Building Highly Available Web Application

SPL-TF-200-CPHAWA-1 - Version 1.0.8

Note: Do not include any personal, identifying, or confidential information into the lab environment. Information entered may be visible to others.

Corrections, feedback, or other questions? Contact us at *AWS Training and Certification*.

Lab overview

Example Corp. creates marketing campaigns for small-to-medium-sized businesses. They recently hired you to work with the engineering teams to build out a proof of concept for their business. To date, they host their clients using an on-premises data center and decided to move their operations to the cloud to save money and transform their business with a cloud-first approach. Some members of their team have cloud experience and recommended the AWS Cloud services to build their solution.

In addition, Example Corp. decided to redesign their web portal. Customers use the portal to access their accounts, create marketing plans, and run data analysis on their marketing campaigns. They would like to have a working prototype in two weeks. You must design an architecture to support this application. Your solution must be fast, durable, scalable, and more cost-effective than their existing on-premises infrastructure.

This lab showcases a mechanism to provision an auto scaling environment for a full stack web application using automation techniques to orchestrate AWS resources. IT teams can adapt this mechanism to rapidly provision infrastructure to securely deliver applications that can meet evolving business requirements.

Objectives

After completing this lab, you should be able to do the following:

- Deploy a virtual network spread across multiple Availability Zones in a Region using a provided CloudFormation template.
- Create a highly available and fully managed relational database across those Availability Zones using Amazon Relational Database Service (Amazon RDS).
- Create a database caching layer using Amazon ElastiCache.
- Use Amazon Elastic File System (Amazon EFS) to provision a shared storage layer across multiple Availability Zones for the application tier, powered by Network File System (NFS).
- Create a group of web servers that automatically scales in response to load variations to complete the application tier.

Duration

This lab requires approximately *90* minutes to complete.

Icon key

Various icons are used throughout this lab to call attention to different types of instructions and notes. The following list explains the purpose for each icon:

- **Expected output:** A sample output that you can use to verify the output of a command or edited file.

- **Note:** A hint, tip, or important guidance.
- **Learn more:** Where to find more information.
- **WARNING:** An action that is irreversible and could potentially impact the failure of a command or process (including warnings about configurations that cannot be changed after they are made).

AWS services not used in this lab

AWS services not used in this lab are disabled in the lab environment. In addition, the capabilities of the services used in this lab are limited to only what the lab requires. Expect errors when accessing other services or performing actions beyond those provided in this lab guide.

Lab environment

The following diagram shows the basic architecture of the lab environment:

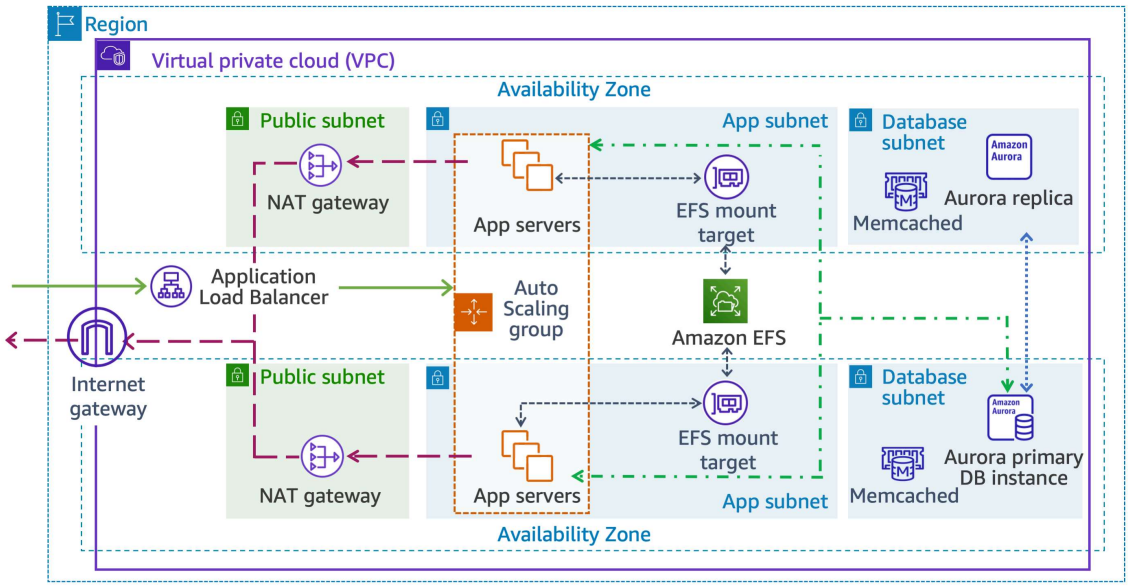

Image description: The preceding diagram depicts the data flow from an external user to an internet gateway, though a Application Load Balancer in a public subnet, to an application server in a private subnet, to a database server in a separate private subnet.

