

How to set up a WordPress Website on Amazon Web Services (AWS)

SCREENSHOT GUIDE



The screenshot displays the AWS Management Console interface with various services categorized into several groups:

- Compute & Networking:** Direct Connect, EC2, Route 53, VPC.
- Storage & Content Delivery:** CloudFront, Glacier, S3, Storage Gateway.
- Database:** DynamoDB, ElastiCache, RDS, Redshift.
- Deployment & Management:** CloudFormation, CloudTrail, CloudWatch, Elastic Beanstalk, IAM, OpsWorks, AWS CloudFormation, AWS CloudFormation, AWS CloudFormation, AWS CloudFormation.
- Analytics:** Data Pipeline, Elastic MapReduce.
- App Services:** AppStream, CloudSearch, Elastic Transcoder, SES, SQS, SWF.
- Applications:** WorkSpaces, Zocalo.
- Mobile Services:** Cognito, Mobile Analytics, SNS.

A large WordPress logo is overlaid on the center of the screenshot, and the text "WORDPRESS" is written in a large, bold, blue font across the middle. The Amazon Web Services logo is visible in the bottom right corner of the screenshot.

Table of Contents

1	Abstract.....	3
2	Overview.....	3
3	Pricing.....	3
4	Create an Amazon Web Services (AWS) Account.....	4
5	Overview of AWS Management Console.....	10
5.1	Shortcut Bar.....	10
5.2	Categories and Services.....	10
5.3	Sidebar.....	11
6	Launch an Instance using Amazon EC2.....	11
6.1	Linux Distributions vs Windows.....	11
6.2	Launch an Instance.....	12
6.3	Assign an Elastic IP.....	19
7	Access an Instance.....	21
7.1	Convert .PEM Key to .PPK PuTTY Key using PUTTYgen.....	21
7.2	Connect to an Instance using PuTTY.....	23
8	Buy a Domain Name using Amazon Route 53.....	27
8.1	Domain WHOIS Record Privacy Protection.....	27
8.2	Register a Domain.....	27
9	Set up DNS using Amazon Route 53.....	32
10	Configure an Instance.....	35
10.1	Update the Software on an Instance.....	35
10.2	Updating GRUB on an Instance.....	37
10.3	Reboot an Instance.....	39
10.4	Install MySQL Database.....	40
10.5	Install PHP and Apache HTTP Server.....	42
11	Transfer Files to an Instance.....	45
11.1	Connect to Instance using FileZilla.....	45
11.2	Transfer WordPress Files to an Instance.....	49
11.3	Extract WordPress Files to an Instance.....	50
12	Configure Apache Web Server.....	53
13	Set up MySQL Database using Adminer.....	58
13.1	Download and Set up Adminer.....	58
13.2	Create Database and Database User for WordPress.....	59

14	Configure WordPress.....	62
14.1	Log in and Configure WordPress using the Admin Dashboard.....	65
15	Final Notes.....	68
15.1	Final Permissions.....	68

1 Abstract

This end-to-end procedure is intended for anyone that is setting up a WordPress website on Amazon Web Services (AWS). It provides information on how to create an account on Amazon Web Services, select and purchase a domain name, configure DNS through Route 53, launch an instance through Elastic Cloud Compute (EC2), update and secure the instance, install all the required software, and set up a WordPress website.

The guide is targeted at system administrators, web developers, and those who are interested in learning something new. It is beneficial to have prior Linux experience, but it's certainly not required.

The guide includes images as well as explicit instructions for those using the text-only version. Amazon and WordPress constantly add new features and update existing components so the information in the guide may have changed since publication.

At the time of this release, AWS is running the following versions:

- Elastic Computer Cloud: July 30, 2014 5:00 PM GMT
- Route 53: July 31, 2014 7:00 PM GMT

2 Overview

WordPress is one of the most popular content management systems (CMS) on the market. It's used in industries around the world and provides a stable platform for companies and individuals to build their web presence. There are many different domain registrars, DNS providers, and hosting providers from which to choose, but for this guide, you'll see how to utilize AWS for all your web needs. AWS is an industry leading, highly scalable collection of services in the cloud.

