



lab



lab title

**AWS X-Ray  
V1.02**



Course title

**BackSpace Academy  
AWS Certification Preparation**



# Table of Contents

## Contents

Table of Contents.....	1
About the Lab .....	2
Integrating a NodeJS Application with AWS X-Ray .....	3
Download the Application .....	3
Integrating with AWS X-Ray.....	4
Creating an Elastic Beanstalk X-Ray Application .....	7
Create the Elastic Beanstalk Environment .....	7
Troubleshooting.....	11
Analyzing Application Performance with the AWS X-Ray Console .....	12
Capturing Calls to the AWS SDK with X-Ray .....	15
Create a Role to Access S3.....	17
Associate IAM Role to Elastic Beanstalk Instances.....	18
Upload new Application Version to Elastic Beanstalk .....	20
Analyse with X-Ray .....	21

## About the Lab

**Please note that not all AWS services are supported in all regions. Please use the US-East-1 (North Virginia) region for this lab.**

These lab notes are to support the hands on instructional videos of the AWS X-Ray section of the AWS Certified Developer Associate course.

**Please note that AWS services change on a weekly basis and it is extremely important you check the version number on this document to ensure you have the latest version with any updates or corrections.**

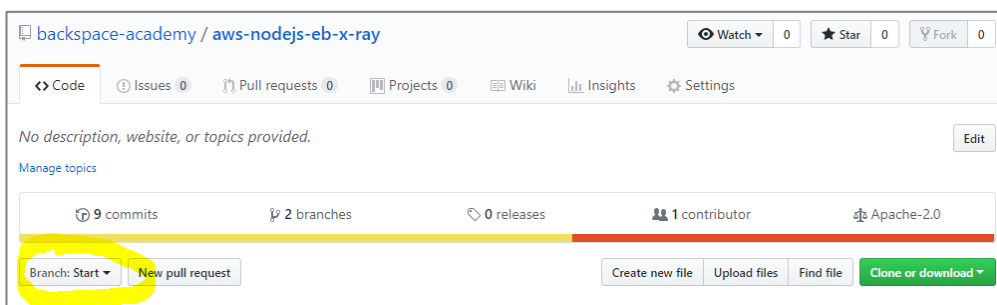
# 🎬 Integrating a NodeJS Application with AWS X-Ray

In this section, we will use the AWS X-Ray SDK for NodeJS to enable a NodeJS application to send segments to the X-Ray daemon.

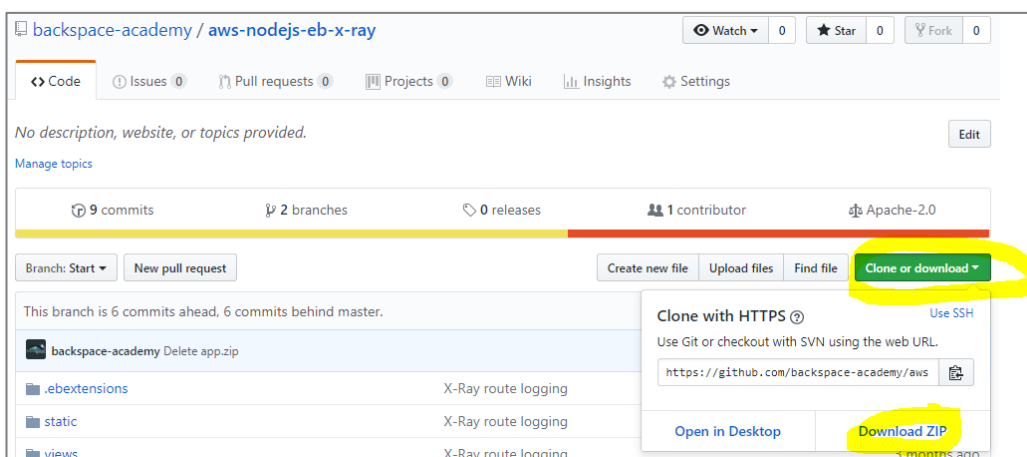
Download the Application

Go to <https://github.com/backspace-academy/aws-nodejs-eb-x-ray/tree/Start>

Make sure the branch is on *Start*

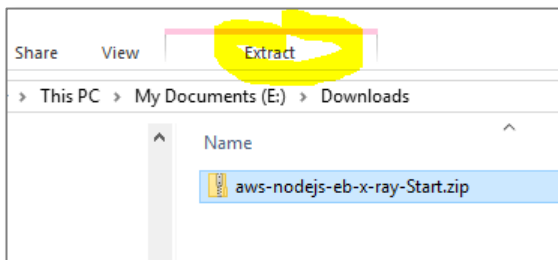


Download the ZIP file containing the application



The code from GitHub will be hidden inside a folder. We need to make sure the code is at the root level of the ZIP file.