The following list details the major resources in the diagram:

- A single AWS Region with one VPC and two Availability Zones.
- Each Availability Zone contains a public subnet, an app subnet, and a database subnet.
- The NAT gateways are located in the public subnet of each Availability Zone.
- An Application Load Balancer and an Auto Scaling group that has app servers in the app subnets of both Availability Zones.
- Each app server communicates with an EFS mount target in its own subnet to reach the Amazon EFS file system, which is not inside an Availability Zone.
- All app servers communicate with an Aurora primary DB instance in one of the database subnets. The other database subnet holds the Aurora replica.
- An Amazon ElastiCache in database subnet that acts as a caching layer.

Start lab

1. To launch the lab, at the top of the page, choose **Start lab**.

You must wait for the provisioned AWS services to be ready before you can continue.

2. To open the lab, choose **Open Console**.

You are automatically signed in to the AWS Management Console in a new web browser tab.

Do not change the Region unless instructed.

Common sign-in errors

Error: You must first sign out

If you see the message, **You must first log out before logging into a different AWS account:**

- Choose the **click here** link.
- Close your **Amazon Web Services Sign In** web browser tab and return to your initial lab page.
- Choose **Open Console** again.

Error: Choosing Start Lab has no effect

In some cases, certain pop-up or script blocker web browser extensions might prevent the **Start Lab** button from working as intended. If you experience an issue starting the lab:

- Add the lab domain name to your pop-up or script blocker's allow list or turn it off.
- Refresh the page and try again.

Task 1: Configure the Network

In this task, you use a CloudFormation template to deploy the networking resources required to support the application.

Task 1.1: Navigate to the CloudFormation console

3. At the top of the page, in the unified search bar, search for and choose `CloudFormation`.

The **CloudFormation** page is displayed.

Task 1.2: Create the CloudFormation stack

4. Choose **Create stack**.

Note: If the console starts you on the Stacks page instead of the Amazon CloudFormation landing page, then you can get to the Create stack page in two steps.

- Choose Create stack .
- Choose With new resources (standard) .

The **Create Stack** page is displayed.

5. Configure the following:

- Select **Template is ready**.
- Select **Amazon S3 URL**.
- Copy the **Task1TemplateUrl** value from the left side of these lab instructions and paste it in the **Amazon S3 URL** text box.
- Choose **Next** .

The **Specify stack details** page is displayed.

6. Set the **Stack name** as

   `VPCStack` .

7. Leave the **Parameters** set to the default values.
8. Choose **Next** .

The **Configure stack options** page is displayed. You can use this page to specify additional parameters. You can browse the page, but leave settings at the default values.

9. Choose **Next** .

The **Review** page is displayed. This page is a summary of all settings.

10. At the bottom of the page, choose **Submit** .

The **stack details** page is displayed.

While the stack is being created, it's listed on the Stacks page with a status of **CREATE_IN_PROGRESS**.

11. Choose the **Stack info** tab.
12. Occasionally choose the console refresh .
13. Wait for the Stack status to change to CREATE_COMPLETE .

Note: This Stack can take up to 5 minutes to deploy the resources.

Task 1.3: View created resources from the console

14. Choose the **Resources** tab.

The list shows the resources that are being created. CloudFormation determines the optimal order for resources to be created, such as creating the VPC before the subnet.

15. Review the resources that were deployed in the stack.
16. Choose the **Events** tab and scroll through the list.

The list shows (in reverse order) the activities performed by CloudFormation, such as starting to create a resource and then completing the resource creation. Any errors encountered during the creation of the stack is listed in this tab.

17. Choose the **Outputs** tab.
18. Review the key value pairs in the Outputs section. These values may be useful in later lab tasks.

Congratulations! You have learned to configure the stack and created all of the resources using the provided CloudFormation template.

