Getting Started with AWS

Hosting a Web App

Getting Started with AWS: Hosting a Web App

Table of Contents

Hosting a Web App on Amazon Web Services

A *web app* is any software that users access through a web browser or specialized web client. Web apps are typically structured into logical tiers. For example, a common structure uses three tiers. The first tier is the web browser, which is responsible for presenting the user interface. The middle tier is an application server, which is responsible for the application's functionality. The third tier is a database server or file system, which is responsible for data storage.

This tutorial walks you through the process of hosting a scalable, robust web app on AWS infrastructure. We'll deploy a sample app, demonstrating best practices. By the end of this tutorial, you should be able to do the following:

- Create a virtual server, called an *EC2 instance*, and use it as an application server in the cloud.
- Create a database server, called a *DB instance*.
- Deploy a sample web app to the application server.
- Set up scaling and load balancing to distribute traffic across a minimum number of application servers.
- Associate a domain name with your web app.

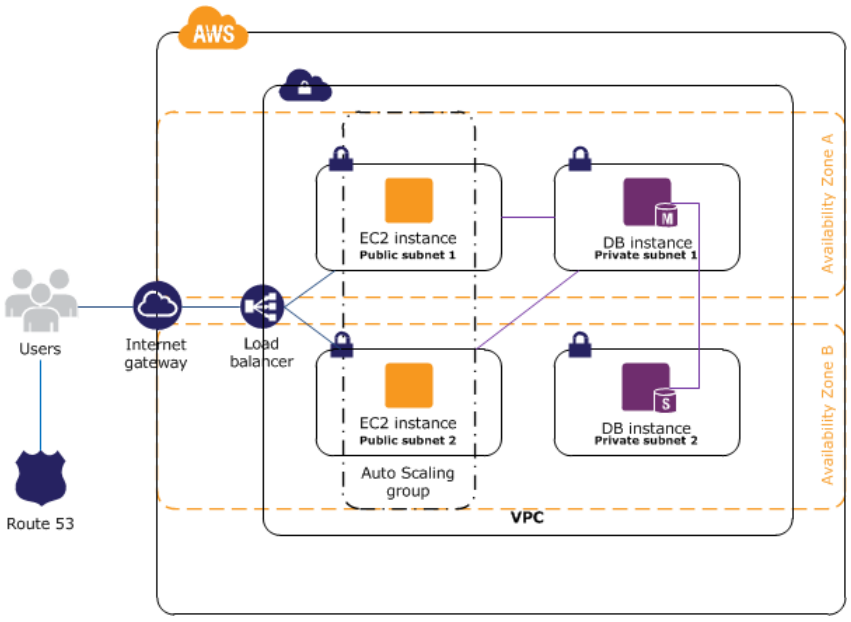
Web App Hosting Architecture

Before you create and deploy a web app, you must design your architecture to ensure that it meets your requirements. The following table shows how Amazon EC2, Amazon EBS, Amazon S3, Auto Scaling, Elastic Load Balancing, Amazon CloudWatch, Amazon Route 53, and Amazon CloudFront work together to provide a seamless and cost-effective architecture.

Requirement	Solution
Low-cost, reliable application and database servers	• Amazon EC2 provides virtual servers in the cloud. You control the protocols, ports, and source IP address ranges that can access your virtual servers. • Amazon EBS provides a persistent file system for Amazon EC2 virtual servers. • Amazon RDS provides a cost-efficient and resizable database server that's easy to administer.

Requirement	Solution
An easy way to provision servers to handle peak capacity without incurring costs when the extra capacity isn't needed	• Elastic Load Balancing supports health checks on hosts, distributes traffic to virtual servers across multiple isolated locations, known as *Availability Zones*, and dynamically adds or removes virtual servers from the load-balancing rotation. • Auto Scaling supports groups of servers that can grow or shrink on demand. • CloudWatch collects metrics data for your virtual servers, which can be used by Auto Scaling.
A reliable and cost-effective way to route users to your web app	• Amazon Route 53 maps human-readable domain names to IP addresses.

The following diagram shows an example architecture for a web app that employs the services described in the previous table. The web and application tiers run on EC2 instances in public subnets. Access to the EC2 instances over SSH is controlled by a security group, which acts as a firewall. The Auto Scaling group maintains a fleet of EC2 instances that can scale to handle the current load. This Auto Scaling group spans multiple Availability Zones to protect against the potential failure of a single Availability Zone. The load balancer distributes traffic evenly among the EC2 instances. When the Auto Scaling group launches or terminates instances based on load, the load balancer automatically adjusts accordingly. The database tier consists of DB instances in private subnets, including a master and a local slave, located in multiple Availability Zones for failover protection. Access to the DB instances from the EC2 instances is controlled by a security group. Amazon Route 53 provides secure and reliable routing of your domain name to your infrastructure hosted on AWS.



Tutorial

This tutorial walks you through the process of hosting a web app on AWS. We'll use the AWS Management Console to access AWS.

1. Create an Application Server (p. 9)
2. Create a Database Server (p. 12)
3. Deploy an App to Your Application Server (p. 17)
4. Scale and Load-Balance Your Web App (p. 26)
5. Associate a Domain Name with Your Website (p. 30)
6. Clean Up (p. 33)

Alternatively, you can use Elastic Beanstalk to create, load balance, scale, and monitor your servers. For more information, see *Getting Started with AWS: Deploying a Web App* or the *AWS Elastic Beanstalk Developer Guide*.

Pricing

You can use the AWS Simple Monthly Calculator to estimate what it would cost to host your web app on AWS.

Note that if you created your AWS account within the last 12 months, you are eligible for the AWS Free Tier.

For more information about AWS pricing, see Pricing.

Setting Up to Host a Web App on AWS

Before you start this tutorial, complete the following tasks if you haven't already.

Tasks

- Sign Up for AWS (p. 4)
- Create an IAM User (p. 5)
- Create a Key Pair (p. 6)
- Configure a Virtual Private Cloud (VPC) (p. 7)

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS and you can start using them immediately. You are charged only for the services that you use.

If you created your AWS account less than 12 months ago, you can get started with AWS for free. For more information, see AWS Free Tier.

If you have an AWS account already, skip to the next step. If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

1. Open http://aws.amazon.com/, and then choose **Create an AWS Account**.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Create an IAM User

Services in AWS require that you provide credentials when you access them, so that the service can determine whether you have permission to access its resources. The console requires your password. You can create access keys for your AWS account to access the command line interface or API. However, we don't recommend that you access AWS using the credentials for your AWS account; we recommend that you use AWS Identity and Access Management (IAM) instead in order to better protect your AWS resources from unauthorized access.

Create an IAM user, and then add the user to an IAM group with administrative permissions or and grant this user administrative permissions. You can then access AWS using a special URL and the credentials for the IAM user.

If you signed up for AWS but have not created an IAM user for yourself, you can create one using the IAM console.

To create a group for administrators

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Groups**, and then choose **Create New Group**.
3. For **Group Name**, type a name for your group, such as **`Administrators`**, and then choose **Next Step**.
4. In the list of policies, select the check box next to the **AdministratorAccess** policy. You can use the **Filter** menu and the **Search** box to filter the list of policies.
5. Choose **Next Step**, and then choose **Create Group**.

Your new group is listed under **Group Name**.

To create an IAM user for yourself, add the user to the administrators group, and create a password for the user

1. In the navigation pane, choose **Users**, and then choose **Create New Users**.
2. In box **1**, type a user name.
3. Clear the check box next to **Generate an access key for each user**.
4. Choose **Create**.
5. In the list of users, choose the name (not the check box) of the user you just created. You can use the **Search** box to search for the user name.
6. Choose the **Groups** tab and then choose **Add User to Groups**.
7. Select the check box next to the administrators group. Then choose **Add to Groups**.
8. Choose the **Security Credentials** tab. Under **Sign-In Credentials**, choose **Manage Password**.
9. Select **Assign a custom password**. Then type a password in the **Password** and **Confirm Password** boxes. When you are finished, choose **Apply**.

To sign in as this new IAM user, sign out of the AWS console, then use the following URL, where *your_aws_account_id* is your AWS account number without the hyphens (for example, if your AWS account number is `1234-5678-9012`, your AWS account ID is `123456789012`):

```
https://your_aws_account_id.signin.aws.amazon.com/console/
```

Enter the IAM user name and password that you just created. When you're signed in, the navigation bar displays "*your_user_name* @ *your_aws_account_id*".

