Lab Setting Up

Start Lab

15 Credits

info_outline
This lab costs 15 Credits to run. You can purchase credits or a subscription under My Account.

01:55:00

Building Scalable Web Applications with AWS Elastic Beanstalk

SPL-45 - Version 1.4.4

Errors or corrections? Email us at aws-course-feedback@amazon.com.

Other questions? Contact us at https://aws.amazon.com/contact-us/aws-training/

Overview

This lab leads you through the common steps of developing a web application and deploying it to production on AWS. At the start of this lab, you will deploy a functioning web application to AWS Elastic Beanstalk and learn how to deploy applications from version control using command line tools. You will expose a scalability problem with the application, and iterate over the application so that it can seamlessly scale, by externalizing server side sessions. You will verify that the issue has been solved with the second deployment. You will learn about AWS Elastic Beanstalk, Amazon ElastiCache, and managing AWS resources in an AWS Elastic Beanstalk application via configuration files.

This lab will demonstrate:

- Putting your application code into version control.
- Deploying the application to AWS using AWS Elastic Beanstalk.

- Externalizing server side sessions to improve scalability.

Lab Pre-requisites

To successfully complete this lab, you should be familiar with basic Linux server administration and comfortable using the Linux command-line. You should also be familiar with web application concepts, like HTTP sessions and cookies. All required code and commands to be executed will be provided.

Other AWS Services

Other AWS Services than the ones needed for this lab are disabled by IAM policy during your access time in this lab. In addition, the capabilities of the services used in this lab are limited to what's required by the lab and in some cases are even further limited as an intentional aspect of the lab design. Expect errors when accessing other services or performing actions beyond those provided in this lab guide.

Start Lab

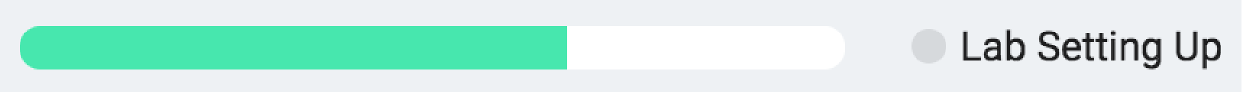
Notice the lab properties below the lab title:

- **setup** - The estimated time to set up the lab environment
- **access** - The time the lab will run before automatically shutting down
- **completion** - The estimated time the lab should take to complete

1. Launch your lab by clicking Start Lab

If you are prompted for a token, use the one distributed to you (or credits you've purchased).

A status bar shows the progress of the lab environment creation process (the AWS Management Console is accessible during lab resource creation, but your AWS resources may not be fully available until the process is complete).



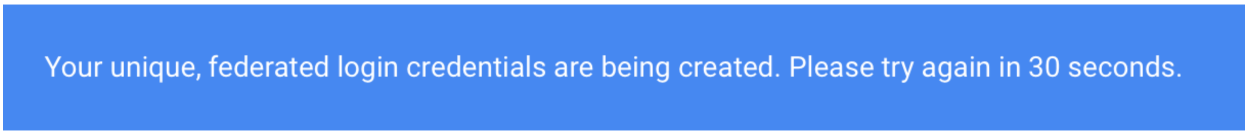
2. Open your lab by clicking Open Console

This will automatically log you into the AWS Management Console.

Please do not change the Region unless instructed.

Common login errors

Error : Federated login credentials



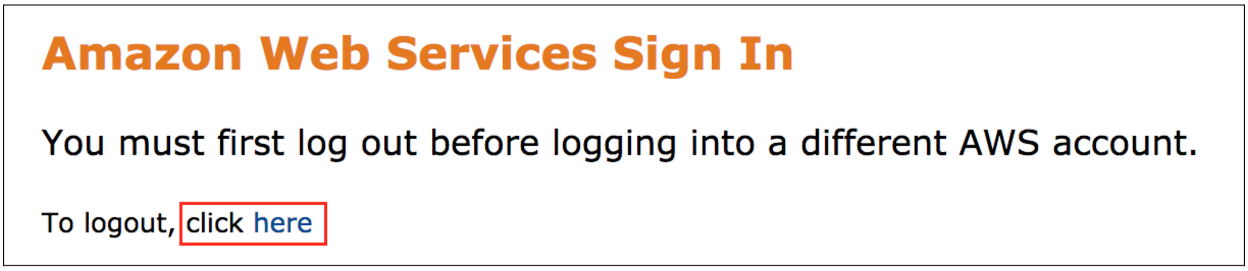
Your unique, federated login credentials are being created. Please try again in 30 seconds.