Task 2: Create an Amazon RDS database

In this task, you deploy a highly available database for use by WordPress.

Task 2.1: Navigate to the Amazon RDS console

19. At the top of the page, in the unified search bar, search for and choose `RDS`.

The **Amazon RDS console** page is displayed.

Task 2.2: Create a new Amazon Aurora database

20. In the left navigation pane, choose **Databases**.
21. Choose **Create database**.

The **Create database** page is displayed.

22. In the **Choose a database creation method** section, select **Standard create**.
23. In the **Engine options** section, configure the following:

- For **Engine type**, select **Aurora (MySQL Compatible)**.

24. In the **Templates** section, select **Production**.
25. In the **Settings** section, configure the following:

- For **DB cluster identifier**, enter `MyDBCluster`
- For **Master username**, enter `admin`
- For **Master password**, paste the **LabPassword** value from the left side of these lab instructions.
- **Confirm master password**, paste the **LabPassword** value from the left side of these lab instructions.

26. In the **Instance configuration** section:

- For **DB instance class**, select **Burstable classes**.
- For **instance type**, select **db.t3.medium**.

27. In the **Availability & durability** section, for **Multi-AZ deployment**, select **Create an Aurora Replica or Reader node in a different AZ**.
28. In the **Connectivity** section:

- For **Virtual private cloud (VPC)**, select **LabVPC**.
- For **DB subnet group**, select **labdbsubnetgroup**.
- For **Public access**, select **No**.
- For **VPC security group**, select **Choose existing**.
- For **Existing VPC security groups**:
  - Select **xxxxx-RDSSecurityGroup-xxxxx**.
  - To remove the default security group, choose the **X**.
- Expand the **Additional configuration** section and configure the following:
  - **Database port**: Leave the configuration at the default value.

29. In the **Monitoring** section, deselect **Enable Enhanced monitoring**.
30. Scroll to the bottom of the page and expand the main **Additional configuration** section.

- In the **Database options** section:

- For **Initial database name**, enter `WPDatabase`

- In the **Encryption** section, deselect **Enable encryption**.
- In the **Maintenance** section, deselect **Enable auto minor version upgrade**.
- In the **Deletion protection** section, deselect **Enable deletion protection**.

31. Scroll to the bottom of the screen and choose **Create database**.
32. On the **Suggested add-ons for mydbcluster** pop-up window, choose **Close**

Note: Your Aurora MySQL DB cluster is in the process of launching. The cluster you configured consists of two instances, each in a different Availability Zone. The Amazon Aurora DB cluster can take up to 5 minutes to launch. Wait for the mydbcluster status to change to *Available*. You do not have to wait for the availability of the instances to continue.

Expected service output:

```
Successfully created database mydbcluster
```

33. Choose **View connection details** displayed on the success message border to save the connection details of your **mydbcluster** database to a text editor.
34. On the Connection details to your database **mydbcluster** pop-up window, choose **Close**.

Task 2.4: Copy database metadata

35. In the left navigation pane, choose **Databases**.
36. Choose the **mydbcluster** link.
37. Choose the **Connectivity & security** tab.
38. Copy the **endpoint** value for the **Writer** instance to a text editor.

Note: To copy the Writer instance endpoint, hover on it and choose the copy icon.

39. Choose the **Configuration** tab.
40. Copy the **Master username** value to a text editor.
41. For **Master password**, use the **LabPassword** value from the left side of these lab instructions.
42. In the left navigation pane, choose **Databases**.
43. Choose the **mydbcluster-instance-x** writer instance link.
44. Choose the **Configuration** tab.
45. Copy the **DB name** value to a text editor.

Additional information: WordPress uses its database to store articles, users, and configuration information.

Congratulations! You have successfully created a highly available & fully managed relational database across those availability zones using Amazon RDS.

Task 3: Create an Amazon ElastiCache for Memcached

In this task, you create a database caching layer using Amazon ElastiCache. This provides a cache around the database for frequently run queries, improving HTTP response time performance and reducing strain on the database instantiated in the previous task.

Task 3.1: Navigate to the Amazon ElastiCache console

46. At the top of the page, in the unified search bar, search for and choose `ElastiCache`.

The **Amazon ElastiCache** page is displayed.

Task 3.2: Create the Amazon ElastiCache cluster

47. In the left navigation pane, under **Resources**, choose **Memcached caches**.
48. Choose **Create Memcached cache**.