3 Pricing

WordPress is free. It is open-source software which means all the code is available for you to view. The benefit to open source software is developers everywhere can add new features, fix bugs, and improve security. Even you can contribute to the code that everyone uses. The downside is hackers can find vulnerabilities much quicker when they can view the code in open source software than they can with proprietary software. WordPress has been around for a long time so you can rest assured that there are very few vulnerabilities. If there are any, they will be fixed ASAP since WordPress is one of the most popular content management systems in the world. There are themes and plugins for WordPress that cost money, but you are not required to use any of them since there are plenty of free themes and plugins.

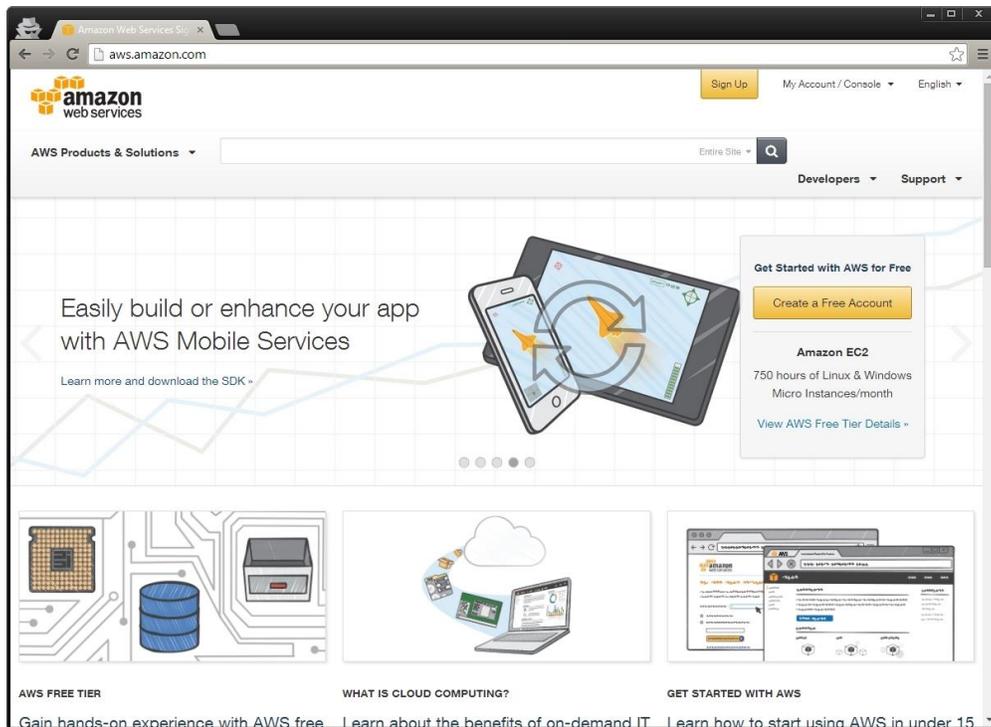
The Amazon account is free, but many of the services pricing is based on usage. If you don't use any of the services, then it won't cost any money. If you do use a service for only a few minutes, you'll only be charged for that usage. Amazon is very transparent with the costs. Pricing is available for each of the services on their website. A quick Google search for *EC2 Pricing* will bring up the information you need.

If you've never used AWS before, you are eligible for the **AWS Free Tier** which is explained at <http://aws.amazon.com/free/>. You are able to use a few of the AWS services like EC2 and S3 for free for a year which is perfect when evaluating whether the services fit your needs.