If you see this message:

- Close the browser tab to return to your initial lab window
- Wait a few seconds
- Click Open Console again

You should now be able to access the AWS Management Console.

Error: You must first log out



Amazon Web Services Sign In

You must first log out before logging into a different AWS account.

To logout, click here

If you see this message:

- Click To logout, click here
- Close the browser tab to return to your initial Qwiklabs window
- Click Open Console again

Task 1: Connect to Your EC2 Instance

In this task, you will connect to your DevServer EC2 instance using SSH. You will use this instance to download and manage your application code for Elastic Beanstalk. You will also use this instance to deploy your Elastic Beanstalk application.

Windows Users: Using SSH to Connect

These instructions are for Windows users only.

If you are using Mac or Linux, skip to the next section.

3. To the left of the instructions you are currently reading, click **Download PPK**.

4. Save the file to the directory of your choice.

You will use PuTTY to SSH to Amazon EC2 instances.

If you do not have PuTTY installed on your computer, download it here.

5. Open PuTTY.exe

6. Configure the PuTTY to not timeout:

- Click **Connection**
- Set **Seconds between keepalives** to

This allows you to keep the PuTTY session open for a longer period of time.

7. Configure your PuTTY session:

- Click **Session**
- **Host Name (or IP address):** Copy and paste the **DevServerIP** shown to the left of these instructions
- In the **Connection** list, expand **SSH**
- Click **Auth** (don't expand it)
- Click **Browse**
- Browse to and select the PPK file that you downloaded
- Click **Open** to select it
- Click **Open**

8. Click **Yes**, to trust the host and connect to it.

9. When prompted **login as**, enter:

This will connect to your EC2 instance.

10. Windows Users: Click here to skip ahead to the next task.

Mac and Linux Users

These instructions are for Mac/Linux users only.

11. To the left of the instructions you are currently reading, click **Download PEM**.

12. Save the file to the directory of your choice.

13. Copy this command to a text editor:

```
chmod 400 KEYPAIR.pem

ssh -i KEYPAIR.pem ec2-user@DevServerIP
```

14. Replace *KEYPAIR.pem* with the path to the PEM file you downloaded.

15. Replace *DevServerIP* with the values of DevServerIP shown to the left of these instructions.

16. Paste the updated command into the Terminal window and run it.

17. Type when prompted to allow a first connection to this remote SSH server.

Because you are using a key pair for authentication, you will not be prompted for a password.

Congratulations! You have successfully connected to your EC2 instance using SSH!

Task 2: Install git and Configure git Configuration Settings

In this task, you will install and configure git for your application.

18. Install git by:

- Entering

- Pressing **Enter**

19. Configure the git global user name by:

- Entering
- Replace **NAME** with your name
- Press **Enter**

20. Configure the git global user email by configuring the following:

- Enter
- Replace **EMAIL** with your email address
- Press **Enter**

Congratulations! You have successfully installed git and configured git's configuration settings. At this point, git knows that it should use the above values when making commits.

Task 3: Download The Application Code and Initialize The git Repository

You are ready to start writing code! Since this lab is not about programming web applications, but rather about deploying them on AWS, the code has been provided for you. In this task, you will create a working directory where you will download successive versions of the application code and commit the versions into version control.

21. Create a working directory where you will place your code and cd into the directory.

```
mkdir ~/code
cd ~/code
```

22. Download v1 of the application code to your current directory.

```
wget https://s3-us-west-2.amazonaws.com/us-west-2-aws-training/awsu-spl/spl-
45/scripts/v1.zip
```

Task 4: Unpack the Application

In this task, you will:

- Create a directory for your application code
- Unzip the application code into the app directory

You will be download successive versions of the code (right now you have v1) and update the code in version control.

23. Create a directory for your application and cd into the directory.

```
mkdir app

cd app
```

24. Unzip the zip the application zip file.

```
unzip ../v1.zip
```

You should see the output of the unzip command. The app directory should now consist of files and directories.

Task 5: Initialize the git Repository

In this task, you will commit your application code into version control and tag it.