The **Cluster settings** page is displayed.

49. On the **Create Memcached cache** page, in the **Choose a cluster creation method** section:
    - For **Deployment option**, choose **Design your own cache**.
    - Next, choose **Standard create**.
50. In the **Cluster info** section:
    - For **Name**, enter `MyWPCache`.
51. In the **Cluster settings** section:
    - For **Node type**, select **cache.t3.micro**.
    - For **Number of nodes**, enter `2`.
52. Choose **Next**.
53. In the **Selected security groups** section:
    - Choose **Manage**.

The **Manage security groups** window is displayed.

54. Select **xxx-ElastiCacheSecurityGroup-xxx** and then **Choose**.
55. Choose **Next**.
56. Choose **Create**.

Expected service output:

```
The cluster was created successfully
```

Congratulations! You have successfully created a database caching layer using Amazon ElastiCache for use by WordPress.

Task 4: Create an Amazon EFS file system

In this task, you provision a shared storage layer using Amazon EFS that creates an NFS cluster across multiple availability zones.

Task 4.1: Navigate to the EFS console

57. At the top of the page, in the unified search bar, search for and choose `EFS`.

The **Amazon Elastic File System** page is displayed.

Task 4.2: Create a new file system

58. Choose **Create file system**.
59. On the **Create file system** page, choose **Customize**.
60. On the **File system settings** page, in the **General** section:
    - For **Name**, enter `myWPEFS`
    - Deselect **Enable automatic backups**.
    - In the **Tags - optional** section:
        - For **Tag key**, enter `Name`
        - For **Tag value – optional**, enter `myWPEFS`
    - Leave all other settings at their default value.
61. Choose **Next**.
62. On the **Network access** page:
    - For **Virtual Private Cloud (VPC)**, select `LabVPC`.
63. On the **Mount targets** page:
    - For **Availability Zone**, select the **Availability Zone ending in "a"**.
    - For **Subnet ID**, select **AppSubnet1**.
    - For **Security group**, select **xxxxx-EFSMountTargetSecurityGroup-xxxxx**
    - To remove the **default** Security group, choose the **X**.
    - For **Availability Zone**, select **Availability Zone ending in "b"**.
    - For **Subnet ID**, select **AppSubnet2**.
    - For **Security group**, select **xxxxx-EFSMountTargetSecurityGroup-xxxxx**.
    - To remove the **default** Security group, choose the **X**.
64. Choose **Next**.
65. On the **File system policy – *optional*** page, choose **Next**.

Note: Configuring this page is not necessary in this lab.

66. On the **Review and create** page, scroll to the bottom of the page and choose **Create**.

Expected service output:

```
Success! File system (fs-xxxxxxx) is available
```

Note: The file system state displays as *Available* after several minutes.

67. Copy the **File system ID** generated for *myWPEFS* to a text editor. It has a format like *fs-a1234567*.

Congratulations! You have successfully created the Amazon EFS that created the NFS cluster across multiple availability zones.

Task 5: Create an Application Load Balancer

In this task, you create the Application Load Balancer and a target group.

Task 5.1: Navigate to the Amazon EC2 console

68. At the top of the page, in the unified search bar, search for and choose `EC2`.

The **EC2 Dashboard** page is displayed.

Task 5.2: Create a Target group

69. In the left navigation pane, choose **Target Groups**.
70. Choose **Create target group**.
71. On the **Specify group details** page, in the **Basic configuration** section:

- For **Choose a target type**, select **Instances**.
- For **Target group name**, enter `myWPTargetGroup`
- For **VPC**, select **LabVPC**.

72. On the **Specify group details** page, in the **Health checks** section:

- For **Health check path**, enter `/wp-login.php`
- Expand the **Advanced health check settings** section and configure the following:
  - For **Healthy threshold**, enter `2`.
  - For **Unhealthy threshold**, enter `10`.
  - For **Timeout**, enter `50`.
  - For **Interval**, enter `60`.

Leave the remaining settings on the page at their default values.

73. Choose **Next**.
74. On the **Register targets** page, scroll to the bottom of the page and choose **Create target group**.

Note: There are no targets to register currently.

