com) Actions ▾

Dashboard Configuration Logs Health NEW Monitoring Alarms Events Tags	Overview <div style="display: flex; justify-content: space-between;"> <div style="text-align: center;"> Health Ok Causes </div> <div style="text-align: center;"> Running Version v3 Upload and Deploy </div> <div style="text-align: center;"> Configuration 64bit Amazon Linux 2015.09 v2.0.8 running Node.js Change </div> </div>
--	---



Eventually, this will end, and you will be sent back to the **All Applications** view. This time, there will not be two green environments under your application - there will be one light-gray one, and one green one. The light-gray one is the terminated environment.

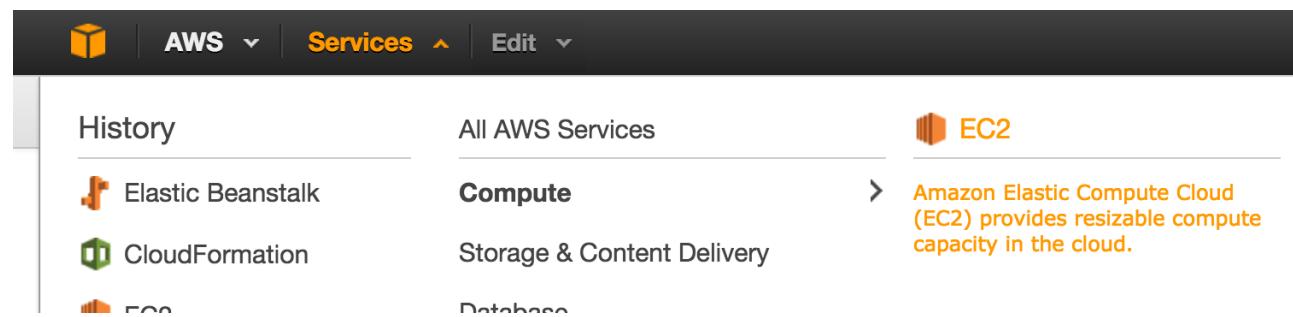
All Applications

Filter by Application

Really Cool App

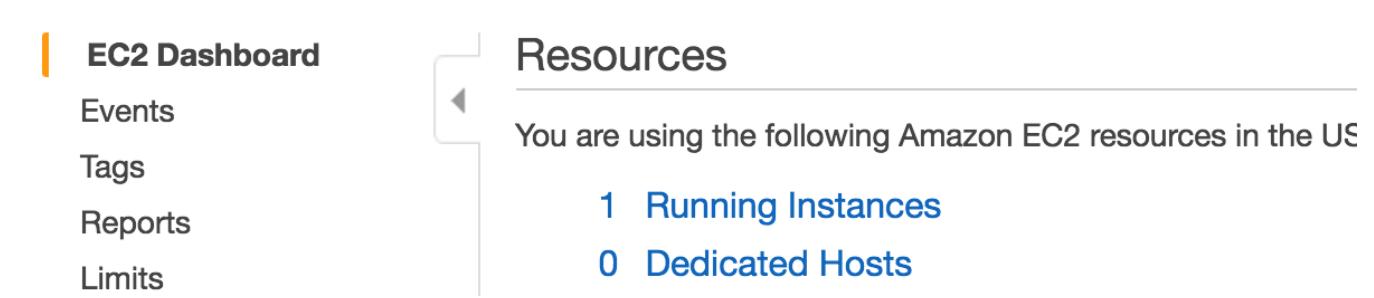
reallyCoolApp-env (Terminated)	reallyCoolApp-round-2
Environment tier: Web Server Running versions: v2	Environment tier: Web Server Running versions: v3

To verify that we reduced the spend of our system, we can enter the EC2 console and make sure that the EC2 instances that Elastic Beanstalk creates on your behalf are terminated. Navigate to the EC2 Console by clicking on the **Services** dropdown in the top-left of the Console, hovering over the **Compute** section, and clicking on the **EC2** section.



The screenshot shows the AWS Services navigation bar. The 'Compute' section is selected, and the 'EC2' service is chosen under it. Other services like Elastic Beanstalk, CloudFormation, and RDS are also listed.

After navigating to the top level of the EC2 console, you should be able to see that there is only 1 Running Instance now if you allowed your old/second Elastic Beanstalk Environment to totally terminate.



The screenshot shows the EC2 Dashboard. In the Resources section, it indicates 1 Running Instances and 0 Dedicated Hosts. A note says: "You are using the following Amazon EC2 resources in the US."

For further verification that actual instances were terminated, click the **Running Instances** link and review the table of instances. One should be **Running**, and one or more should display as red and **Terminated**. This is / these are instance(s) which belonged to the now-terminated Elastic Beanstalk Environment.

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with links like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Instances (which is selected and highlighted in orange), and Spot Requests. The main area has tabs for Launch Instance, Connect, and Actions. Below that is a search bar with placeholder text "Filter by tags and attributes or search by keyword". A table lists two instances:

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
<input type="checkbox"/>	reallyCoolAp...	i-5e4f6799	t1.micro	us-west-2a	running	2/2 checks ...	None	ec2-52-37-198-89.u
<input type="checkbox"/>	reallyCoolAp...	i-7d7c54ba	t1.micro	us-west-2a	terminated		None	

You've learned how to run two types of low-interruption, high-automation deploys on AWS Elastic Beanstalk.