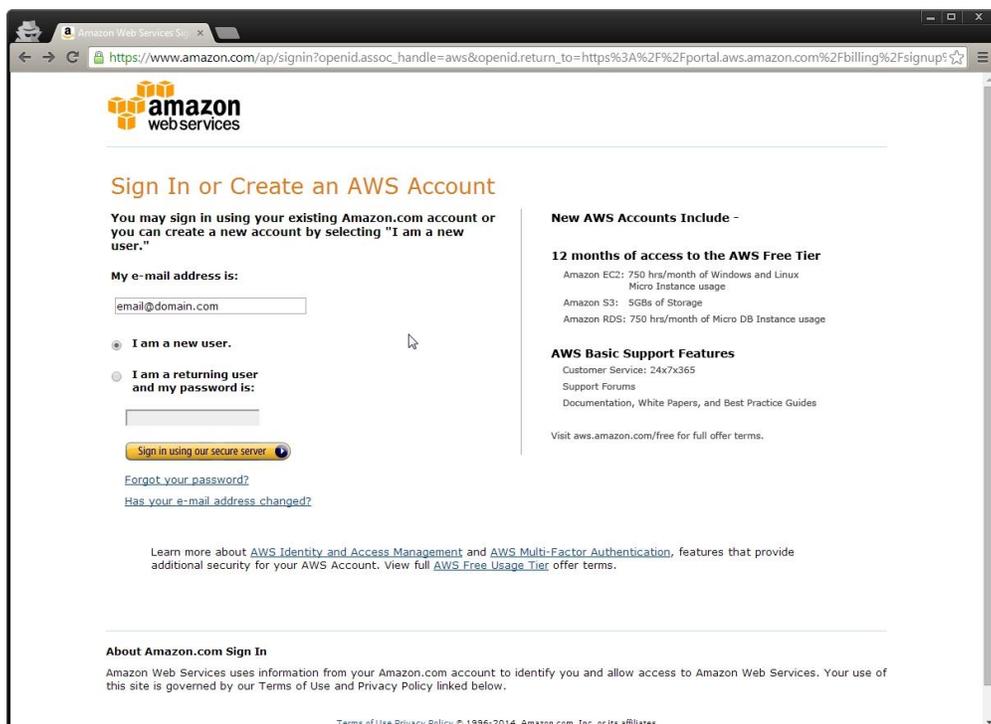
4 Create an Amazon Web Services (AWS) Account

In order to use AWS, you need an Amazon account. If you never purchased from Amazon.com, you'll have to create an account. If you have purchased from Amazon.com, you can use your existing account to access AWS.

Open your web browser to <http://aws.amazon.com> and click on **Sign Up**.



If you already have an Amazon account, you can type in your email address and password to log in. If you don't have an Amazon account, type in your email address, select **I am a new user** and click **Sign in using our secure server**.



On the **Login Credentials** page, fill in all the name, email, and password textboxes and click **Continue**.

The screenshot shows the 'Login Credentials' page on the Amazon Web Services website. The page title is 'Login Credentials' and it includes the Amazon Web Services logo. Below the title, there is a sub-header 'Login Credentials' and a brief instruction: 'Use the form below to create login credentials that can be used for AWS as well as Amazon.com.' The form contains several input fields: 'My name is:' with the value 'John Doe', 'My e-mail address is:' with 'email@domain.com', and 'Type it again:' with 'email@domain.com'. A note below these fields states: 'note: this is the e-mail address that we will use to contact you about your account'. There are two password fields: 'Enter a new password:' and 'Type it again:', both containing masked characters (dots). A yellow 'Continue' button is located below the password fields. At the bottom of the page, there is a section titled 'About Amazon.com Sign In' with a paragraph of text and links for 'Terms of Use' and 'Privacy Policy'. The footer includes the copyright notice '© 1996-2014, Amazon.com, Inc. or its affiliates' and 'An amazon.com company'.

On the **Contact Information** page, fill in your name, address, phone number, and security check textboxes.

The screenshot shows the 'Contact Information' page on the Amazon Web Services website. The page title is 'Contact Information' and it includes the Amazon Web Services logo. Below the title, there is a sub-header 'Contact Information' and a brief instruction: 'Use the form below to provide your contact information.' The form contains several input fields: 'Full Name*' with 'John Doe', 'Company Name' with 'Company', 'Country*' with a dropdown menu set to 'United States', 'Address*' with '123 Sesame Street' and a sub-field for 'Apartment, suite, unit, building, floor, etc.', 'City*' with 'City', 'State / Province or Region*' with 'State', 'Postal Code*' with '12345', and 'Phone Number*' with '123-555-6789'. There is a 'Security Check' section with a CAPTCHA image showing the characters '7TNHPM' and a 'Refresh Image' link. Below the CAPTCHA, there is a text input field for the characters, containing '7TNHPM'. The page also features a language selector set to 'English' and a 'Sign Up' button.

Scroll to the bottom of the page and read the **AWS Customer Agreement**. Click **Create Account and Continue** when you are ready to proceed.