If you don't want the URL for your sign-in page to contain your AWS account ID, you can create an account alias. From the IAM dashboard, click **Customize** and enter an alias, such as your company name. To sign in after you create an account alias, use the following URL:

```
https://your_account_alias.signin.aws.amazon.com/console/
```

To verify the sign-in link for IAM users for your account, open the IAM console and check under **IAM users sign-in link** on the dashboard.

Create a Key Pair

AWS uses public-key cryptography to secure the login information for your instance. A Linux instance has no password; you use a key pair to log in to your instance securely. You specify the name of the key pair when you launch your instance, then provide the private key when you log in using SSH.

If you haven't created a key pair already, you can create one using the Amazon EC2 console.

To create a key pair

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the navigation bar, in the region selector, click **US West (Oregon)**.
3. In the navigation pane, click **Key Pairs**.
4. Click **Create Key Pair**.
5. Enter a name for the new key pair in the **Key pair name** field of the **Create Key Pair** dialog box, and then click **Create**. Choose a name that is easy for you to remember.
6. The private key file is automatically downloaded by your browser. The base file name is the name you specified as the name of your key pair, and the file name extension is `.pem`. Save the private key file in a safe place.

   Important
   This is the only chance for you to save the private key file. You'll need to provide the name of your key pair when you launch an instance and the corresponding private key each time you connect to the instance.
7. Prepare the private key file. This process depends on the operating system of the computer that you're using.
   - If your computer runs Mac OS X or Linux, use the following command to set the permissions of your private key file so that only you can read it.

     ```
     $ chmod 400 my-key-pair.pem
     ```

   - If your computer runs Windows, use the following steps to convert your .pem file to a .ppk file for use with PuTTY.

     a. Download and install PuTTY from http://www.chiark.greenend.org.uk/~sgtatham/putty/. Be sure to install the entire suite.
     b. Start PuTTYgen (for example, from the **Start** menu, click **All Programs > PuTTY > PuTTYgen**).
     c. Under **Type of key to generate**, select **SSH-2 RSA**.
     d. Click **Load**. By default, PuTTYgen displays only files with the extension `.ppk`. To locate your `.pem` file, select the option to display files of all types.
     e. Select your private key file and then click **Open**. Click **OK** to dismiss the confirmation dialog box.

f. Click **Save private key**. PuTTYgen displays a warning about saving the key without a passphrase. Click **Yes**.
g. Specify the same name that you used for the key pair (for example, *my-key-pair*) and then click **Save**. PuTTY automatically adds the `.ppk` file extension.

Configure a Virtual Private Cloud (VPC)

Amazon VPC enables you to launch AWS resources into a virtual network that you've defined, called a *virtual private cloud (VPC)*. This tutorial requires the use of a VPC. Therefore, we'll check whether you already have a default VPC, and create a VPC otherwise.

To test whether you have a default VPC

1. Open the Amazon VPC console at https://console.aws.amazon.com/vpc/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Your VPCs**.
4. One of the following is true:

   - The list is empty, so you do not have a default VPC.
   - The list has a default VPC (a VPC with a CIDR block of `172.31.0.0/16`).
   - The list has one or more non-default VPCs (a VPC with a CIDR block that is not `172.31.0.0/16`).

If you have a default VPC, you can use it for this tutorial, and you can skip the next procedure. Otherwise, use the following procedure to create a VPC with two public subnets for use with this tutorial.

To create a VPC

1. On the VPC dashboard, click **Start VPC Wizard**.
2. On the **Step 1: Select a VPC Configuration** page, ensure that **VPC with a Single Public Subnet** is selected, and click **Select**.
3. On the **Step 2: VPC with a Single Public Subnet** page, do the following:

   a. In **VPC name**, enter a friendly name for your VPC.
   b. In **Availability Zone**, select the first Availability Zone from the list.
   c. In **Subnet name**, update the name from **`Public subnet`** to `Public subnet 1`.
   d. Leave the other default configuration settings, and click **Create VPC**.
   e. On the confirmation page, click **OK**.

4. In the navigation pane, click **Route Tables**. Find the route table where the **Main** column is **Yes**. This is the main route table. Click the **Name** column for the main route table, enter **`Main`**, and press Enter. Click the **Name** column for the other route table, enter **`Custom`**, and press Enter.
5. Add a second public subnet as follows, so that you'll have two subnets for your application servers. (Note that a default VPC already has a public subnet for each Availability Zone.)

   a. In the navigation pane, click **Subnets**.
   b. Click **Create Subnet**
   c. In **Name tag**, enter the name **`Public subnet 2`**.

d. In **VPC**, select your VPC.
e. In **Availability Zone**, select the second Availability Zone from the list.
f. In **CIDR block**, enter **`10.0.1.0/24`**.
g. Click **Yes, Create**.
h. Select the subnet named `Public subnet 2`, and then select the **Route Table** tab. Click **Edit**, select the route table named `Custom` from **Change to**, and then click **Save**. Note that this step is necessary to make this subnet a public subnet with a route to the Internet.

Next, we need to add private subnets for your database servers to your default VPC or the VPC that you just created.

To add two private subnets to your VPC

1. In the navigation pane, click **Subnets**.
2. Click **Create Subnet**
3. In **Name tag**, enter the name **`Private subnet 1`**.
4. In **VPC**, select your VPC.
5. In **Availability Zone**, select the first Availability Zone from the list.
6. In **CIDR block**, enter **`10.0.2.0/24`**.
7. Click **Yes, Create**.
8. Click **Create Subnet**
9. In **Name tag**, enter the name **`Private subnet 2`**.
10. In **VPC**, select your VPC.
11. In **Availability Zone**, select the second Availability Zone from the list.
12. In **CIDR block**, enter **`10.0.3.0/24`**.
13. Click **Yes, Create**.

For more information about Amazon VPC, see the *Amazon VPC User Guide*.

Step 1: Create an Application Server

You can use Amazon EC2 to create a virtual server to run your web app. These virtual servers are called *EC2 instances*. Typically, you start from a base image called an *Amazon Machine Image (AMI)*.

To create a virtual server using Amazon EC2, complete the following tasks.

Tasks

Create a Security Group for Your Amazon EC2 Instance

A security group acts as a firewall that controls the traffic allowed to reach one or more EC2 instances. When you launch an instance, you can assign it one or more security groups. You add rules to each security group that control the traffic allowed to reach the instances to which the security group is assigned. Note that you can modify the rules for a security group at any time; the new rules take effect immediately.

For this tutorial, we'll create a security group and add the following rules:

- Allow inbound HTTP access from anywhere
- Allow inbound SSH traffic from your computer's public IP address so that you can connect to your instance

To create and configure your security group

1. Decide who requires access to your instance; for example, a single computer or all the computers on a network that you trust. For this tutorial, you can use the public IP address of your computer, which you can get using a service. For example, AWS provides the following service: http://checkip.amazonaws.com. To locate another service that provides your IP address, use the search phrase "what is my IP address".

If you are connecting through an ISP or from behind your firewall without a static IP address, you need to find out the range of IP addresses used by client computers. If you don't know this address range, you can use `0.0.0.0/0` for this tutorial. However, this is unsafe for production environments because it allows everyone to access your instance using **SSH**.

2. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

   Important
   Be sure that you are using the Amazon EC2 console. If you're using the Amazon VPC console instead of the Amazon EC2 console, these directions will not match what you see.

3. In the navigation bar, verify that **US West (Oregon)** is the selected region.
4. In the navigation pane, click **Security Groups**, and then click **Create Security Group**.
5. Enter **`WebServerSG`** as the name of the security group, and provide a description.
6. Select your VPC from the list.
7. On the **Inbound** tab, add the rules as follows:

   a. Click **Add Rule**, and then select **SSH** from the **Type** list. Under **Source**, select **Custom IP** and enter the public IP address range that you decided on in step 1 in the text box.
   b. Click **Add Rule**, and then select **HTTP** from the **Type** list.

8. Click **Create**.

For more information, see Security Groups in the *Amazon EC2 User Guide for Linux Instances*.

Create an IAM Role

All requests to AWS must be cryptographically signed using credentials issued by AWS. Therefore, you need a strategy for managing credentials for software that runs on an EC2 instance. You must distribute, store, and rotate these credentials in a way that keeps them secure but also accessible to the software.

We designed IAM roles so that you can effectively manage AWS credentials for software running on your instances. You create an IAM role and configure it with the permissions that the software requires. For more information about the benefits of this approach, see IAM Roles for Amazon EC2 in the *Amazon EC2 User Guide for Linux Instances* and Roles (Delegation and Federation) in *IAM User Guide*.