25. Execute the following git commands to put your code into version control.

```
git init
git add -A .
git status
```

26. Commit your code by entering:

```
git commit -m "First commit: v1"
```

27. Tag your code by entering:

```
git tag -a v1 -m "v1: disk-based sessions"
```

You have just committed the first version of the code and also tagged it with a symbolic name "v1" (with a message describing what that milestone you have achieved with the tag) so that you can retrieve it later.

In real projects, you would iterate over the code many times, committing several versions, and testing. After several iterations would you would have your code the way you wanted and tagged with a symbolic name.

Task 6: Initialize AWS Elastic Beanstalk

You will now create an AWS Elastic Beanstalk "application" and "environment" and initialize the environment with the committed version of your code.

First, you need to install the AWS Elastic Beanstalk CLI package. This package provides the "eb" command uses.

28. Install the AWS Elastic beanstalk CLI package by running:

```
sudo pip install --upgrade awsebcli
```

29. Initialize the Elastic Beanstalk environment.

eb init sets default values for Elastic Beanstalk applications created with EB CLI by prompting you with a series of questions. It initializes the AWS Elastic Beanstalk configuration.

```
eb init
```

30. In the wizard, configure the following:

- **Select a default region:** Type the number that corresponds to the **Region** you are using. (The **Region** value is located to the left of these instructions.)
- **Enter Application Name:**
- **It appears you are using Python. Is this correct?:**
- **Select a platform version:** (Python 2.7)
- **Do you want to continue with AWS CodeCommit:**
- **Do you want to set up SSH for your instances?:**
- **Select a keypair:**

This will initialize your Elastic Beanstalk configuration with your application name and the intended environment.

The AWS Elastic Beanstalk configuration for your application has been created. The configuration is stored in the ".elasticbeanstalk" sub-directory under your application directory "app". If you are curious, you can inspect its contents to see what kind of information it contains.

Task 7: Deploy Your Application

In this task, you will start the AWS Elastic Beanstalk environment you configured in the previous task and deploy your application to it.

Create an Application Environment.

31. Copy and paste the **ebCreate** command string located to the left of these instructions into your SSH session.

The **eb create** command, creates a new environment and deploys an application version to it.

The **eb create** command has a significant number of parameters to run to create your environment. These parameters have been added to the command for you. The command creates the Beanstalk environment in a VPC with an RDS database instance and an elastic load balancer.

32. Use the following guide to answer the **eb create** command prompts.

- **Enter Environment Name:**
- **Enter DNS CNAME prefix:**
- Replace **Number** with a random number
- **Enter an RDS DB username:**
- **Enter an RDS DB master password:**
- **Retype password to confirm:**

Once completed, you should see a message like this:

```
WARNING: You have uncommitted changes.
Creating application version archive "app-v1-170524_001702".
Uploading spl45/app-v1-170524_001702.zip to S3. This may take a while.
Upload Complete.
```

The environment will take 5-8 minutes to initialize.

33. Exit log monitoring by pressing: **Control + C**.

34. View the Elastic Beanstalk events by entering:

This allows you to monitor the progress of the environment build.

35. Exit the **events** view by pressing

36. Check the status of your Elastic Beanstalk application by running:

Task 8: Switch to the New Elastic Beanstalk Console

The directions for your lab are written using the latest Elastic Beanstalk Management Console UI. In this task, you will determine if you are using the latest Management Console UI. If you are not using the latest, you will change to the latest management console UI.

37. In the **AWS Management Console** on the **Services** menu, click **Elastic Beanstalk**.

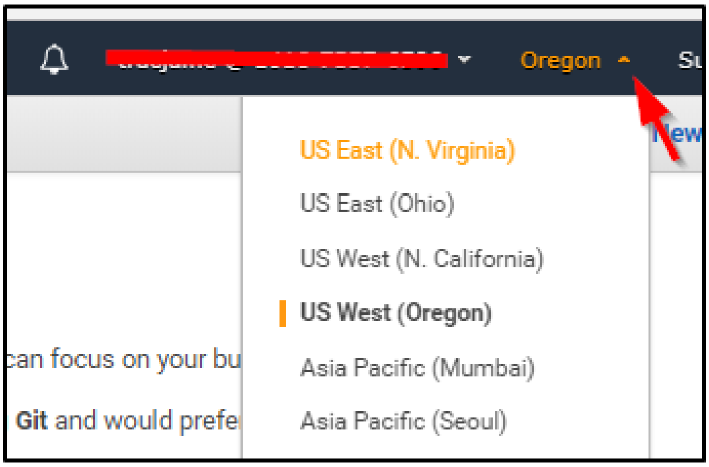
38. Switch to the region that you launched your Elastic Beanstalk application in.