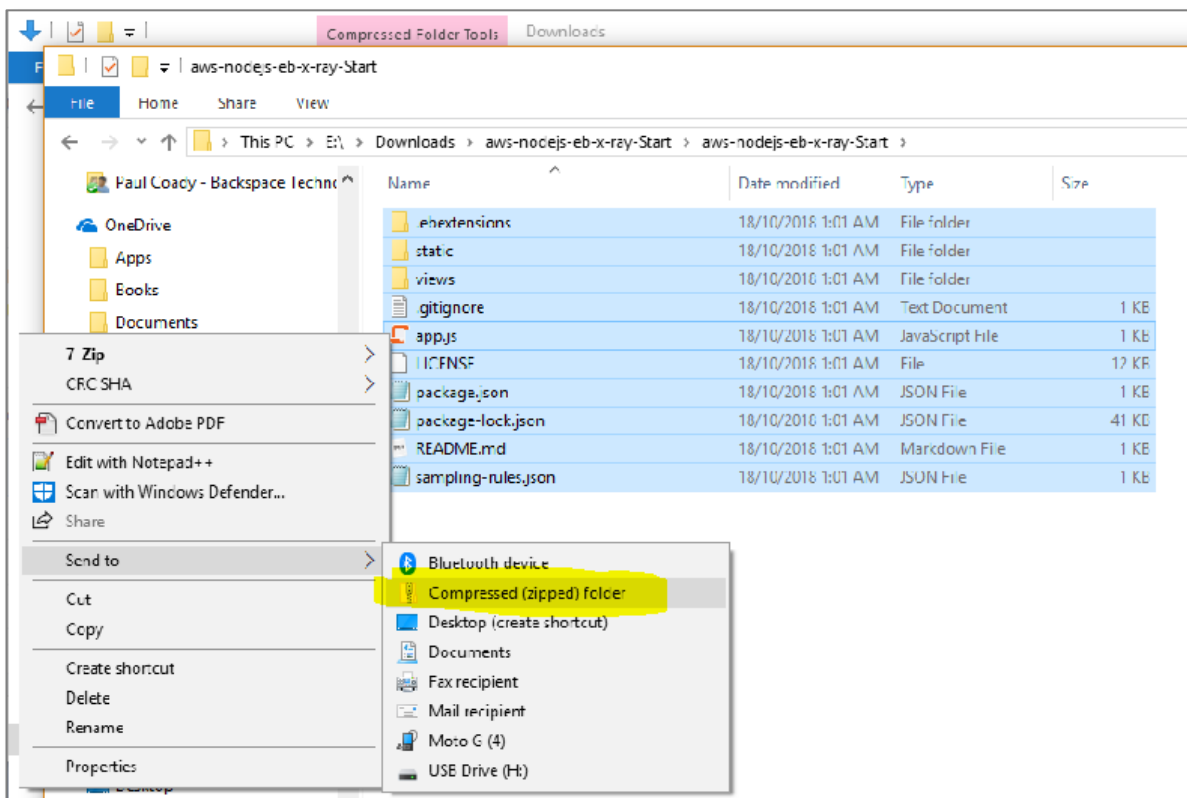
Extract the ZIP file.



Open the folder

Select the code

Create another ZIP file



## Integrating with AWS X-Ray

In the `.ebextensions` folder there will be a config file to set up the configuration of Elastic Beanstalk for X-Ray:

- Setting option `XRayEnabled` to true
- Defining the location for X-Ray logs to be stored.

```
option_settings:
  aws:elasticbeanstalk:xray:
    XRayEnabled: true

files:
  "/opt/elasticbeanstalk/tasks/taillogs.d/xray-daemon.conf" :
    mode: "000644"
    owner: root
    group: root
    content: |
      /var/log/xray/xray.log
```

The *sampling-rules.json* file contains the sampling rule (capture everything):

- Target of one sample per second
- 100% of all samples over the target

```
{
  "version": 1,
  "default": {
    "fixed_target": 1,
    "rate": 1.0
  }
}
```

The application code in *app.js* to integrate with X-Ray has the following:

- to load the AWS X-Ray SDK
  - `require('aws-xray-sdk');`
- to load the AWS SDK and capture calls to it
  - `XRay.captureAWS(require('aws-sdk'));`
- to load the http client and capture http traffic
  - `XRay.captureHTTPs(require('http'));`
- to set the default sampling rules
  - `XRay.middleware.setSamplingRules('sampling-rules.json');`
- to start a segment called myfrontend
  - `app.use(XRay.express.openSegment('myfrontend'));`
- to start a subsegment called 'Page Render'
  - `XRay.captureAsyncFunc('Page Render', function(seg) {`
- To close subsegment called 'Page Render'
  - `seg.close();`
- To close segment called myfrontend
  - `app.use(XRay.express.closeSegment());`

```

// Include the AWS X-Ray SDK
var XRay = require('aws-xray-sdk');
// Capture calls to the AWS SDK
var AWS = XRay.captureAWS(require('aws-sdk'));
// Capture http traffic
var http = XRay.captureHTTPs(require('http'));

var express = require('express');
var bodyParser = require('body-parser');

// Set region for AWS SDKs
AWS.config.region = process.env.REGION

// Configure sampling rules
XRay.middleware.setSamplingRules('sampling-rules.json');

var app = express();

app.set('view engine', 'pug');
app.set('views', __dirname + '/views');
app.use(bodyParser.urlencoded({ extended: false }));

//Start X-ray segment myfrontend
app.use(XRay.express.openSegment('myfrontend'));

app.get('/', function (req, res) {
  // Start X-ray subsegment 'Page Render'
  XRay.captureAsyncFunc('Page Render', function (seg) {
    res.render('index', {
      title: 'BackSpace Academy and AWS X-Ray'
    });
    seg.close(); // Close X-ray subsegment 'Page Render'
  });
  res.status(200).end();
});

//Close X-ray segment myfrontend
app.use(XRay.express.closeSegment());

var port = process.env.PORT || 3000;

var server = app.listen(port, function () {
  console.log('Server running at http://127.0.0.1:' + port + '/');
});

```

# **Creating** an Elastic Beanstalk X-Ray Application

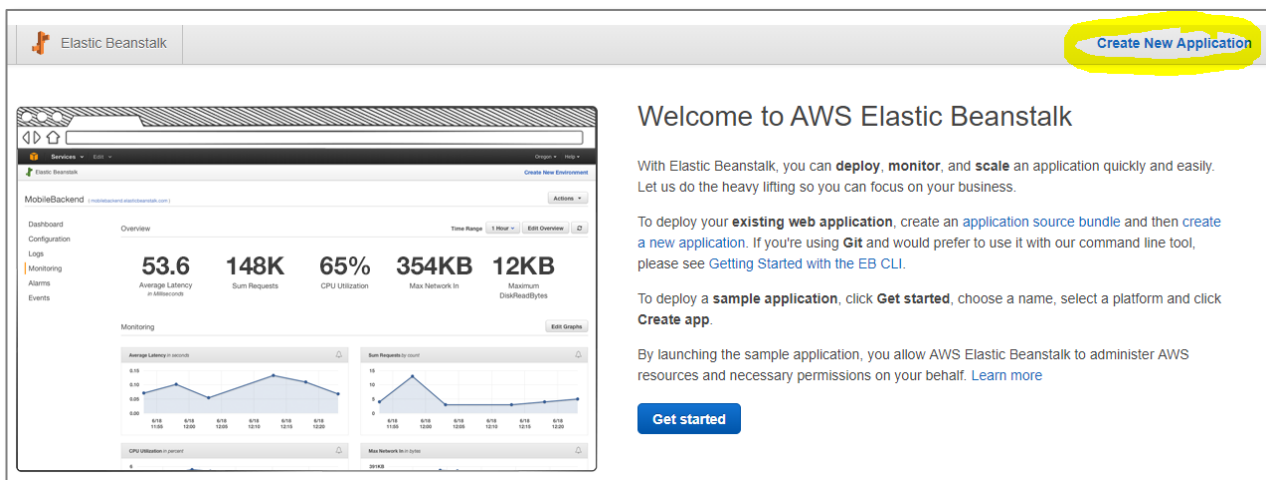
In this section, we will use the Elastic Beanstalk console to launch an Elastic Beanstalk environment that will run both our application and the X-ray daemon.