The following procedure creates an IAM role that grants the web app full access to AWS. In production, you can restrict the services and resources that a web app can access.

To create an IAM role with full access to AWS

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, click **Roles**, and then click **Create New Role**.
3. On the **Set Role Name** page, enter a name for the role, and then click **Next Step**. Remember this name, as you'll need it when you launch your instance.
4. On the **Select Role** page, under **AWS Service Roles**, select **Amazon EC2**.
5. On the **Attach Policy** page, select the **PowerUserAccess** policy, and then click **Next Step**.
6. Review the role information and then click **Create Role**.

Launch Your EC2 Instance

Now that you've created your key pair, security group, and IAM role, you're ready to launch your instance.

Important
If you created your AWS account less than 12 months ago, and have not already exceeded the free tier benefits for Amazon EC2 and Amazon EBS, this instance will not cost you anything, because we help you select options that are within the free tier benefits. Otherwise, you'll incur the standard Amazon EC2 usage fees for this instance from the time that you launch it until you terminate it, even if it remains idle. The total charges are minimal if you complete the tutorial without interruption and terminate your instance when you are finished. For more information, see Amazon EC2 Pricing.

To launch an EC2 instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Instances**, and then click **Launch Instance**.
4. On the **Choose an Amazon Machine Image** page, click **Free tier only** and then select an Amazon Linux AMI with the HVM virtualization type.
5. On the **Choose an Instance Type** page, the `t2.micro` instance is selected by default. To stay within the free tier, keep this instance type. Click **Next: Configure Instance Details**.
6. On the **Configure Instance Details** page, do the following:
   a. T2 instances must be launched into a subnet. Select your VPC from **Network** and select one of your public subnets from **Subnet**.
   b. Ensure that for **Auto-assign Public IP**, **Enable** is selected in the list. Otherwise, your instance will not get a public IP address or a public DNS name.
   c. Select your IAM role from **IAM role**. Note that you must select an IAM role when you launch an instance; you can't add a role to a running instance.
   d. Click **Review and Launch**. If you are prompted to specify the type of root volume, make your selection and then click **Next**.
7. On the **Review Instance Launch** page, click **Edit security groups**. On the **Configure Security Group** page, click **Select an existing security group**, select the **`WebServerSG`** security group that you created, and then click **Review and Launch**.
8. On the **Review Instance Launch** page, click **Launch**.
9. In the **Select an existing key pair or create a new key pair** dialog box, select **Choose an existing key pair**, then select the key pair you created in Setting Up to Host a Web App on AWS (p. 4). Click the acknowledgment check box, and then click **Launch Instances**.
10. In the navigation pane, click **Instances** to see the status of your instance. Initially, the status of your instance is `pending`. After the status changes to `running`, your instance is ready for use.

Step 2: Create a Database Server

You can use Amazon Relational Database Service (Amazon RDS) to run your database server. In this step, you launch a *Multi-AZ DB instance*. This means that Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone. Updates to your DB instance are synchronously replicated across Availability Zones to the standby in order to keep them in sync and protect your latest database updates against DB instance failure. During certain types of planned maintenance, or in the event of DB instance failure or Availability Zone failure, Amazon RDS automatically fails over to the standby. Because the name record for your DB instance remains the same, your app can resume database operation without the need for manual administrative steps.

Tasks

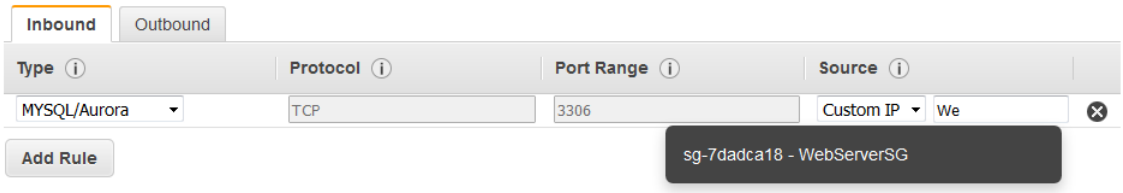
Create a Security Group for Your DB Instance

Create a security group that allows your application server to access your database server.

To create a security group for your DB instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

   Important
   Be sure that you are using the Amazon EC2 console. If you're using the Amazon VPC console instead of the Amazon EC2 console, these directions will not match what you see.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Security Groups**, and then click **Create Security Group**.
4. Enter **`DBServerSG`** as the name of the security group, and provide a description.
5. Select your VPC from the list.
6. On the **Inbound** tab, click **Add Rule**. Select `MYSQL/Aurora` from the **Type** list. Under **Source**, select **Custom IP**, start typing **`WebServerSG`** in the text box, and then select the WebServerSG security group.



7. Click **Create**.

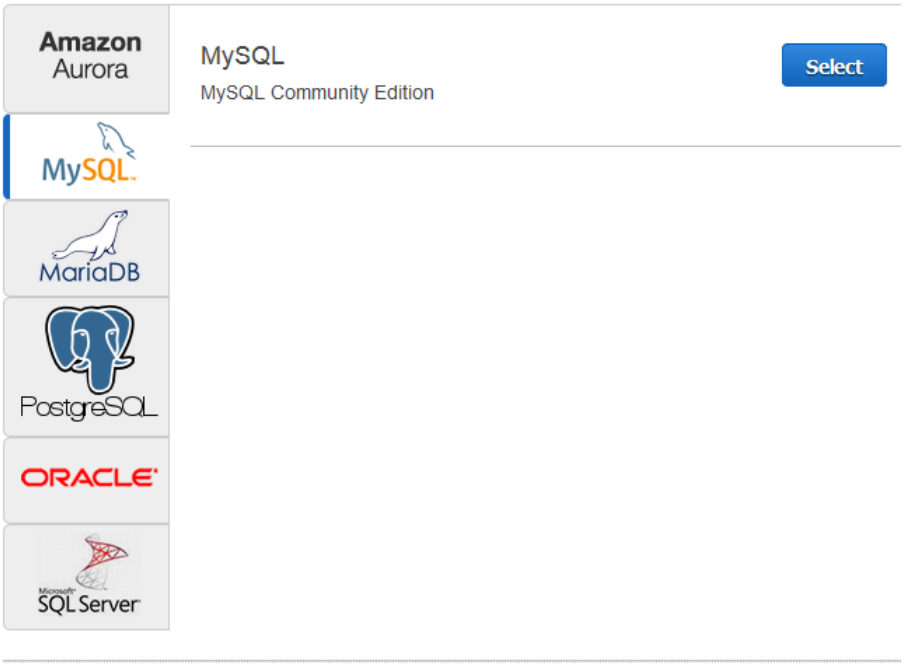
Launch a DB Instance

Now that you've created your security group, you're ready to launch your instance

> **Important**
> You'll incur the standard Amazon RDS usage fees for this instance from the time that you launch it until you terminate it, even if it remains idle. The total charges are minimal if you complete the tutorial without interruption and terminate your DB instance when you are finished. For more information, see Amazon RDS Pricing.