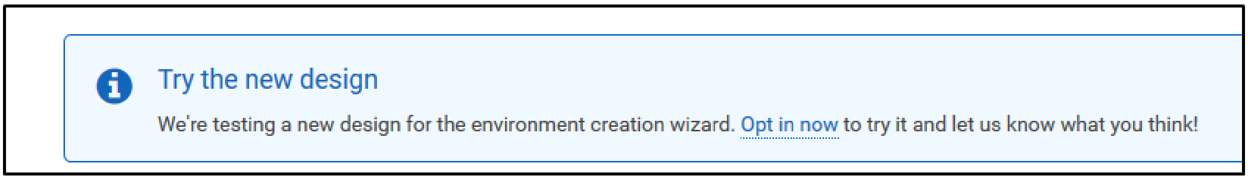
You can switch regions by:

- Clicking the drop-down at the top-right of the screen to the left of **Support**
- Selecting the correct region

39. At the top-right of the screen, click Create New Application

40. If you see the message: **Try the new design**:




Try the new design

We're testing a new design for the environment creation wizard. Opt in now to try it and let us know what you think!

- Click **Opt in now**.
- At the top-left of the screen, click the **Elastic Beanstalk** icon.

You will now be using the new Elastic Beanstalk management console UI and you will be at the main menu of Elastic Beanstalk.

41. If you do not see the message above, close the **Create New Application** window.

This means that you are already using the latest Elastic Beanstalk Management UI.

Task 9: Log into Your Elastic Beanstalk Application

In this task, you will:

- Log into your application that you created with Elastic Beanstalk
- Create a few blog entries in your application

42. Click **app-dev**.

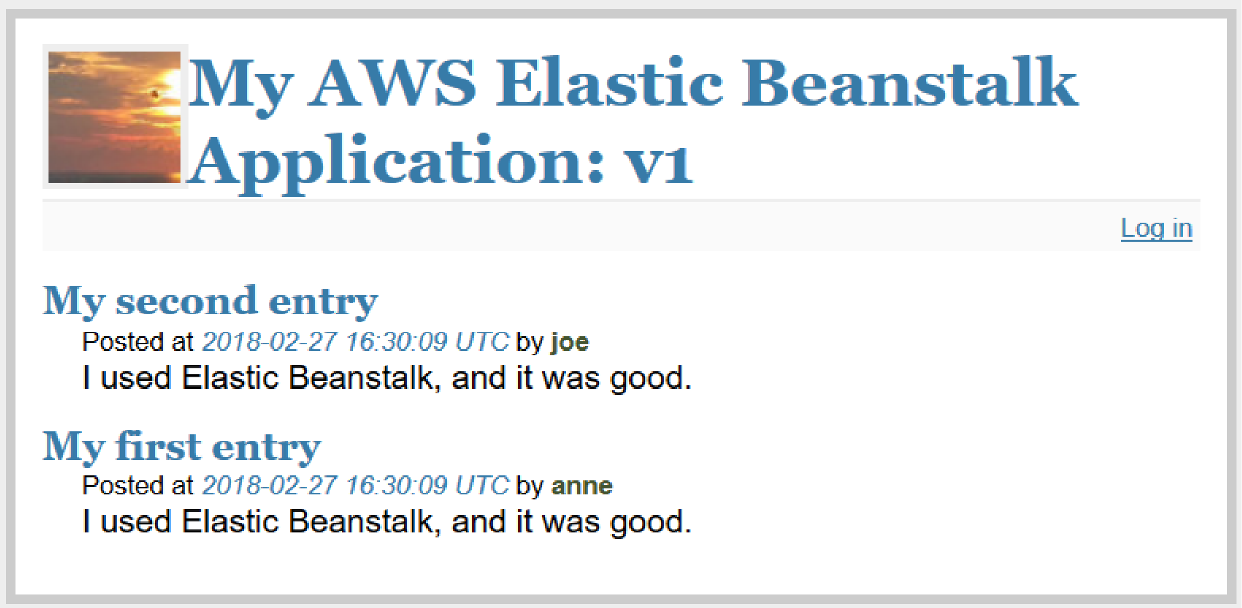
This also shows you the status of your application.