Create the Elastic Beanstalk Environment

From the AWS console click *Services*

Select *Elastic Beanstalk*

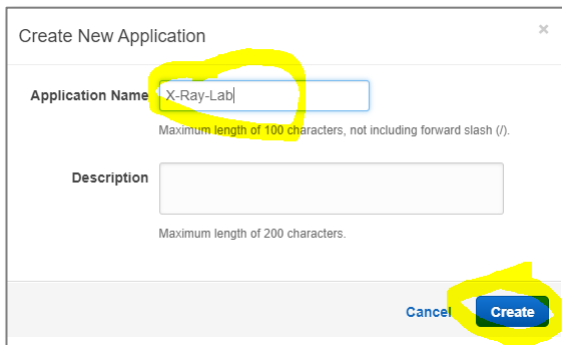
Click *Create New Application*



Give your application a name

Click *Create*





Create New Application

Application Name

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

Description

Maximum length of 200 characters.

Cancel Create

Select *Actions*

Click *Create environment*



All Applications > X-Ray-Lab

Environments

Application versions

Saved configurations

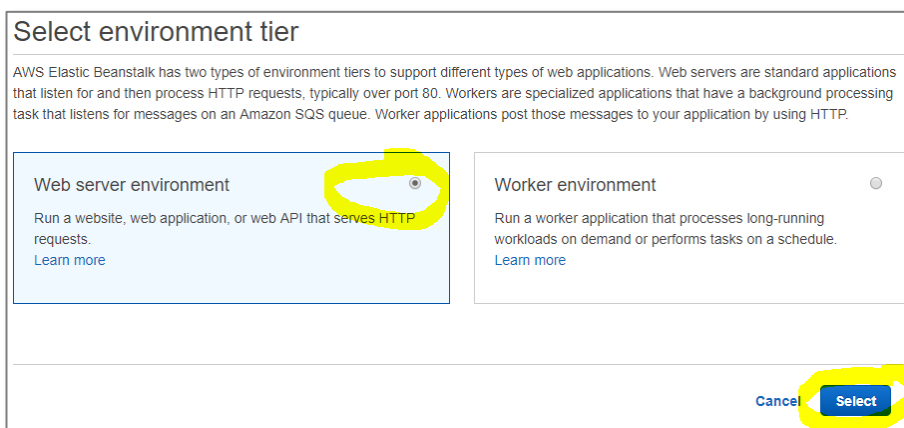
No environments currently exist for this application. [Create one now.](#)

Actions

- Create environment
- Restore terminated environment
- Swap environment URLs
- Delete application

Select *Web server environment*

Click *Select*



Select environment tier

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 ☒

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](#)

Cancel Select

Select *Preconfigured platform*

Select *Node.js*

Base configuration

**Platform**

☒ Preconfigured platform  
Platforms published and maintained by AWS Elastic Beanstalk.

☐ Custom platform  
Platforms created and owned by you. [Learn more](#)

Node.js

-- Choose a custom platform --

Select *Upload your code*

Click *Upload*

**Application code**

☐ Sample application  
Get started right away with sample code.

☐ Existing version  
Application versions that you have uploaded for backspace-lab.

-- Choose a version --

☒ Upload your code  
Upload a source bundle from your computer or copy one from Amazon S3.

Upload

Upload the zip file you created previously

Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

**Source code origin**

☒ Local file  
(Maximum size 512 MB)

Choose File | app.zip

☐ Public S3 URL

https://s3.amazonaws.com

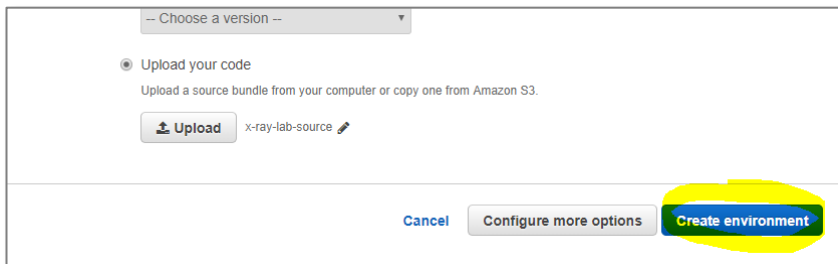
**Version label**

x-ray-lab-source

Unique name for this version of your application code.

Cancel Upload

Click *Create environment*



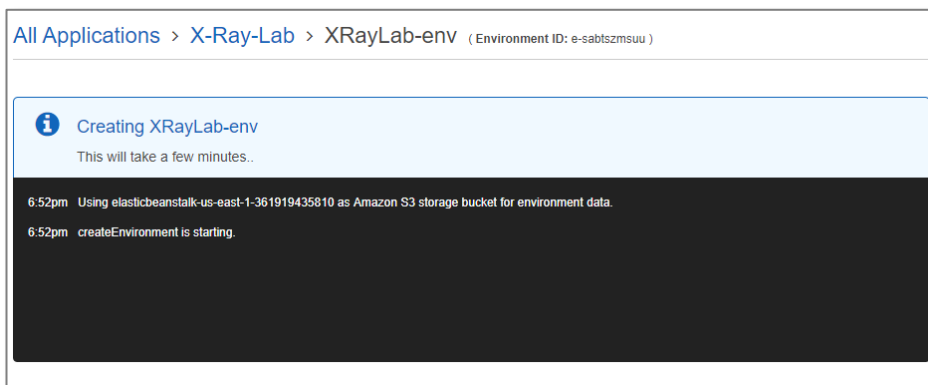
-- Choose a version --

☒ Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

x-ray-lab-source

Elastic Beanstalk will now start creating your environment.