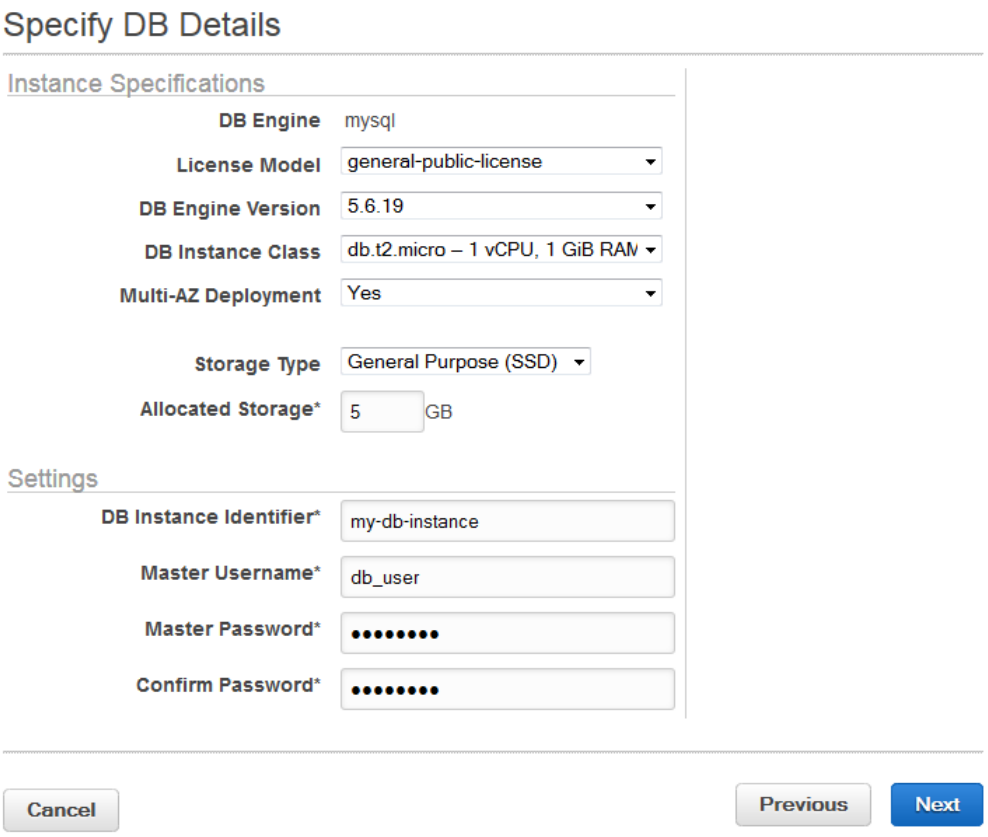
To launch an instance

1. Open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Subnet Groups**, and then click **Create DB Subnet Group**.
4. Complete the **Create DB Subnet Group** page as follows:
   a. Enter a name in **Name**. For example, **`my-db-subnet-group`**.
   b. Enter a description in **Description**.
   c. Select your VPC from **VPC ID**.
   d. In **Availability Zone**, select the first Availability Zone from the list. Select the private subnet from **Subnet ID**, and then click **Add**.
   e. In **Availability Zone**, select the second Availability Zone from the list. Select the private subnet from **Subnet ID**, and then click **Add**.
   f. Click **Create**.
5. In the navigation pane, click **Instances**, and then click **Launch DB Instance**.
6. On the **Select Engine** page, select the **MySQL** tab, and then click **Select**.



7. On the **Do you plan to use this database for production purposes** page, under **Production**, select **MySQL**, and then click **Next Step**.
8. On the **Specify DB Details** page, do the following:

    a. Keep the default license model and the default DB engine version.
    b. Select **`db.t2.micro`** from **DB Instance Class**.
    c. In **Multi-AZ Deployment**, select **Yes**. Although the Multi-AZ deployment is more expensive, it is a best practice.
    d. Select **General Purpose (SSD)** from **Storage Type** and keep the default value for **Allocated Storage**.
    e. In **DB Instance Identifier**, enter **`my-db-instance`**.
    f. In **Master Username**, enter **`db_user`**.
    g. Enter a password in **Master Password** and **Confirm Password**. Record your password in a safe place.
    h. Click **Next Step**.

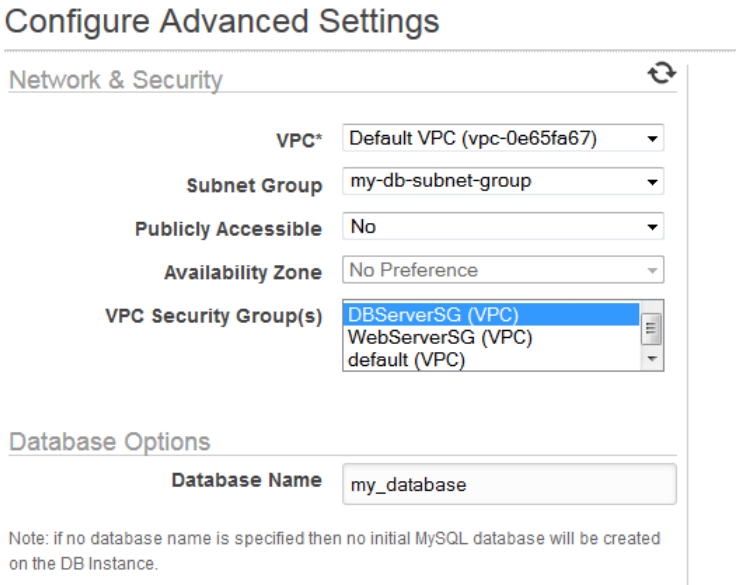


9. On the **Configure Advanced Settings** page, do the following:

   a. Under **Network & Security**, select your VPC from **VPC** and select **DBServerSG** from **VPC Security Groups**.

   b. Under **Database Options**, in **Database Name**, enter **`my_database`**.

   c. Click **Launch DB Instance**.



10. Launching can take a few minutes to complete. When you see the notice that your instance is being created, click **View Your DB Instances**.

Initially, the status of your DB instance is `creating`. After the status changes to `available`, your DB instance ready for use.

Step 3: Deploy Your App

Deploy the app to your EC2 instance by completing the following tasks. For this tutorial, you'll install Drupal and create a test page.

Tasks

- Connect to Your Linux Instance (p. 17)
- Configure the EC2 Instance (p. 19)
- Create a Custom AMI (p. 25)

Connect to Your Linux Instance

After you launch your instance, you can connect to it and use it the way that you'd use a computer sitting in front of you.

Before you connect to your instance, get the public DNS name of the instance using the Amazon EC2 console. Select the instance and locate **Public DNS** on the **Description** tab.

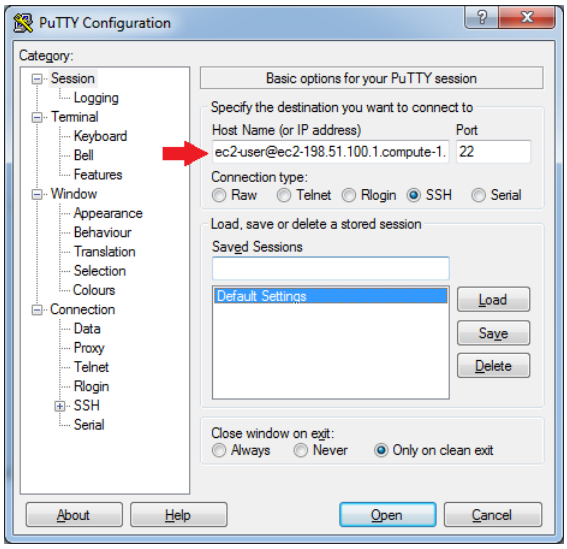
> **Tip**
> If your instance doesn't have a public DNS name, open the VPC console, select the VPC, and check the **Summary** tab. If either **DNS resolution** or **DNS hostnames** is **no**, click **Edit** and change the value to **yes**.

Prerequisites

The tool that you use to connect to your Linux instance depends on the operating system running on your computer. If your computer runs Windows, you'll connect using PuTTY. If your computer runs Linux or Mac OS X, you'll connect using the SSH client. These tools require the use of your key pair. Be sure that you created your key pair as described in Create a Key Pair (p. 6).

To connect to your Linux instance from Windows using PuTTY

1. Start PuTTY (from the **Start** menu, click **All Programs > PuTTY > PuTTY**).
2. In the Category pane, select **Session** and complete the following fields:
   a. In **Host Name**, enter ec2-user@*public_dns_name*.
   b. Ensure that **Port** is 22.

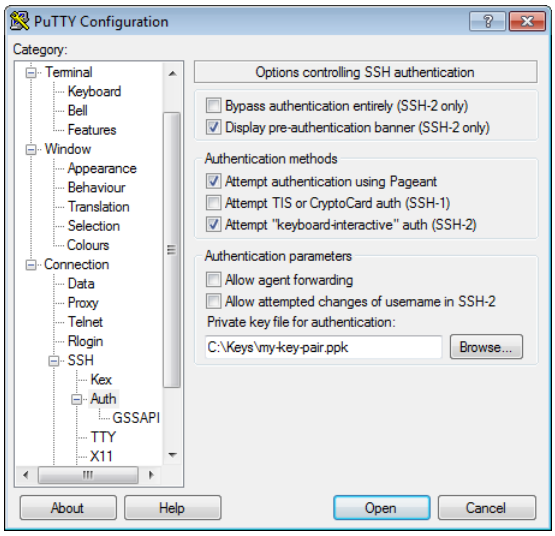


3. In the **Category** pane, under **Connection type**, expand **SSH**, and then select **Auth**. Complete the following:

   a. Click **Browse**.

   b. Select the `.ppk` file that you generated for your key pair, as described in Create a Key Pair (p. 6), and then click **Open**.

   c. Click **Open** to start the PuTTY session.



4. If this is the first time you have connected to this instance, PuTTY displays a security alert dialog box that asks whether you trust the host you are connecting to. Click **Yes**. A window opens and you are connected to your instance.