Expected service output:

```
Successfully created target group: myWPTargetGroup
```

Task 5.3: Create an Application Load Balancer

75. In the left navigation pane, choose **Load Balancers**.
76. Choose **Create load balancer**.
77. In the Application Load Balancer section, choose **Create**.
78. On the **Create Application Load Balancer** page, in the **Basic Configuration** section:

- For **Load balancer name**, enter `myWPAppALB`.

79. On the **Create Application Load Balancer** page, in the **Network mapping** section:

- For **VPC**, choose `LabVPC`.
- For **Mappings**:
  - Select the first Availability Zone listed, and choose `PublicSubnet1` from the Subnet dropdown menu.
  - Select the second Availability Zone listed, and choose `PublicSubnet2` from the Subnet dropdown menu.

80. On the **Create Application Load Balancer** page:

- In the **Security groups** section:
  - From the **Security groups** dropdown menu, choose **xxxxx-AppInstanceSecurityGroup-xxxxx**.
  - To remove the **default** security group, choose the **X**.
- In the **Listeners and routing** section:
  - For **Listener HTTP:80**, choose **myWPTargetGroup** for the **Default action**.

81. Scroll to the bottom of the page and choose **Create load balancer**.

Expected service output:

```
Successfully created load balancer:myWPAppALB
```

82. Choose **View load balancer**.

Note: The load balancer is in the *Provisioning* state for few minutes and then changes to *Active*. Wait for the load balancer **State** to change to **Active**.

83. Copy the **DNS name** to a text editor.

WARNING: Verify that all the resources from previous tasks are created successfully before continuing to the next task.

Congratulations! You have created the target group and a public facing load balancer.

Task 6: Create a launch template using CloudFormation

In this task, you use a CloudFormation template to deploy the WordPress user data within an Amazon Elastic Compute Cloud (Amazon EC2) Auto Scaling launch template. The template includes the EFS mount points and the Aurora configurations.

Task 6.1: Navigate to the CloudFormation console

84. At the top of the page, in the unified search bar, search for and choose `CloudFormation`.

The **CloudFormation** page is displayed.

Task 6.2: Create the CloudFormation stack

85. Choose **Create stack**.

Note: If the console starts you on the Stacks page instead of the Amazon CloudFormation landing page, then you can get to the Create stack page in two steps.

- Choose **Create stack**.
- Choose **With new resources (standard)**.

86. On the **Create Stack** page:

- Select **Template is ready**.
- Select **Amazon S3 URL**.
- Copy the **Task6TemplateUrl** value from the left side of these lab instructions and paste it in the **Amazon S3 URL** text box.
- Choose **Next**.

87. On the **Specify stack details** page:

- In the **Provide a stack name** section, for **Stack name**, enter `WPLaunchConfigStack`.
- In the **Parameters** section, for **DB Name**, paste the **initial database name** you copied in Task 2.

Note: Make sure that you paste the *initial database name*, not the cluster name.

- For **Database endpoint**, paste the **writer endpoint** you copied in Task 2.
- For **Database User Name**, paste the **Master username** you copied in Task 2.
- For **Database Password**, paste the **LabPassword** value from the left side of these lab instructions.
- For **WordPress admin username**, defaults to `wpadmin`.
- For **WordPress admin password**, paste the **LabPassword** value from the left side of these lab instructions.
- For **WordPress admin email address**, input a valid email address.
- For **Instance Type**, leave the default value of **t3.medium**.
- For **ALBDnsName**, paste the **DNS name** value you copied in Task 5.
- For **LatestAL2AmiId**, leave the default value.
- For **WPElasticFileSystemID**, paste the **File system ID** value you copied in Task 4.

88. Choose **Next**.

89. On the **Configure stack options** page, choose **Next**.

Note: You can use this page to specify additional parameters. You can browse the page, but leave settings at their default values.

90. On the **Review** page, scroll to the bottom of the page and choose **Submit**.

Note: The **Review** page is a summary of all settings.

The **stack details** page is displayed.

While the stack is being created, it's listed on the Stacks page with a status of **CREATE_IN_PROGRESS**.

91. Choose the **Stack info** tab.

92. Occasionally choose the console refresh.

93. Wait for the stack status to change to CREATE_COMPLETE.

Note: This stack can take up to 5 minutes to deploy the resources.

Task 6.3: View created resources from the console

94. Choose the **Resources** tab.

The list shows the resources that are created.

Congratulations! You have created the Launch template which is used to launch WordPress servers.