All Applications > X-Ray-Lab > XRayLab-env (Environment ID: e-sabtszmsuu)

**Creating XRayLab-env**

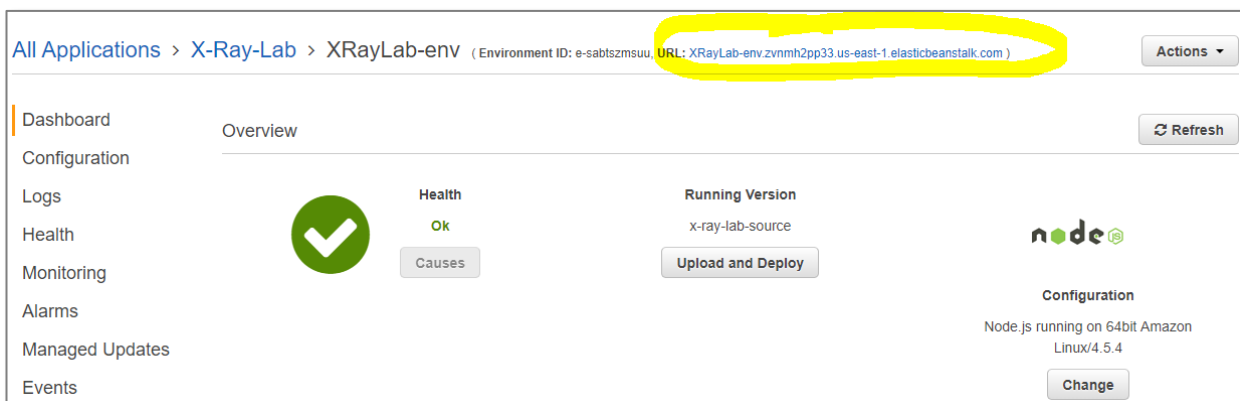
This will take a few minutes..

6:52pm Using elasticbeanstalk-us-east-1-361919435810 as Amazon S3 storage bucket for environment data.

6:52pm createEnvironment is starting.

After about 15 minutes your environment will be created (if any problems see troubleshooting below)

Click on the endpoint for your application




All Applications > X-Ray-Lab > XRayLab-env (Environment ID: e-sabtszmsuu, URL: XRayLab-env.zvmh2pp33.us-east-1.elasticbeanstalk.com)

Actions

Dashboard Overview Refresh

Configuration

Logs

Health  Health **Ok** Causes

Monitoring

Alarms

Managed Updates

Events

Running Version

x-ray-lab-source

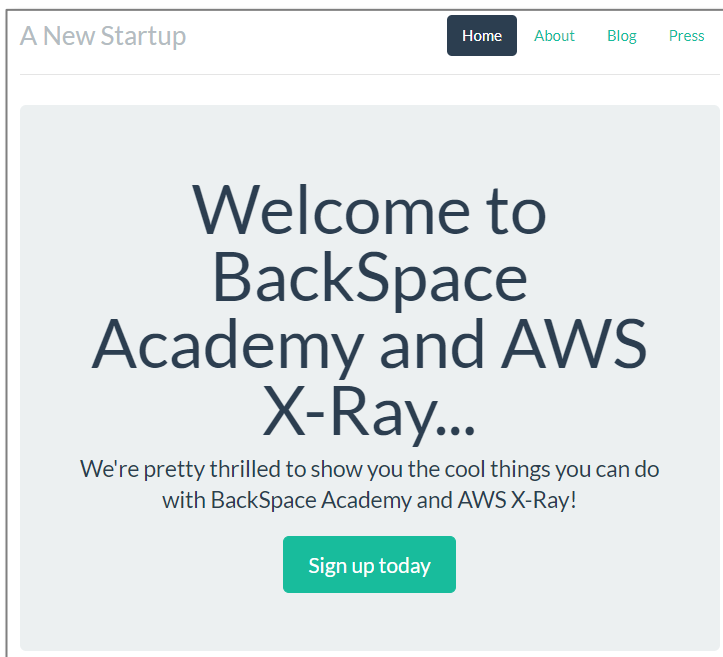
Upload and Deploy

Configuration

Node.js running on 64bit Amazon Linux/4.5.4

Change

You should now see the running web application



## Troubleshooting

502 Bad Gateway
nginx/1.12.1

This is most probably caused by Elastic Beanstalk being unable to find your application in the ZIP file.

Make sure the application is not inside a folder before uploading the zip file. If you download the ZIP file directly from GitHub it will put the application inside a folder inside the Zip file and Elastic Beanstalk won't be able to find it.

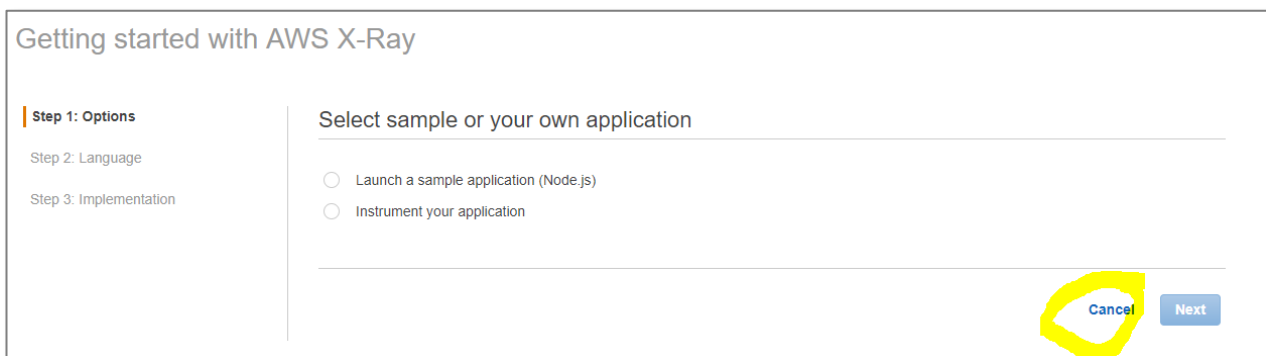
# ▶ **Analyzing** Application Performance with the AWS X-Ray Console

In this section, we will use the AWS X-Ray console to view the service diagram and traces.