To connect to your instance from Linux or Mac OS X using SSH

1. Use the **ssh** command to connect to the instance. You'll specify the private key (`.pem`) file and ec2-user@*public_dns_name*.

```
$ ssh -i /path/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazon
aws.com
```

You'll see a response like the following.

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com
(10.254.142.33)'
can't be established.
RSA key fingerprint is
1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

2. Enter `yes`.

   You'll see a response like the following.

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)

to the list of known hosts.
```

Configure the EC2 Instance

To configure the instance, we'll start the web server, install the app, and then configure the app to use our database server. If you don't have the files for your web app, you can use the Drupal application, as we do in this tutorial.

Tasks

Start the Web Server

Use the following procedure to start the web server.

To start the web server

1. To ensure that your software packages are up to date on your instance, use the following command to perform a quick software update.

```
[ec2-user ~]$ sudo yum update -y
```

2. Install the Apache web server, PHP, and MySQL software packages using the following command.

```
[ec2-user ~]$ sudo yum install -y httpd24 php56 mysql55-server php56-mysqlnd
 php56-mbstring php56-gd php56-opcache
```

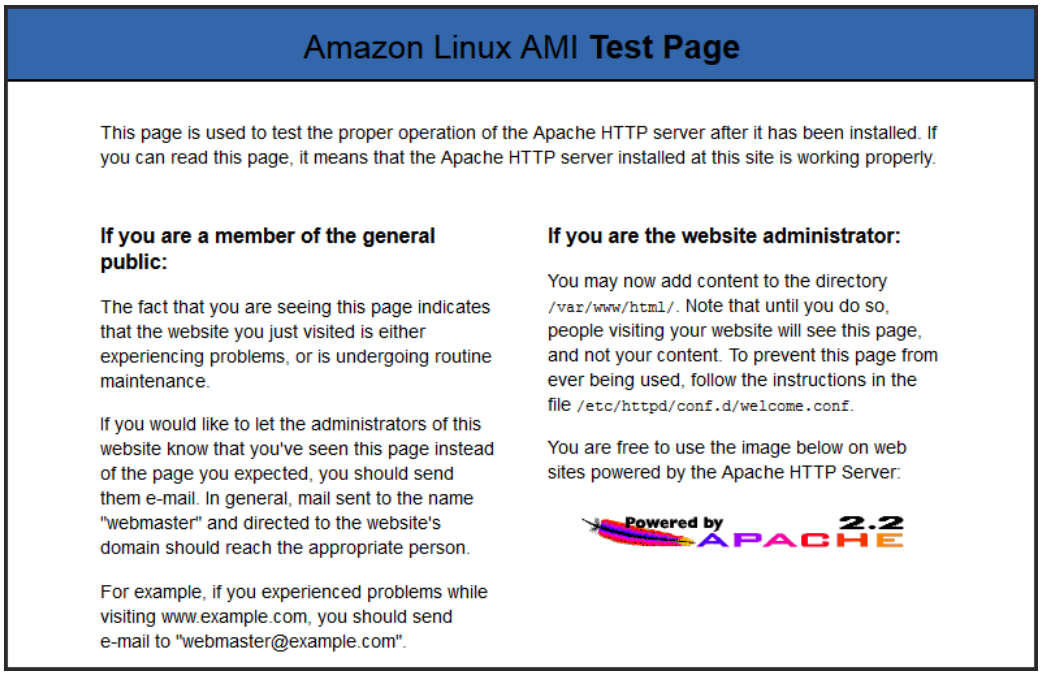
3. Start the Apache web server using the following command.

```
[ec2-user ~]$ sudo service httpd start
```

4. Configure the Apache web server to start at each system boot using the following command.

```
[ec2-user ~]$ sudo chkconfig httpd on
```

5. Before you continue, verify that the web server is running. In a web browser on your computer, paste the public DNS name of your instance into the address bar and press Enter. This displays the Apache test page. If you don't see the test page for Apache, verify that your security group allows HTTP traffic.



Install the App

Now you are ready to install and configure the web app. The procedures depend on the app that you're running. This tutorial installs Drupal to demonstrate how to download files to your instance and configure your app to use your database server.

To install Drupal

1. Go to https://www.drupal.org/project/drupal and note the version of Drupal that you'd like to use.
2. On your instance, use the following command to download Drupal, where *x.y* is the Drupal version you chose.

```
[ec2-user ~]$ wget http://ftp.drupal.org/files/projects/drupal-x.y.tar.gz
```

3. Extract Drupal using the following command, where *x.y* is the Drupal version.

```
[ec2-user ~]$ tar -xzvf drupal-x.y.tar.gz
```

(Optional) You can use the following command to verify that the current directory contains the compressed and uncompressed versions.

```
[ec2-user ~]$ ls
drupal-x.y drupal-x.y.tar.gz
```

(Optional) To remove the compressed version of Drupal, use the following command, where *x.y* is the Drupal version.

```
[ec2-user ~]$ rm drupal-x.y.tar.gz
```

4. Create the `files` directory using the following commands.

```
[ec2-user ~]$ cd drupal-x.y/
[ec2-user drupal-x.y]$ mkdir sites/default/files
```

Copy the `settings.php` file using the following command.

```
[ec2-user drupal-x.y]$ cp sites/default/default.settings.php sites/default/settings.php
```

5. Move Drupal to `/var/www/html/` using the following command.

```
[ec2-user drupal-x.y]$ sudo rsync -avh . /var/www/html/
```

(Optional) To remove the decompressed version of Drupal, you can use the following commands, where *x.y* is the Drupal version.

```
[ec2-user drupal-x.y]$ cd ..
[ec2-user ~]$ rm -rf drupal-x.y/
```

6. Grant ownership of the `/var/www` directory and its contents to the `apache` user using the following command.

```
[ec2-user ~]$ sudo chown -R apache /var/www
```

7. Enable clean URLs, as recommended by Drupal, using the following command.

```
[ec2-user ~]$ sudo vim /etc/httpd/conf/httpd.conf
```

Find the section `<Directory "/var/www/html">`, and set `AllowOverride` as follows:

```
AllowOverride All
```

Save the file (using Esc, :x, Enter), and then restart the Apache web server using the following command.