43. Wait until the health of your application displays **Ok**

It will take about 10 - 15 minutes till your application health displays **Ok**.

44. Click the environment URL at the top of the page.

This will display your application home page.



My AWS Elastic Beanstalk Application: v1

Log in

My second entry

Posted at *2018-02-27 16:30:09 UTC* by joe

I used Elastic Beanstalk, and it was good.

My first entry

Posted at *2018-02-27 16:30:09 UTC* by anne

I used Elastic Beanstalk, and it was good.

45. Click **Log in** to log into your application.

The application has two accounts already populated, for demonstration purposes. They are:

- **User 1:**
- **User 2:**
- **Password:**

46. Log into your application as **anne** or **joe**.

47. Add a few blog entries.

Next, you will force AWS Elastic Beanstalk to run your application on more than one instance. This will ensure that your application works seamlessly when the number of instances increases.

At this point, your application is running on only one instance.

Task 10: Scale the Environment to Two Amazon EC2 Instances

You have deployed version "v1" of your web application to Elastic Beanstalk and you are able to see the application using your browser. Currently, your application is only running on one instance.

Verify that Your Application is Only Running on One Instance

In this section, you will verify that your application is only running on one instance.

48. In the **AWS Management Console**, on the **Services** menu, click **EC2**.

49. In the left navigation pane, click **Instances**. You should see the following instances:

- **devserver** - The EC2 instance that you are running your Elastic Beanstalk and git commands on
- **app-dev** - The EC2 instance with the name of the AWS Elastic Beanstalk environment that you picked when running the "eb init" command

When your application scales, AWS Elastic Beanstalk will launch multiple instances to run it.

Make Elastic Beanstalk Launch Multiple Application Instances

50. On the **Services** menu, click **Elastic Beanstalk**.

51. Click **app-dev**.

52. In the left navigation pane, click **Configuration**.

This will allow you to modify the environment.

53. Under **Capacity**, click **Modify**, then configure:

- **Instances Min:**
- Scroll to the bottom of the screen, then click Apply

This will modify the minimum number of instances used for your application. You should see a message that Elastic Beanstalk is updating your environment.

You should also ensure that the Elastic Load Balancer is balancing requests across both instances.

54. In the **Services** menu, click **EC2**.

55. In the left navigation pane, click **Instances**.

You should see your new **app-dev** instance starting up.

56. Wait for your new **app-dev** instance to display the following:

- **Instance State:**
- **Status Checks:**

You can click **refresh** to update the status.

57. In the left navigation pane, click **Load Balancers**.

Your load balancer should be selected

58. Click the **Listeners** tab.

Notice that there is a listener on port 80 (HTTP) that forwards traffic to a specific target.

59. Click the target link that it is forwarding traffic to.

Your should now be in the load balancer **Target Groups** section. Your target group should be selected.

60. Click the **Targets** tab.

You should have two instances listed as **Registered targets** for your load balancer. These are the two app-dev instances that were launched for your application.

Task 11: Configure Cross-Zone Load Balancing

In this task, you will allow requests from your browser to go to both Amazon EC2 instances equally so that you can study the effect of having more than one Amazon EC2 instance. Without Cross-Zone Load Balancing, it is more likely that requests made in a short interval from a single browser will go to instances in a single Availability Zone. Since you have one Amazon EC2 instance in each Availability Zone, without Cross-Zone Load Balancing, all requests would go to the same Amazon EC2 instance.

Discover Your Application Behavior

61. Go back to the web application (My AWS Elastic Beanstalk Application: v1) window that you launched in your browser.

62. If you logged out, log back into your application as either anne or joe.

63. Refresh the window a few times and observe what happens each time.

64. Notice that sometimes you see the Home Screen, and sometimes you see the Blog Entry Screen.

The application behaves as if it has "forgotten" that you are already logged in, but after a refresh or two, it "remembers" again. Worse: you could have logged in with a different username and refreshing the screen would arbitrarily show that you are logged in as "anne" or "joe".