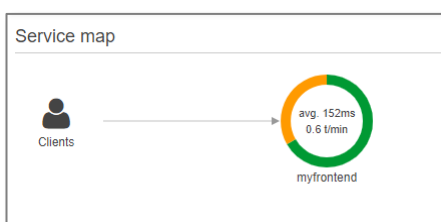
From the AWS console click *Services*

Select *X-Ray*

If you see a Getting Started screen click *Cancel*

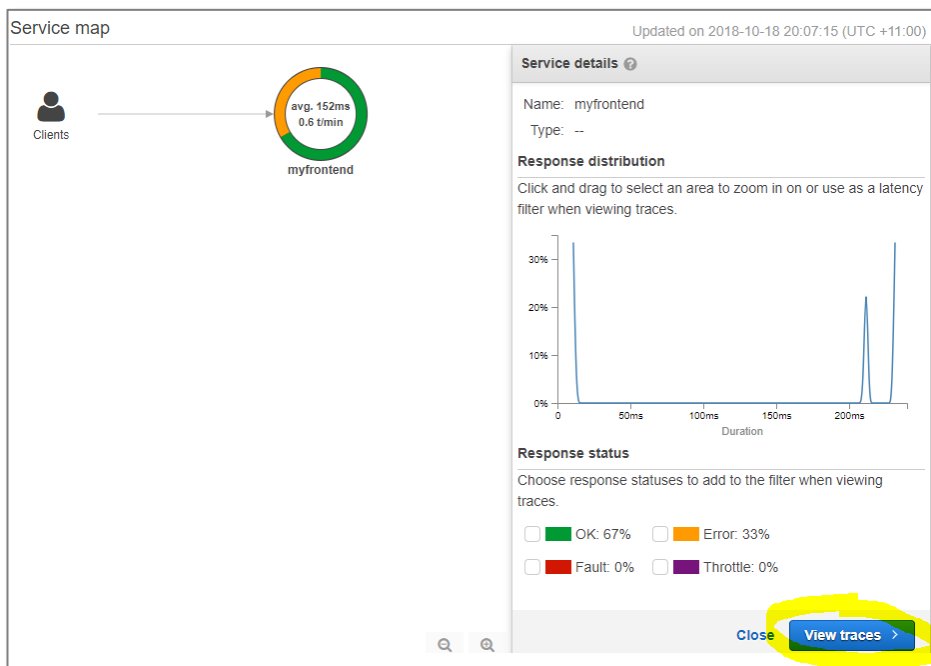


After about a minute the Service map will appear



Click on *myFrontEnd* node

Click on *View traces*



You will now be able to see more performance information about the requests to the table

Click on the trace with a 404 error to see the timeline

Trace overview

Group by: URL Done 100% scanned (found 3 traces)

URL	Avg response time	% of Traces	Respor
http://xraylab-env.zvnmh2pp33.us-east-1.elasticbeanstalk.com/favi...	11.0 ms	33.33%	0 OK, 0
http://xraylab-env.zvnmh2pp33.us-east-1.elasticbeanstalk.com/	211 ms	33.33%	1 OK, 0
http://18.235.151.46/	233 ms	33.33%	1 OK, 0

Trace list

ID	Age	Method	Response	Response time	URL	Clie
adb30d96	7.9 min	GET	200	233 ms	http://18.235.15...	60.19
34ba55dc	3.6 min	GET	200	211 ms	http://xraylab-en...	103.7
8c6d1913	3.5 min	GET	404	11.0 ms	http://xraylab-en...	103.7

Click on the trace to get more information

Traces > Details

Timeline Raw data

Method	Response	Duration	Age	ID
GET	404	11.0 ms	5.5 min (2018-10-18 09:06:23 UTC)	1-5bc84d0f-8b139f915484086f8c6d1913

Name	Res.	Duration	Status	0.0ms	1.0ms	2.0ms	3.0ms	4.0ms	5.0ms	6.0ms	7.0ms	8.0ms	9.0ms	10ms	11ms
myfrontend	404	11.0 ms	⚠												

Segment - myfrontend

Overview

Resources

Annotations

Metadata

Exceptions

Segment ID	a8b54509f5bdd173
Parent ID	
Name	myfrontend
<b>Time</b>	
Start time	2018-10-18 09:06:22.645 (UTC)
End time	2018-10-18 09:06:22.656 (UTC)
Duration	11.0 ms
In progress	False
<b>Errors &amp; Faults</b>	
Error	True
Fault	False
<b>Request &amp; Response</b>	
Request url	http://xraylab-env.zvnmh2pp33.us-east-1.elasticbeanstalk.com/favicon.ico
Request method	GET
Request user_agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36
Request client_ip	103.75.207.55
Request x_forwarded_for	
Response status	404

Close

# Capturing Calls to the AWS SDK with X-Ray

In this section, we will use the AWS X-Ray SDK to capture calls to Amazon S3.

Open App.js in a code editor

Add code to:

- initialise AWS S3
- call S3.listBuckets
- render page in the listBuckets callback

```
// Include the AWS X-Ray SDK
var XRay = require('aws-xray-sdk');
// Capture calls to the AWS SDK
var AWS = XRay.captureAWS(require('aws-sdk'));
// Capture http traffic
var http = XRay.captureHTTP(require('http'));
// Initialise S3
var S3 = new AWS.S3();

var express = require('express');
var bodyParser = require('body-parser');

// Set region for AWS SDKs
AWS.config.region = process.env.REGION

// Configure sampling rules
XRay.middleware.setSamplingRules('sampling-rules.json');

var app = express();

app.set('view engine', 'pug');
app.set('views', __dirname + '/views');
app.use(bodyParser.urlencoded({ extended: false }));

//Start X-ray segment myfrontend
app.use(XRay.express.openSegment('myfrontend'));

app.get('/', function (req, res) {
  XRay.captureAsyncFunc('Page Render', function (seg) {
    S3.listBuckets(function (err, data) {
      var bucketList = '';
      if (err) bucketList = JSON.stringify(err); // an error occurred
      else { // successful response
        for (var a = 0; a < data.Buckets.length; a++) {
          bucketList += JSON.stringify(data.Buckets[a].Name);
        }
      }
    });
  });
});
```



```

    }
    // Render page after listBuckets finished
    res.render('index', {
      title: 'BackSpace Academy and AWS X-Ray',
      bucketList: bucketList
    });
    seg.close();
  });
});

//Close X-ray segment myfrontend
app.use(XRay.express.closeSegment());

var port = process.env.PORT || 3000;

var server = app.listen(port, function () {
  console.log('Server running at http://127.0.0.1:' + port + '/');
});