Task 7: Create the application servers by configuring an Auto Scaling group and a scaling policy

In this task, you create the WordPress application servers by configuring an Auto Scaling group and a scaling policy.

Task 7.1: Creating an Auto Scaling group

95. At the top of the page, in the unified search bar, search for and choose `EC2`.

The **EC2 Dashboard** page is displayed.

96. In the left navigation pane, under the **Auto Scaling** section, choose **Auto Scaling Groups**.
97. Choose **Create Auto Scaling group**.
98. On the **Choose launch template or configuration** page:

- In the **Name** section, for **Auto Scaling group name**, enter `WP-ASG`.
- In the **Launch template** section, for **Launch Template**, select **LabLaunchTemplate**.

99. Choose **Next**.
100. On the **Choose instance launch options** page, in the **Network** section:

- For **VPC**, select **LabVPC**.
- For **Availability Zones and subnets**, choose **AppSubnet1** and **AppSubnet2**.

101. Choose **Next**.
102. On the **Configure advanced options** page, configure the following:

- For **Load balancing**, select **Attach to an existing load balancer**.
- For **Attach to an existing load balancer**, select **Choose from your load balancer target groups**.
- For **Existing load balancer target groups**, select **myWPTargetGroup | HTTP**.
- For **Health checks**, select **Turn on Elastic Load Balancing health checks**.
- For **Health check grace period**, leave at the default value of 300 or more.
- For **Monitoring**, select **Enable group metrics collection within CloudWatch**.

103. Choose **Next**.
104. On the **Configure group size and scaling - *optional*** page:

- In the **Group Size** section, for **Desired capacity**, enter `2`.
- In the **Scaling** section:
  - For **Minimum capacity**, enter `2`
  - For **Maximum capacity**, enter `4`
- In the **Automatic scaling - *optional*** section, select the **Target tracking scaling policy** option.

The remaining settings on this section can be left at their default values.

105. Choose **Next**.
106. In the **Add notifications** page, choose **Next**.
107. In the **Add tags** page, choose **Add tag** and in the **Tags (1)** section:
    - For **Key**, enter `Name`.
    - For **Value**, enter `wp-ha-app`.
108. Choose **Next**.
109. On the **Review** page, review the Auto Scaling group configuration for accuracy, and then at the bottom of the page, choose **Create Auto Scaling group**.

Expected service output:

```
WP-ASG, 1 Scaling policy created successfully. Group metrics collection is enabled.
```

Now that you have created your Auto Scaling group, you can verify that the group has launched your EC2 instances.

110. Choose the Auto Scaling group **WP-ASG** link.
111. To review information about the Auto Scaling group, examine the **Group Details** section.
112. Choose the **Activity** tab.

The **Activity History** section maintains a record of events that have occurred in your Auto Scaling group. The Status column contains the current status of your instances. When your instances are launching, the status column shows *PreInService*. The status changes to *Successful* once an instance is launched.

113. Choose the **Instance management** tab.

Your Auto Scaling group has launched two Amazon EC2 instances and they are in the *InService* lifecycle state. The Health Status column shows the result of the Amazon EC2 instance health check on your instances.

If your instances have not reached the *InService* state yet, you need to wait a few minutes. You can choose the refresh button to retrieve the current lifecycle state of your instances.

114. Choose the **Monitoring** tab. Here, you can review monitoring-related information for your Auto Scaling group.

This page provides information about activity in your Auto scaling group, as well as the usage and health status of your instances. The **Auto Scaling** tab displays Amazon CloudWatch metrics about your Auto Scaling group, while the **EC2** tab displays metrics for the Amazon EC2 instances managed by the Auto Scaling group.

Task 7.2: Verify the target groups are healthy

115. In the left navigation pane, choose **Target Groups**.
116. Choose the **myWPTargetGroup** link.
117. In the **Targets** tab, wait until the instance Health status is displayed as *healthy*.

Note: It can take up to 5 minutes for the health checks to show as healthy.

Task 7.3: Login to the application

118. In the left navigation pane, choose **Load Balancers**.
119. Copy the **DNS name** to a text editor and append the value `/wp-login.php` to the end of the DNS name to complete your WordPress application URL.

Expected output:

myWPAppELB-4e009e86b4f704cc.elb.us-west-2.amazonaws.com/wp-login.php

120. Paste the WordPress application URL value into a new browser tab.

The **WordPress login** page is displayed.