```
[ec2-user ~]$ sudo service httpd restart
```

Configure Drupal

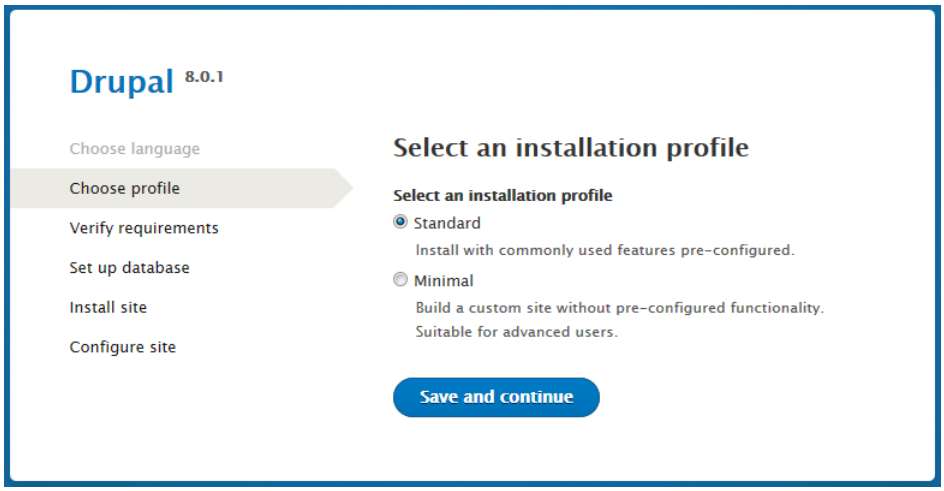
Drupal includes an installation wizard that you can use to configure your website. In the previous procedure, you installed Drupal to the Apache document root, so you can start the installation wizard by opening the website.

Note
This procedure covers Drupal 8. If you are configuring Drupal 7, the steps are similar but there are some differences.

To configure Drupal

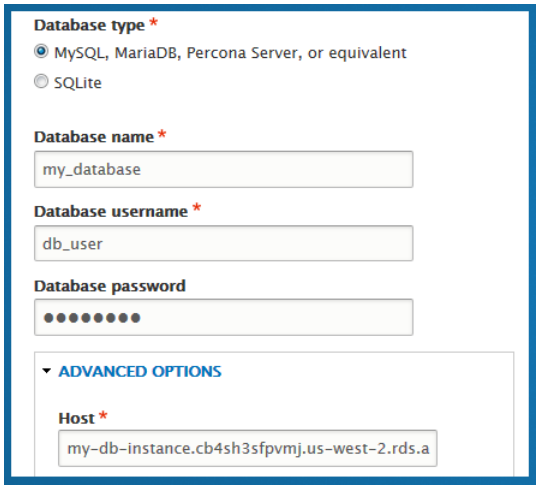
1. Open a web browser on your computer, and enter the public DNS address of your instance in the address bar.
2. On the **Choose language** page, select the language and then click **Save and continue**.

3. On the **Choose profile** page, click **Standard** and then click **Save and continue**.



4. If you see the **Verify requirements** page, address any errors and then continue.
5. On the **Set up database** page, do the following:

a. Select **MySQL, MariaDB, Percona Server, or equivalent** as the database type.
b. In **Database name**, enter the name of the database on your DB instance. In this tutorial, we used the name **`my_database`**.
c. In **Database username**, enter the username for your database. In this tutorial, we used **`db_user`**.
d. In **Database password**,enter the password that you used when you created your DB instance.
e. Expand **ADVANCED OPTIONS**.
f. In **Host**, enter the endpoint of your DB instance. (To find this endpoint, select the DB instance in the Amazon RDS console. Do not include the **`:3306`**.)
g. Click **Save and continue**.

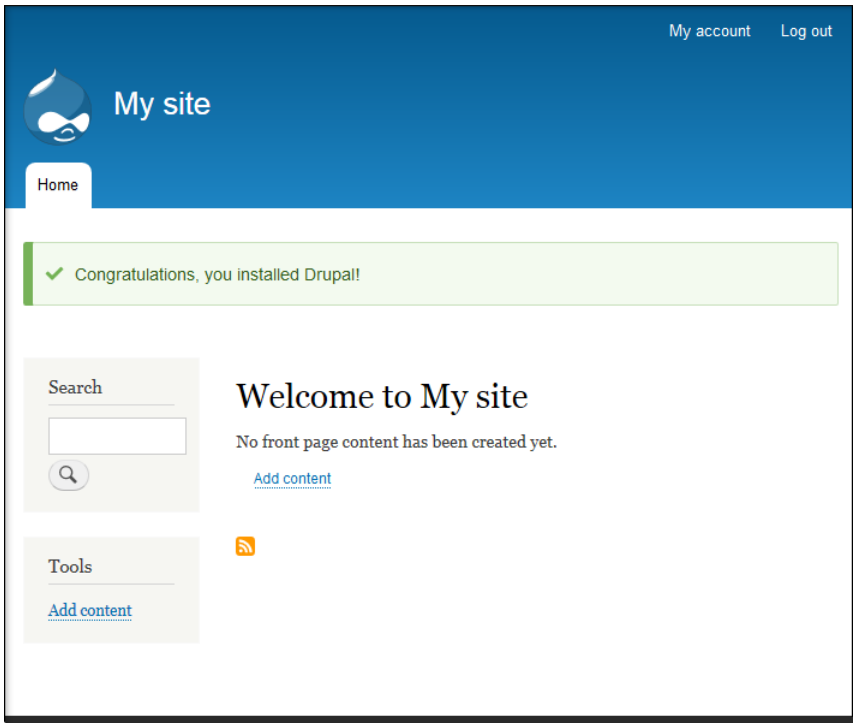


6. After the install completes, the **Configure site** page is displayed. On the **Configure site** page, do the following:

a. In **Site name**, enter a name, such as **`My site`**.
b. In **Site e-mail address**, enter an email address.
c. In **Username**, enter a user name.
d. In **Password**, enter a password. In **Confirm password**, enter the same password.
e. Under **Regional Settings**, select the default country and time zone.
f. Click **Save and continue**.

Test the Website

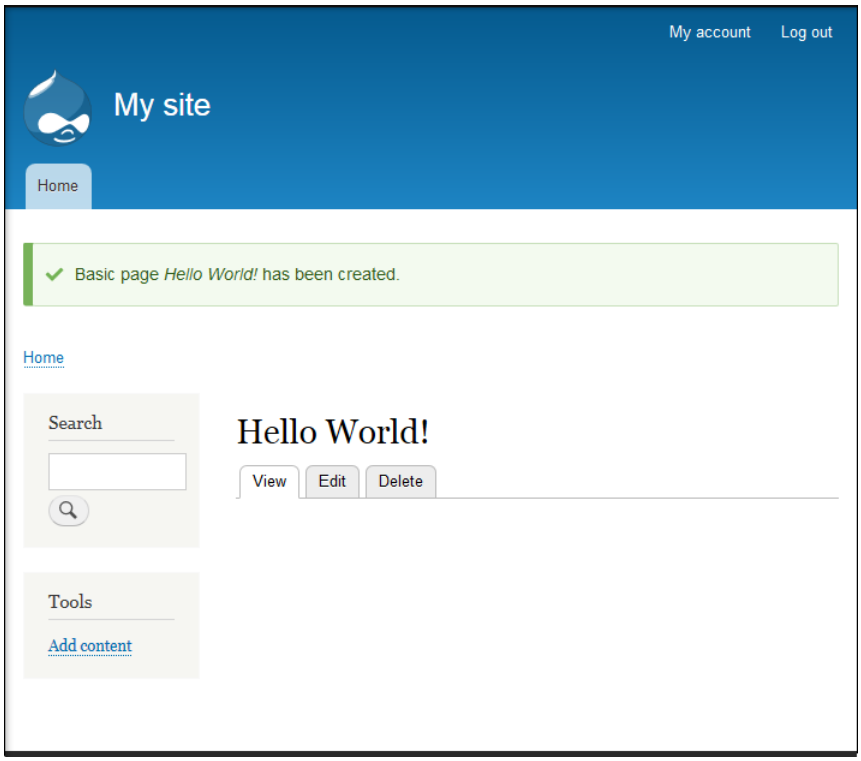
Upon success, your website is displayed.



If you'd like, you can create a front page.

To add a page

1. From your Drupal website, click **Add content** and then click **Basic page**.
2. In **Title**, enter **`Hello World!`**.
3. Expand **Promotion Options** and then select **Promoted to front page**.
4. Click **Save and publish**. Your updated website looks something like this.



Create a Custom AMI

Now that you have customized your EC2 instance, you can create your own AMI. With your own AMI, you can quickly launch a new EC2 instance with the same configuration as this one.

To create an AMI from a running Amazon EBS-backed instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Instances**.
4. On the **Instances** page, select your instance, click **Actions**, select **Image**, and then click **Create Image**.
5. In the **Create Image** dialog box, specify a unique image name and an optional description of the image (up to 255 characters), and then click **Create Image**. Click **Close**.

To view the status of your AMI, go to the **AMIs** page. While the AMI is being created, its status is `pending`. If you go to the **Snapshots** page, you'll see that we created a snapshot that is used to create the root device volume of any instance that you launch using your new custom AMI.

Now that you have a custom AMI, you no longer need the instance that you created it from, because we'll use Auto Scaling to launch new instances in the next step. To terminate the instance, go to the **Instances** page, select it, click **Actions**, select **Instance State**, and then click **Terminate**, and then click **Yes, Terminate**.

Step 4: Scale and Load-Balance Your Web App

In this step you'll configure Auto Scaling and Elastic Load Balancing for your EC2 instances. Auto Scaling is designed to launch or terminate EC2 instances automatically based on the needs of your application. Auto Scaling launches or terminates instances based on the scaling policies that you create. For example, you can set up Auto Scaling to launch an additional instance whenever CPU usage exceeds 60 percent for ten minutes, or you could tell Auto Scaling to terminate half of your instances over the weekend when you expect traffic to be low. You can also monitor your instances to ensure that they are performing optimally and that you always have at least one healthy instance running. For more information, see Auto Scaling.

Elastic Load Balancing is designed to help you improve the availability and scalability of your application. It makes it easy for you to distribute and balance incoming application traffic between two or more EC2 instances. You can dynamically add or remove instances from the load balancer as the capacity requirements of your application change.

As soon as your load balancer becomes available, you're billed for each hour or partial hour that you keep the load balancer running. Note that load balancing is a small cost relative to instance hours.

For more information, see Elastic Load Balancing Pricing. For more information about load balancers, see the Elastic Load Balancing Developer Guide.

Tasks

Configure Auto Scaling and Load Balancing

Now, you'll set up the basic infrastructure for a web app that is load-balanced and auto-scaled with a minimum number of one instance and maximum number of one instance, so that you are only charged for one instance. We configure Auto Scaling to scale out by one whenever there is a change in capacity. We create a CloudWatch alarm to automatically add instances when CPU utilization is high. When you deploy your own web app, you should follow best practices to ensure that your app can run even if it loses one Availability Zone. You can also create additional policies, such as a scale-in policy.

To scale and load-balance your app

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Load Balancers**, and then click **Create Load Balancer**.
4. On the **Define Load Balancer** page, do the following:
   a. Enter a name for the load balancer. For example, **`my-lb`**.
   b. Select your VPC from the **Create LB Inside** list.
   c. If you selected a nondefault VPC, you are prompted to select subnets under **Select Subnets**. Select your two public subnets.
   d. Click **Next: Assign Security Groups**.
5. On the **Assign Security Groups** page, click **Select an existing security group**, and then select your **`WebServerSG`** security group. Click **Next: Configure Security Settings**.
6. We are creating a basic load balancer, so you can click **Next: Configure Health Check** to continue to the next step.
7. On the **Configure Health Check** page, do the following:
   a. Leave **Ping Protocol** and **Ping Port** at their default values.
   b. Set **Ping Path** to **`/`**. This sends queries to your default page, whether it is named `index.html` or something else.
   c. Set **Healthy Threshold** to **`2`**.
   d. Click **Next: Add EC2 Instances**.
8. Click **Next: Add Tags** to skip the **Add Instances to Load Balancer** page. We'll use Auto Scaling to add our EC2 instances to the load balancer.
9. Click **Review and Create** to skip adding tags.
10. Review your settings and then click **Create**. After the load balancer is created, click **Close**.
11. In the navigation pane, click **Launch Configurations**. If you are new to Auto Scaling, you see a welcome page; click **Create Auto Scaling group**.
12. Click **Create a new launch configuration**, and then click **Next Step**.
13. On the **Choose AMI** page, select the **My AMIs** tab, and then select the AMI that you created in Create a Custom AMI (p. 25).
14. On the **Choose Instance Type** page, select the **General purpose** tab, select the `t2.micro` instance type, and then click **Next: Configure details**.
15. On the **Configure details** page, do the following:
    a. Under **Create Launch Configuration**, enter a name for your launch configuration (for example, **`my-lc`**) and select your IAM role from **IAM role**.
    b. Expand **Advanced Details**.
    c. In **User data**, select **As text** and then enter the following script.

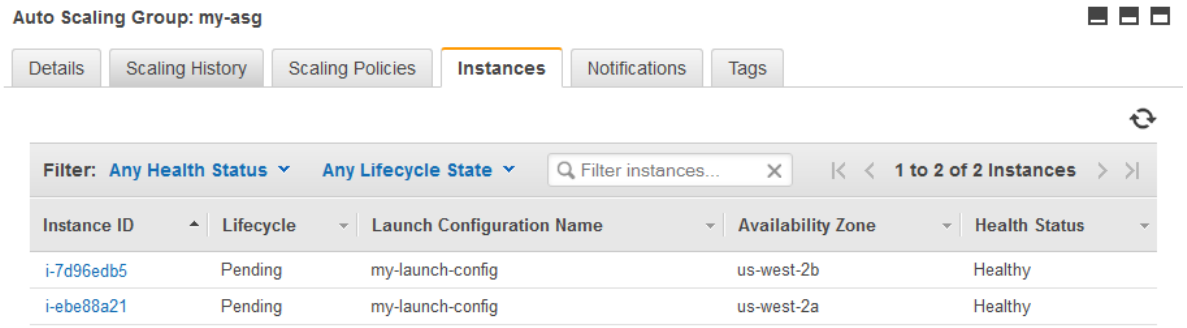
```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