```

Open *index.pug* in the views folder

Add a place for the bucketList variable to be displayed.

```

doctype html
html(lang="en")
  head
    meta(charset="utf-8")
    meta(name="viewport" content="width=device-width, initial-scale=1.0")
    meta(name="description" content="")
    meta(name="author" content="")
    title #{title} Example
    // Bootstrap core CSS
    link(href="static/bootstrap/css/theme/flatly/bootstrap.css" rel="stylesheet")
    // Custom styles for this template
    link(href="static/bootstrap/css/jumbotron-narrow.css" rel="stylesheet")
  body
    .container
      .header
        ul.nav.nav-pills.pull-right
          li.active
            a(href="#") Home
          li
            a(href="#") About
          li
            a(href="#") Blog
          li
            a(href="#") Press
        h3.text-muted A New Startup
      .jumbotron
        h1 Welcome to #{title}...
        p.lead
          | We're pretty thrilled to show you the cool things you can do with #{title}!
        h2 Bucket List
        p

```

```

    #{bucketList}
    p
      a.btn.btn-lg.btn-success(data-toggle="modal" href="#") Sign up today
    .footer
      p @ A New Startup 2016
      script(src="static/jquery/jquery-1.11.3.min.js")
      script(src="static/bootstrap/js/bootstrap.min.js")

```

Save the changed files and create a new zip file.

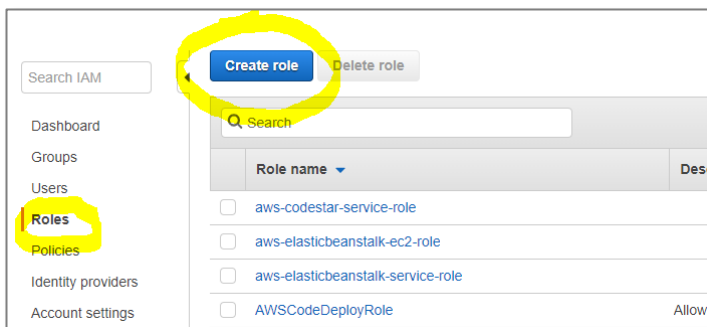
## Create a Role to Access S3

By default, Elastic Beanstalk uses *aws-elasticbeanstalk-ec2-role* for instances it launches. This role doesn't have read access for S3 so we need to create a new role.

Go to the IAM Management Console

Select Roles

Click *Create role*

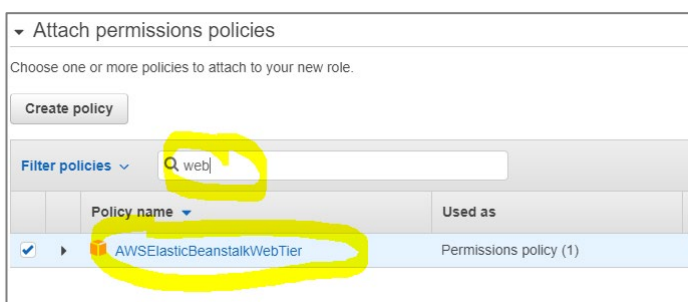


Select *AWS Service*

Select *EC2*

Click *Next: Permissions*

Select *AWSElasticBeanstalkWebTier* policy



## Select *AmazonS3ReadOnlyAccess*

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies

	Policy name	Used as
<input type="checkbox"/>	AmazonDMSRedshiftS3Role	None
<input type="checkbox"/>	AmazonS3FullAccess	None
<input checked="" type="checkbox"/>	AmazonS3ReadOnlyAccess	Permissions policy (1)
<input type="checkbox"/>	QuickSightAccessForS3StorageManagementA...	None

Click Next: Tags

Click Next: Review

Call the role *EB-EC2-S3-Read-Only*

Make sure both policies are attached and click *Create role*

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name\*

Use alphanumeric and '+', '@', '-' characters. Maximum 64 characters.

Role description

Maximum 1000 characters. Use alphanumeric and '+', '@', '-' characters.

Trusted entities AWS service:

Policies

- ☒ AWSElasticBeanstalkWebTier
- ☒ AmazonS3ReadOnlyAccess

Permissions boundary Permissions boundary is not set

No tags were added.

\* Required

Cancel Previous **Create role**

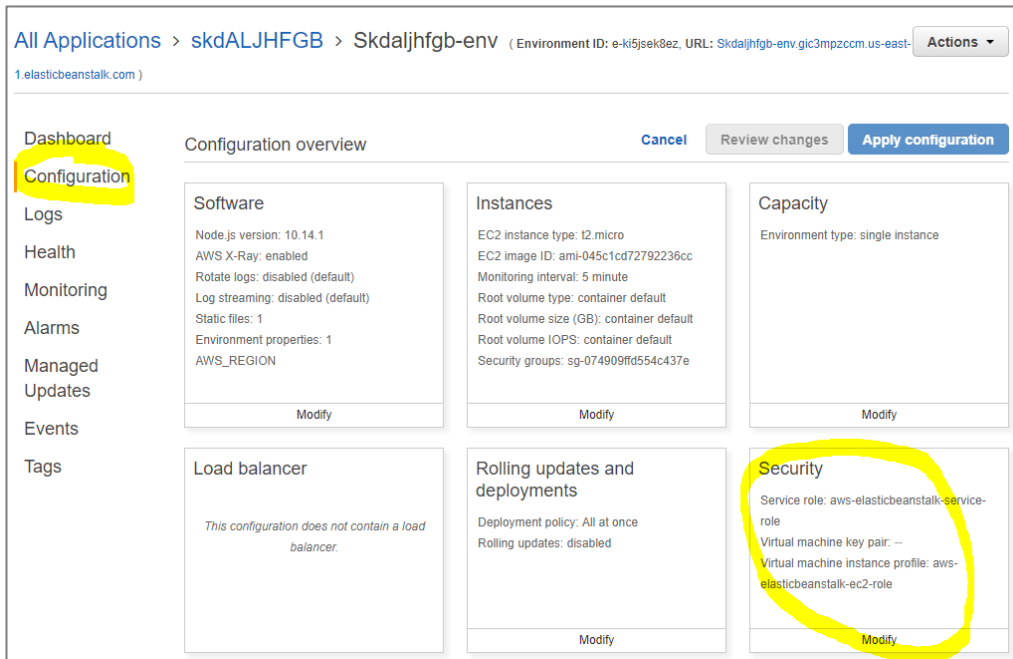
## Associate IAM Role to Elastic Beanstalk Instances