121. On the **WordPress login** page:
    - For **Username or Email Address**, enter `wpadmin`.
    - For **Password**, paste the **LabPassword** value from the left side of these lab instructions.
122. Choose the **Log in** button.

The **WordPress** management website is displayed.

Note: Keep the browser tab open, you will return to it in later task.

Congratulations! You have now created a highly-available auto-scaling deployment of WordPress application that scales in and out.

Task 8: Chaos testing with AWS Fault Injection Simulator

In this task, you test the application high availability using chaos engineering. You randomly terminate EC2 instances and see if the auto scaling group and load balancer can reroute the traffic to healthy hosts and start new EC2 instances to keep the required capacity.

Additional information: However, turning off EC2 instances manually is not a scalable solution. You use the **AWS Fault Injection Simulator** to automate this test.

Task 8.1: Navigate to the console

123. At the top of the page, in the unified search bar, search for and choose `AWS FIS`.

The **AWS FIS** page is displayed.

Task 8.2: Test scenario and assumptions

Additional information: To run a successful fault injection test you need to form an assumption and test it. The assumption for this experiment is:

Example: If one of the Availability Zone (AZ) has an outage, and all the EC2 in that AZ stops, our web application remains available. The load balancer routes the traffic to other AZs and the auto scaling group starts new EC2s to keep the required number of instances.

124. From **Create experiment from scenario** drop-down menu, choose **Create experiment template**.
125. On the **Account targeting** pop-up page, choose the option **This AWS account** and choose **Confirm**.
126. On the **Create experiment template** page, in the **Description and name** section:
    - For **Description**, enter `Terminate instances in an AZ to simulate AZ outage`
    - For **Name - optional**, enter `TerminateInstancesinAZ`
127. In the **Actions** section:
    - Choose **Add action**.
    - For **Name**, enter `TerminateInstances`

- For **Action type**, select **EC2** and **aws:ec2:terminate-instances**.
- Choose **Save** .

128. In the **Targets** section, configure:

- Choose **Edit** .
- For **Target method**, select **Resource tags, filters and parameters**.
- For **Resource tags**:
  - Choose **Add new tag** .
    - For **Key**, enter `Name`
    - For **Value**, enter `wp-ha-app`
- For **Resource filters - *optional***:
  - Choose **Add new filter** .
    - For **Attribute path**, enter `Placement.AvailabilityZone`
    - For **Values**, enter **LabRegion** appended with “a” at the end.

      Expected value: *us-west-2a*
  - Choose **Add new filter** .
    - For **Attribute path**, enter `State.Name`
    - For **Values**, enter `running` .
- Choose **Save** .

129. In the **Service Access** section, configure:

- Select **Use an existing IAM role**.
- For **IAM role**, select **xxxx-FISRole-xxxx**.

130. Choose **Create experiment template** .

The **Create experiment template** window is displayed.

131. For **To confirm that you want to create an experiment template without a stop condition, enter create in the field**, enter `create` .

132. Choose **Create experiment template** .

Expected service output:

> You successfully created experiment template EXT9VzxxxxX / Terminate instances in an AZ to simulate AZ outage.

Task 8.3: Start the experiment

133. In the left navigation pane, choose **Experiment templates**.

134. Choose the **Experiment template ID** link.

135. Choose **Start experiment** .

136. Choose **Start experiment** .

The **Start experiment** window is displayed.

137. For **To confirm that you want to start the experiment, enter start in the field:**, enter

`start`.

138. Choose **Start experiment**.

Expected service output:

```
You successfully started experiment EXPU8kxxxxxE2.
```

Note: Wait for the State to change to **Completed**.

Task 8.4: Observe the experiment

139. Return to the WordPress website browser tab and refresh the page every few seconds.

Expected output: The website should continue to load correctly.

140. Return to the AWS Management Console and search for

`EC2`.

141. In the left navigation pane, choose **Auto Scaling Groups**.

142. Select **WP-ASG**.

143. Choose the **Activity** tab.

Expected output: You should start to see that the EC2 instance in availability zone ending in "a" was taken out of service. It might take a few minutes for the health check to detect the EC2 is unhealthy, so keep refreshing the Instance list. You should also see new instances being launched by the auto scaling group to replenish the lost capacity.

Congratulations! You have successfully conducted the experiment, the WordPress site remained functional. The website auto-healed from the AZ outage and website is back to the desired capacity in just a few minutes. This verifies highly-available architecture is working as intended.