    d. Select **Assign a public IP address to every instance**.
    e. Click **Skip to review**.

16. On the **Review** page, click **Edit security groups**. Click **Select an existing security group**, select the **`WebServerSG`** security group that you created, and then click **Review**.
17. On the **Review** page, click **Create launch configuration**.
18. In the **Select an existing key pair or create a new key pair** dialog box, select **Choose an existing key pair**, then select the key pair you created in Setting Up to Host a Web App on AWS (p. 4). Click the acknowledgment check box, and then click **Create launch configuration**.
19. On the **Configure Auto Scaling group details** page, do the following:
    a. Enter a name for the Auto Scaling group. For example, **`my-asg`**.
    b. In **Group size**, type **`2`** in the text box.
    c. Select your VPC from the **Network** list and your two public subnets from the **Subnet** list. This is a best practice for building fault-tolerant apps. If one Availability Zone experiences an outage, traffic will be routed to the other Availability Zone.
    d. Expand **Advanced Details**. Select **Receive traffic from Elastic Load Balancer(s)**. Select your load balancer from the text field.
    e. Click **Next: Configure scaling policies**.
20. On the **Configure scaling policies** page, do the following:
    a. Select **Use scaling policies to adjust the capacity of this group**.
    b. Under **Increase Group Size**, click **Add new alarm**.
    c. In the **Create Alarm** dialog box, click **create topic**. Enter a name for the SNS topic and an email address for the recipient. Leave the default settings for **Whenever** (Average of CPU Utilization). For **Is**, select **`>`** and specify **`60`** percent. For **For at least**, specify **`2`** consecutive periods of **`5 Minutes`**. Click **Create Alarm**.
    d. Under **Increase Group Size**, in **Take the action**, type **`1`** in the middle box. This means that the alarm adds 1 instance to the Auto Scaling group whenever the CPU utilization threshold that you set is reached.
    e. (Optional) Under **Decrease Group Size**, you can use similar steps to create a policy for scaling down.
    f. Click **Review**.
21. On the **Review** page, click **Create Auto Scaling group**.
22. Click **Close**.

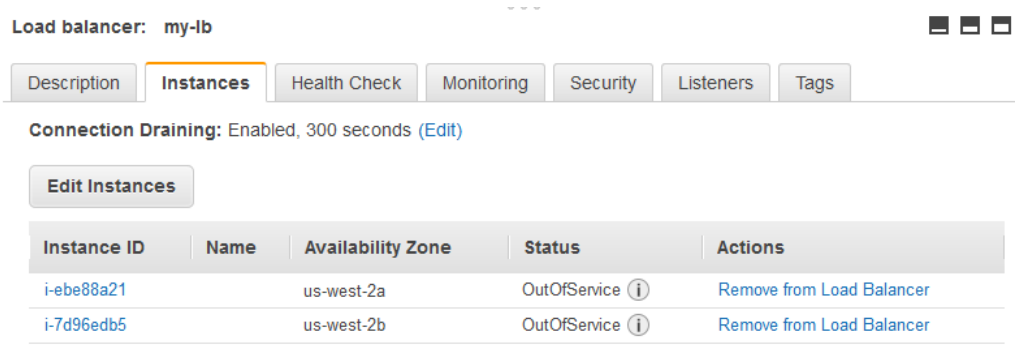
Test Your Load Balancer

First, verify that your instances are ready. From the **Auto Scaling Groups** page, select your Auto Scaling group, and then select the **Instances** tab.



Initially, your instances are in the `Pending` state. When their states are `InService`, they are ready for use.

Next, verify that your instances are registered with the load balancer. From the **Load Balancers** page, select your load balancer, and then select the **Instances** tab.



If the state of your instances is `OutOfService`, it's possible that they are still registering. When their states are `InService`, they are ready for use. After your instances are ready, you can test your load balancer as follows.

To test your load balancer

1. From the **Load Balancers** page, select your load balancer.
2. On the **Description** tab, locate the DNS name. This name has the following form:

```
my-lb-xxxxxxxxxx.us-east-1.elb.amazonaws.com
```

3. In a web browser, paste the DNS name for the load balancer into the address bar and press Enter. You'll see your website displayed.

Step 5: Associate a Domain Name with Your Website Using Amazon Route 53

The easiest way for your customers to access your website is through a memorable domain name. In the procedures on this page, replace "example.com" with your domain name.

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. It is designed as an extremely reliable and cost-effective way to route visitors to websites by translating domain names (such as `www.example.com`) into the numeric IP addresses (such as `192.0.2.1`) that computers use to connect to each other. With Amazon Route 53, you pay only for the domains you configure and the number of queries that the service answers. For more information, see Amazon Route 53.

To associate a domain name with your website, use Amazon Route 53 to complete the following tasks.

Tasks

Register a Domain Name

If you haven't already done so, register your domain name. The Internet Corporation for Assigned Names and Numbers (ICANN) manages domain names on the Internet. You register a domain name using a *domain name registrar*, an ICANN-accredited organization that manages the registry of domain names. The website for your registrar will provide detailed instructions and pricing information for registering your domain name. For more information, see the following resources:

- To use Amazon Route 53 to register a domain name, see Registering Domain Names Using Amazon Route 53 in the *Amazon Route 53 Developer Guide*.
- For a list of accredited registrars, see the Accredited Registrar Directory.