Go to the Elastic Beanstalk console

Go to your environment

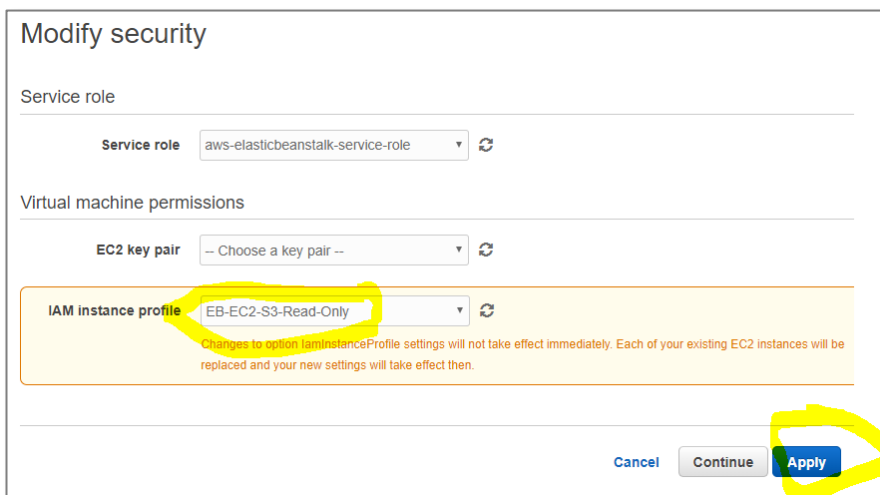
Select *Configuration*

Select *Security – Modify*



Select the IAM role you created previously for *IAM instance profile*.

Click *Apply*



Click *Confirm*

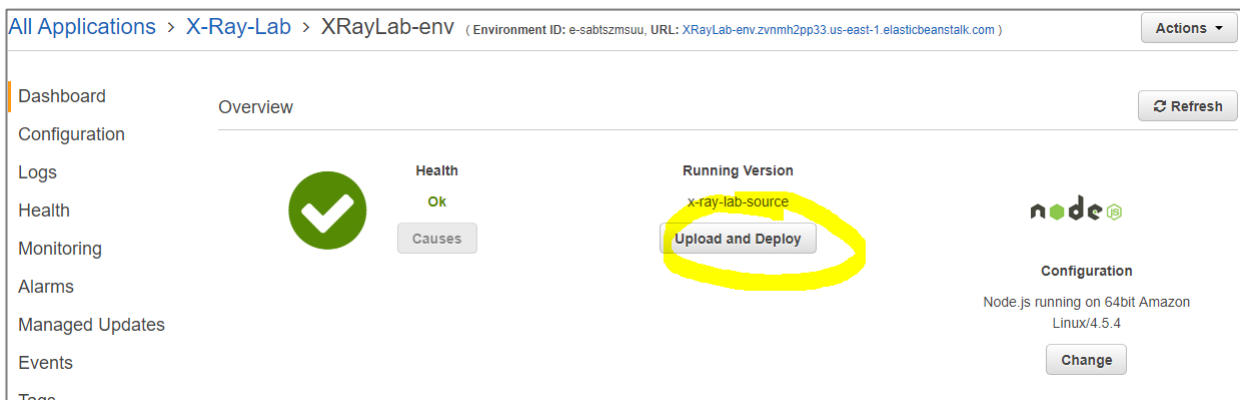


## Upload new Application Version to Elastic Beanstalk

Wait for the update to complete (this will take some time)

Go to the Dashboard

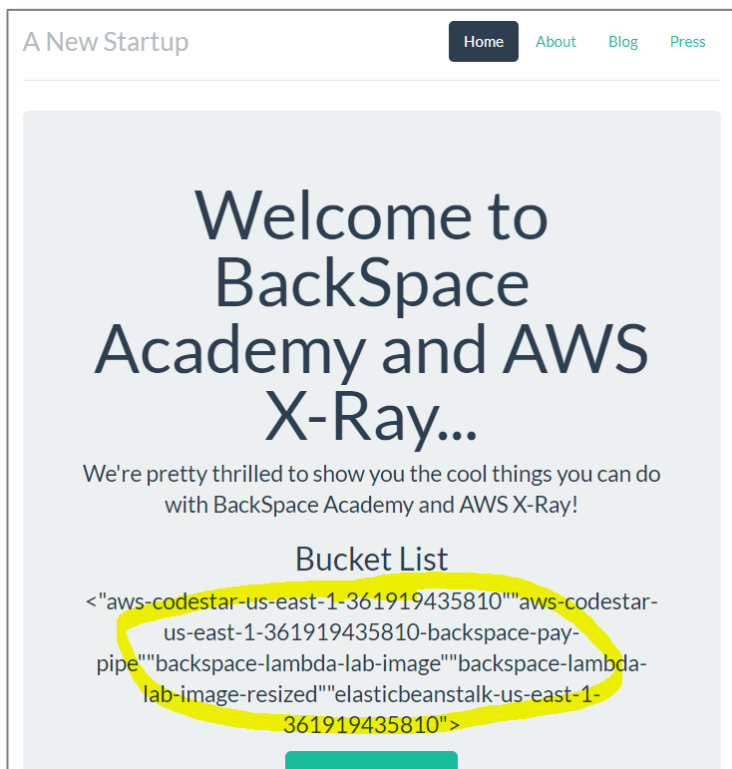
Click *Upload and Deploy*



Upload the new zip file

When the environment has been updated click on the endpoint again

You should see the list of buckets in your account.