Challenge yourself!

Speedup the WordPress Website

Challenge Task

Add a content delivery network using Amazon CloudFront to speed up global delivery of the WordPress application then replace the Site Address (URL) with the CloudFront DNS hostname and view the public facing WordPress site created in this lab.

Navigate **here** for a solution.

Additional information: Amazon CloudFront can speed up the delivery of your websites, whether its static objects (e.g., images, style sheets, JavaScript, etc.) or dynamic content (e.g., videos, audio, motion graphics, etc.), to viewers across the globe. The CDN offers a multi-tier cache by default that improves latency and lowers the load on origin servers when the object is not already cached at the Edge.

Optional Task

You created an Elasticache for Memcached instance in Task 3. Now, configure the WordPress to use that service. To improve the site's response time and to reduce the strain on your backend database, WordPress is configured to use Memcached as a caching layer for common requests.

Note: For the purposes of testing, you can also add dummy content to the WordPress website.

Conclusion

Congratulations! You now have successfully:

- Deployed a virtual network spread across multiple Availability Zones in a Region using a provided CloudFormation template.
- Created a highly available and fully managed relational database across those Availability Zones using Amazon Relational Database Service (Amazon RDS).
- Created a database caching layer using Amazon ElastiCache.
- Used Amazon Elastic File System (Amazon EFS) to provision a shared storage layer across multiple Availability Zones for the application tier, powered by Network File System (NFS).
- Created a group of web servers that would automatically scale in response to load variations to complete the application tier.

End lab

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

144. Return to the **AWS Management Console**.
145. At the upper-right corner of the page, choose **AWSLabsUser**, and then choose **Sign out**.
146. Choose **End lab** and then confirm that you want to end your lab.

Appendix

Challenge task solution

Navigate to the CloudFormation console

147. At the top of the page, in the unified search bar, search for and choose `CloudFormation`.

Create the CloudFormation stack

148. Choose **Create stack**.

Note: If the console starts you on the Stacks page instead of the Amazon CloudFormation landing page, then you can get to the Create stack page in two steps.

- Choose **Create stack**.
- Choose **With new resources (standard)**.

The **Create Stack** page is displayed.

149. Configure the following:

- Select **Template is ready**.
- Select **Amazon S3 URL**.
- Copy the **ChallengeTaskTemplateUrl** value from the left side of these lab instructions and paste it in the **Amazon S3 URL** text box.
- Choose **Next**.

The **Specify stack details** page is displayed.

150. Set the **Stack name** as

`WPCloudFrontStack`.

151. Configure the following **parameters**:
    - For **CloudFront Certificate ARN**, leave this optional parameter empty.
    - For **Domain name**, leave this optional parameter empty.
    - For **ALBDnsName**, paste the load balancer **DNS name** value you copied in Task 4.
152. Choose **Next**.

The **Configure stack options** page is displayed. You can use this page to specify additional parameters. You can browse the page, but leave settings at their default values.

153. Choose **Next**.

The **Review** page is displayed. This page is a summary of all settings.

154. Scroll to the bottom of the page and choose **Submit**.

The **stack details** page is displayed.

While the stack is being created, it's listed on the Stacks page with a status of **CREATE_IN_PROGRESS**.

155. Choose the **Stack info** tab.
156. Occasionally choose the console refresh.
157. Wait for the stack status to change to CREATE_COMPLETE.

Note: This stack can take up to 5 minutes to deploy the resources.

View created resources from the console

158. Choose the **Resources** tab.

The list shows the resources that are created.

159. Choose the **Outputs** tab.
160. Copy the **DnsHostname** value to a Text Editor.

Login to WordPress Admin console

161. Return to WordPress Management site.
162. In the left menu, navigate to **Settings** > **General** tab.
163. Replace the **Site Address(URL)** with the **DNSHostname** copied earlier.
164. Browse to the **CloudFront DNS name** to view the public facing WordPress site.

Additional information: The CloudFront distribution caches the content, improves latency for users across the globe and lowers the load on origin WordPress servers.

Return to the instructions

For more information about AWS Training and Certification, see *https://aws.amazon.com/training/*.

Your feedback is welcome and appreciated.
If you would like to share any feedback, suggestions, or corrections, please provide the details in our *AWS Training and Certification Contact Form*.