Create a Hosted Zone for Your Domain

A *hosted zone* is a container for the information about how you want to route traffic on the Internet for a domain (such as `example.com`) and its subdomains (such as `www.example.com`).

To create a hosted zone

1. Open the Amazon Route 53 console at https://console.aws.amazon.com/route53/.
2. If you are new to Amazon Route 53, you see a welcome page; click **Get Started Now** under **DNS Management**. Otherwise, click **Hosted Zones** in the navigation pane.
3. Click **Create Hosted Zone**.
4. In **Domain Name**, enter your domain name.
5. Click **Create**.

Create Resource Record Sets for Your Domain and Subdomain

Create an alias resource record set that routes queries for your domain name to your load balancer.

To configure the alias record set for your root domain

1. On the **Hosted Zones** page, select the hosted zone that you created for your domain.
2. Click **Go to Record Sets**.
3. Click **Create Record Set**.
4. Under **Create Record Set**, do the following:
    a. Leave the default name, which is the root domain.
    b. From **Type**, select **A — IPv4 address**.
    c. In **Alias**, click **Yes**. An alias enables Amazon Route 53 to associate your domain name with an AWS resource, such as an Elastic Load Balancing load balancer or an Amazon S3 bucket.
    d. Click **Alias Target**. Select the endpoint for your load balancer from the list.
    e. From **Routing Policy**, select **Simple**.
    f. Leave **Evaluate Target Health** set to **No**.
    g. Click **Create**.

Set Up a DNS Provider

If you registered a new domain name and have used that name while doing this tutorial, you're ready to set up Amazon Route 53 as your DNS provider.

Alternatively, if you're reusing a domain name that was previously associated with another website, you might need to transfer other DNS records from your current DNS provider to Amazon Route 53 in order to ensure the continued availability of the services hosted under the domain name. To determine which DNS records you must replicate in Amazon Route 53, check the DNS record settings configured for the domain in your current DNS provider. Two records that you should not transfer to Amazon Route 53 are the Start of Authority (SOA) and Name Server (NS) records. These records were set by Amazon Route 53 when the name servers were allocated, and they should not be changed.

First, log into the domain name registrar that you used to register your domain name. Use the web interface provided by the registrar to set the name servers for your domain to the name server values displayed under **Name Servers** in the details for the hosted zone. How you do this depends on the registrar that you used.

Wait between two to 48 hours for the Internet DNS resolver network to propagate name server changes. To see if the name server change has gone through, use a command line utility such as dig (for Mac OS X, Unix, or Linux) or nslookup (for Windows). The following example shows how use dig to see which name servers are associated with your domain.

```
dig example.com
```

When the AUTHORITY SECTION of the output shows the AWS name servers that you allocated using Amazon Route 53, the DNS changes have propagated through the DNS resolver network.

```
;; AUTHORITY SECTION:
example.com. 118928 IN NS ns-806.awsdns-36.net.
example.com. 118928 IN NS ns-1456.awsdns-54.org.
example.com. 118928 IN NS ns-1713.awsdns-22.co.uk.
example.com. 118928 IN NS ns-105.awsdns-13.com.
```

After your DNS changes have propagated, you'll be able to view your website using your custom domain name.

If you open your www subdomain (such as `www.example.com`) in your web browser, it redirects to your domain (such as `example.com`).

Step 6: Clean Up

After completing this tutorial, be sure to delete the AWS resources that you created so that you no longer accrue charges.

Tasks

Delete the Amazon Route 53 Hosted Zone

Before you delete the hosted zone, you must delete the record sets that you created. You do not need to delete the NS and SOA records; these are automatically deleted when you delete the hosted zone.

To delete the record sets

1. Open the Amazon Route 53 console at https://console.aws.amazon.com/route53/.
2. In the list of hosted zones, click the name of your hosted zone.
3. In the list of record sets, select the check boxes that correspond to the A records that you created. The type of each record set is listed in the **Type** column.
4. Click **Delete Record Set**.
5. When prompted for confirmation, click **Confirm**.

To delete an Amazon Route 53 hosted zone

1. Continuing from the previous procedure, click **Back to Hosted Zones**.
2. Select the check box that corresponds to your hosted zone, and then click **Delete Hosted Zone**.
3. When prompted for confirmation, click **Confirm**.

Delete the Auto Scaling Group

If you decide that you no longer need an Auto Scaling group, you can delete it. Deleting an Auto Scaling group using the console terminates all EC2 instances in the Auto Scaling group and any CloudWatch alarms for your scaling policies. After you delete the Auto Scaling group, you can delete the launch configuration.

To delete the Auto Scaling group

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Auto Scaling Groups**.
4. Select your Auto Scaling group, click **Actions**, and then click **Delete**.
5. When prompted for confirmation, click **Yes, Delete**. The values of the **Desired**, **Min**, and **Max** columns change to 0 and the EC2 instances enter the `terminating` state.

To delete the launch configuration

1. In the navigation pane, click **Launch Configurations**.
2. Select your launch configuration, click **Actions**, and then click **Delete launch configuration**.
3. When prompted for confirmation, click **Yes, Delete**.

Delete the Load Balancer

If you decide that you no longer need the load balancer that you created in the US West (Oregon) region earlier, you can delete it.

> **Important**
> The EC2 instances associated with the load balancer continue to run after you delete the load balancer. You continue to accrue charges for the instances while they are running.

To delete your load balancer

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Load Balancers**.
4. Select the load balancer, click **Actions**, and then click **Delete**.
5. When prompted for confirmation, click **Yes, Delete**.

Delete Your Custom AMI

If you decide that you no longer need an AMI, you can deregister it. After you deregister an AMI, you can't use it to launch new instances. However, it doesn't affect any instances that you already launched from the AMI. It also doesn't affect the snapshot that was created when you created the AMI. You'll continue to accrue charges for the snapshot until you delete it.

To deregister an AMI and delete the associated snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.

3. In the navigation pane, click **AMIs**.
4. Select the AMI, click **Actions**, and then click **Deregister**. When prompted for confirmation, click **Continue**.
5. In the navigation pane, click **Snapshots**.
6. Select the snapshot, click **Actions**, and then click **Delete**. When prompted for confirmation, click **Yes, Delete**.

Terminate the DB Instance

If you decide that you no longer need a DB instance, you can terminate it. As soon as the status of the DB instance changes to `deleted`, you stop accruing charges for the DB instance.

To terminate your DB instance

1. Open the Amazon RDS console at https://console.aws.amazon.com/rds/.
2. In the navigation bar, verify that **US West (Oregon)** is the selected region.
3. In the navigation pane, click **Instances**.
4. Select the DB instance, click **Instance Actions**, and then click **Delete**.
5. On the **Delete DB Instance** dialog box, select **No** from the **Create final snapshot?** list, and then click **Yes, Delete**.

   If this were a production database, we would recommend that you create a final snapshot so that you could restore the DB instance later if needed.

Related Resources

The following table lists some of the AWS resources that you'll find useful as you work with AWS.

Resource	Description
AWS Products & Services	Information about the products and services that AWS offers.
AWS Documentation	Official documentation for each AWS product, including service introductions, service features, and API reference.
AWS Discussion Forums	Community-based forums for discussing technical questions about Amazon Web Services.
Contact Us	A central contact point for account questions such as billing, events, and abuse. For technical questions, use the forums.
AWS Support Center	The hub for creating and managing your AWS Support cases. Also includes links to other helpful resources, such as forums, technical FAQs, service health status, and AWS Trusted Advisor.
AWS Support	The home page for AWS Support, a one-on-one, fast-response support channel to help you build and run applications in the cloud.
AWS Architecture Center	Provides the necessary guidance and best practices to build highly scalable and reliable applications in the AWS cloud. These resources help you understand the AWS platform, its services and features. They also provide architectural guidance for design and implementation of systems that run on the AWS infrastructure.
AWS Security Center	Provides information about security features and resources.
AWS Economics Center	Provides access to information, tools, and resources to compare the costs of Amazon Web Services with IT infrastructure alternatives.
AWS Technical Whitepapers	Provides technical whitepapers that cover topics such as architecture, security, and economics. These whitepapers have been written by the Amazon team, customers, and solution providers.

Resource	Description
AWS Blogs	Provides blog posts that cover new services and updates to existing services.
AWS Podcast	Provides podcasts that cover new services, existing services, and tips.