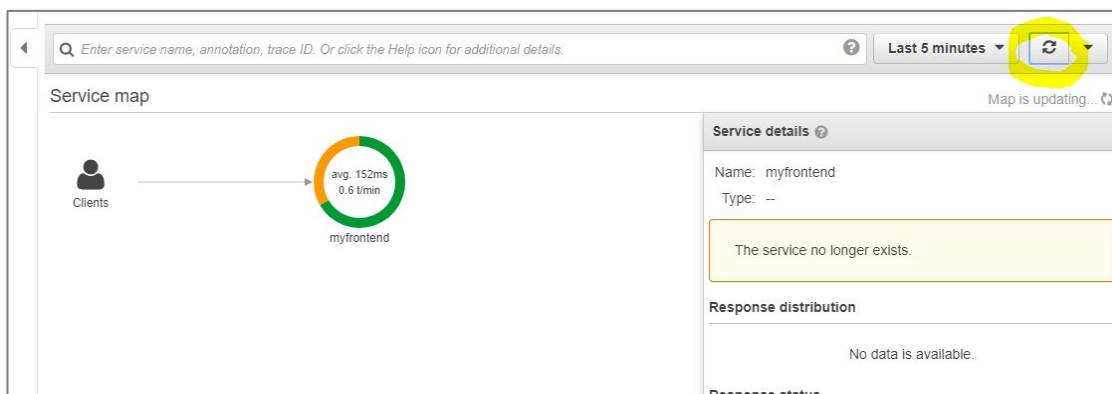
## Troubleshooting

If you don't see a list of buckets there is most probably an error with your code. You can download the completed master branch from:

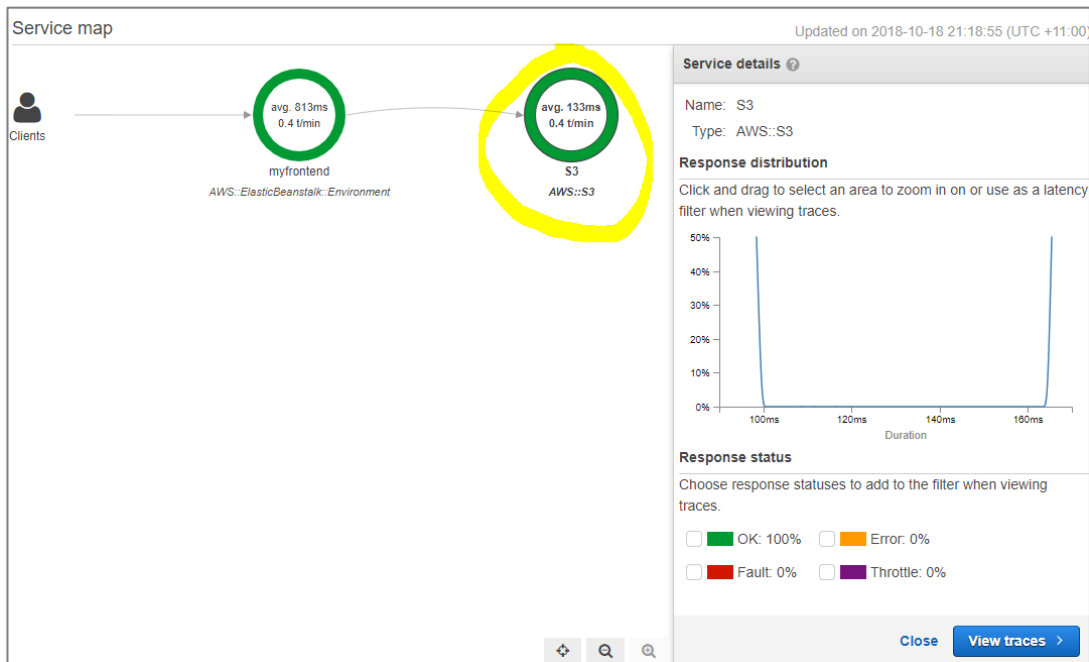
<https://github.com/backspace-academy/aws-nodejs-eb-x-ray/tree/master>

## Analyse with X-Ray

Go to the X-Ray console to and click the refresh button



You should see the new node for the call to S3 from the AWS SDK



## Cleaning Up

Deleting the Elastic Beanstalk application will delete the resources created.

From the AWS console click **Services**

Select **Elastic Beanstalk**

Select **Actions**

Click **Delete application**

All Applications

Filter by Application Name:

backspace-lab

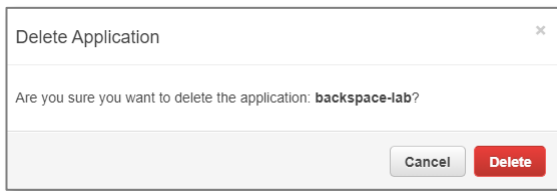
Actions

- Create environment
- Delete application**
- View application versions
- View saved configurations
- Restore terminated environment

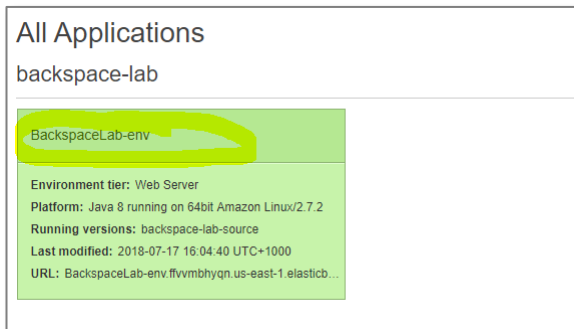
BackspaceLab-env

Environment tier: Web Server  
Platform: Java 8 running on 64bit Amazon Linux/2.7.2  
Running versions: backspace-lab-source  
Last modified: 2018-07-17 15:12:50 UTC+1000  
URL: BackspaceLab-env flvmbhyqn.us-east-1.elasticb...

Click **Delete**



Click on the environment to check it is being deleted



Your environment should now be terminating