Externalize Your Sessions

Your application is running on more than one Amazon EC2 instance, but is not behaving properly with respect to sessions.

In this section, you will "externalize" the session storage to an external store that all Amazon EC2 instances running your application can access uniformly.

We will replace this with code that uses memcache to store sessions:

65. In your SSH session open your **application.py** file by entering:

Currently, your code is using a file to store sessions.

Current Code

```
session_opts = {
    'session.type': 'file',
    'session.data_dir': "./cache",
    'session.cookie_key': 'beaker',
    'session.cookie_expires': True,
    'session.timeout': 60
}
```

Next, you will update this so that your code uses memcache instead. This is the code that will be used later.

Updated Code

```
import settings
session_opts = {
    'session.type': 'ext:memcached',
    'session.lock_dir': './lock',
    'session.url': settings.MEMCACHED_ENDPOINT + ":" +
settings.MEMCACHED_PORT,
    'session.cookie_key': 'beaker',
    'session.cookie_expires': True,
    'session.timeout': 60
}
```

This code uses two values defined in the settings module. The settings module lives in the file **settings.py** and the relevant line below retrieves the memcache endpoint from the environment variable: **MEMCACHED_ENDPOINT = os.environ['MEMCACHED_ENDPOINT']** These modifications will make your code use memcache.

66. In the **AWS Management Console**, in the **Services** menu, click **Elastic Beanstalk**.

67. Click **app-dev**.

68. On the left navigation pane, click **Configuration**.

69. Under **Software**, click **Modify**.

70. Scroll down to the **Environment Properties** section, then configure:

- **Name:**
- **Value:** Paste the value of the **ElastiCacheClusterEndpoint**

The **ElastiCacheClusterEndpoint** is located to the left of these instructions.

71. Click Apply

Elastic Beanstalk will now deploy this environment variable to the instances hosting the application. This will take a few minutes to complete.

Update Your Code

A zip file with the updated code that contains the aforementioned changes (including the code to read the new MEMCACHED_ENDPOINT environment variable) has been prepared for you. Download and unzip the code into the app folder:

72. In your remote SSH session, download v2 of the application code to your code directory.

```
cd ~/code
wget https://s3-us-west-2.amazonaws.com/us-west-2-aws-training/awsu-spl/spl-
45/scripts/v2.zip
```

73. Navigate to the app directory and unzip v2 of the application code to the directory. You will overwrite the files.

```
cd app
unzip ../v2.zip
```

74. When asked to replace, enter

This will replace all of the files with the new code.

75. Commit the new code to git by running the following commands:

```
git add -A .

git commit -m "v2: memcache-based sessions"
```

76. Deploy the new application version to Elastic Beanstalk by running:

```
eb deploy
```

If you receive the message: **ERROR: InvalidParameterValueError - Environment named app-dev is in an invalid state for this operation. Must be Ready.**, wait a few minutes, then retry

77. Wait for a message to be displayed that says that the environment has been updated.

78. Log back into your application now and quickly refresh the browser several times.

You will not see the inconsistency that you saw before.

End Lab

Follow these steps to close the console, end your lab, and evaluate the experience.

79. Return to the AWS Management Console.

80. On the navigation bar, click **<yourusername>@<AccountNumber>**, and then click **Sign Out**.

81. Click End Lab

82. Click OK

83. (Optional):

- Select the applicable number of stars
- Type a comment
- Click **Submit**
    - 1 star = Very dissatisfied
    - 2 stars = Dissatisfied
    - 3 stars = Neutral
    - 4 stars = Satisfied
    - 5 stars = Very satisfied

You may close the dialog if you don't want to provide feedback.

Conclusion

Congratulations! You have now successfully:

- Deployed a web application to AWS using AWS Elastic Beanstalk
- Used git and eb command line tools to iteratively improve and deploy successive versions of your application
- Made your application scalable by externalizing sessions into an in-memory store
- Learned how to use Elastic Beanstalk configuration files to add AWS Resources to your application

Additional Resources

- AWS Training and Certification
- eb create

For feedback, suggestions, or corrections, please email us at aws-course-feedback@amazon.com.

Ready for more?

Here's another lab we think you'll like.

Hands-On Lab

Exploring Google Ngrams with Amazon EMR