The screenshot shows the AWS Console Sign-up page. The browser address bar displays the URL: https://portal.aws.amazon.com/billing/signup?nc1=h_ct&redirect_url=https%3A%2F%2Faws.amazon.com%2Fregistration-confirmation&new_acco. The form includes the following fields and elements:

- City***: Text input with "City" entered.
- State / Province or Region***: Text input with "State" entered.
- Postal Code***: Text input with "12345" entered.
- Phone Number***: Text input with "123-555-6789" entered.
- Security Check**: A CAPTCHA image showing the characters "7TNHPM". Below it is a "Refresh Image" link and a text input field containing "7TNHPM".
- AWS Customer Agreement**: A checkbox is checked, with the text "Check here to indicate that you have read and agree to the terms of the AWS Customer Agreement".
- Create Account and Continue**: A prominent yellow button.

At the bottom of the page, there are links for "Privacy Policy" and "Terms of Use", a copyright notice "© 2014 Amazon Web Services, Inc. or its affiliates. All rights reserved.", and the text "An amazon.com company".

On the **Payment Information** page, fill in your credit card information and billing address. Click **Continue**.

The screenshot shows the AWS Console Sign-up page at the "Payment Information" step. The browser address bar displays the same URL as the previous screenshot. The page features the Amazon Web Services logo and the text "Amazon Web Services Sign Up". A progress bar at the top indicates the current step: "Contact Information" (completed), "Payment Information" (current), "Identity Verification", "Support Plan", and "Confirmation".

The "Payment Information" section includes the following content:

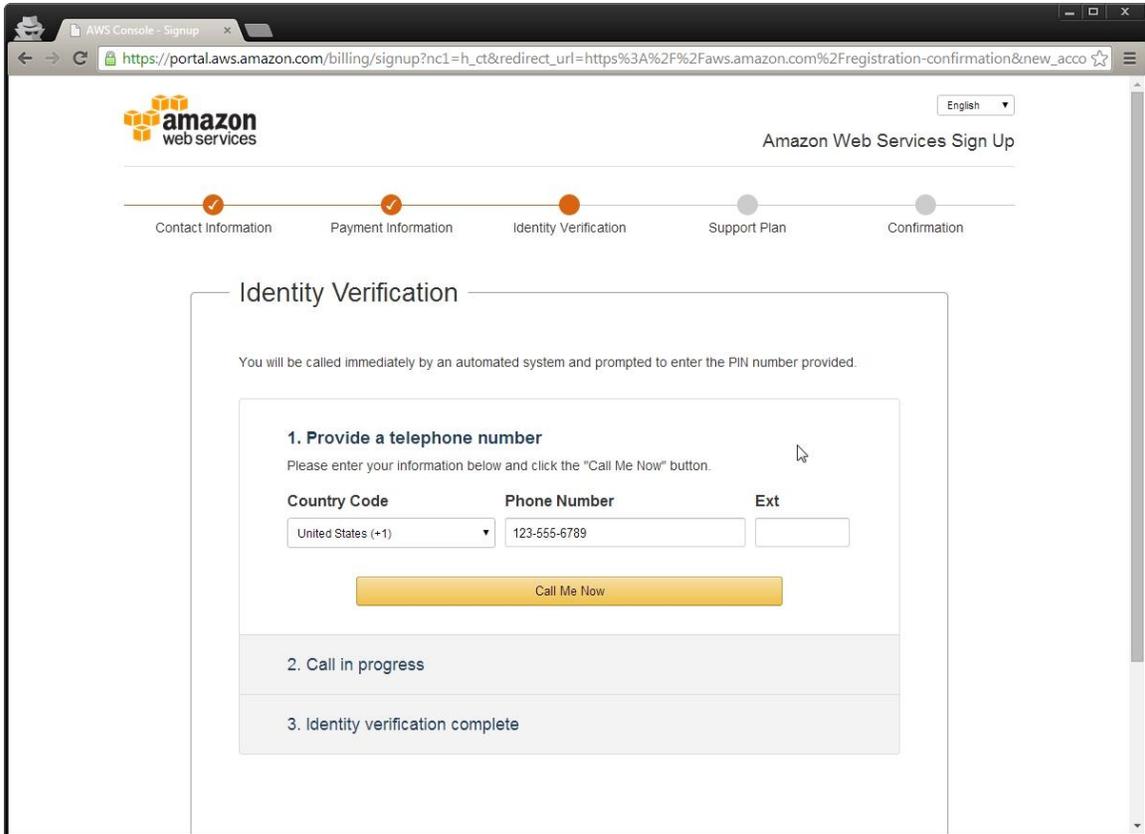
- A message: "Please enter your payment information below. You will be able to try a broad set of AWS products for free via the Free Usage Tier. We will only bill your credit card for usage that is not covered by our Free Usage Tier."
- A table showing the AWS Free Usage Tier benefits:

AWS Free Usage Tier	Compute Amazon EC2	Storage Amazon S3	Database Amazon RDS
free for 1 year	750hrs/month*	5GB	750hrs/month*

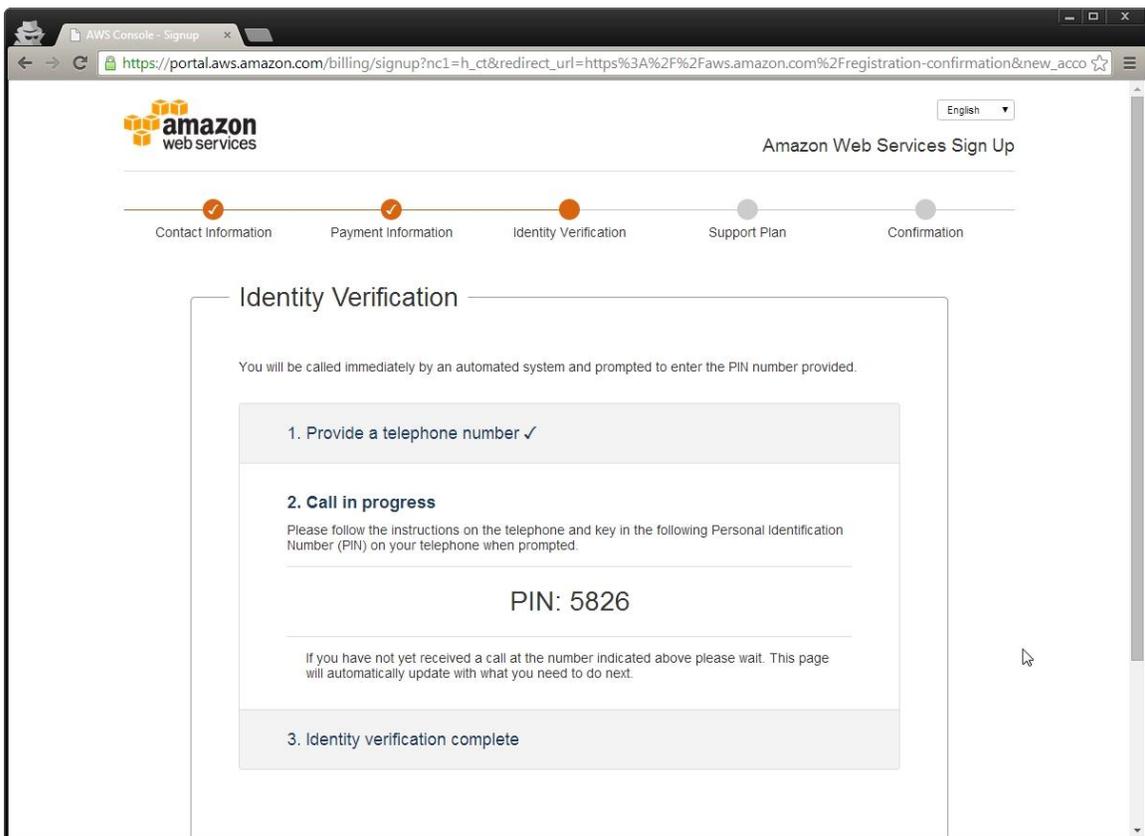
*View full offer details >

- Credit Card Number**: Text input with "XXXXXXXXXXXXXXXXXXXX".
- Expiration Date**: Two dropdown menus showing "01" and "2014".
- Cardholder's Name**: Text input with "John Doe".
- Choose Your Billing Address**: A section with the text "Select the address associated with your credit card." and two radio buttons: "Use my contact address" (selected) and "Use a new address".

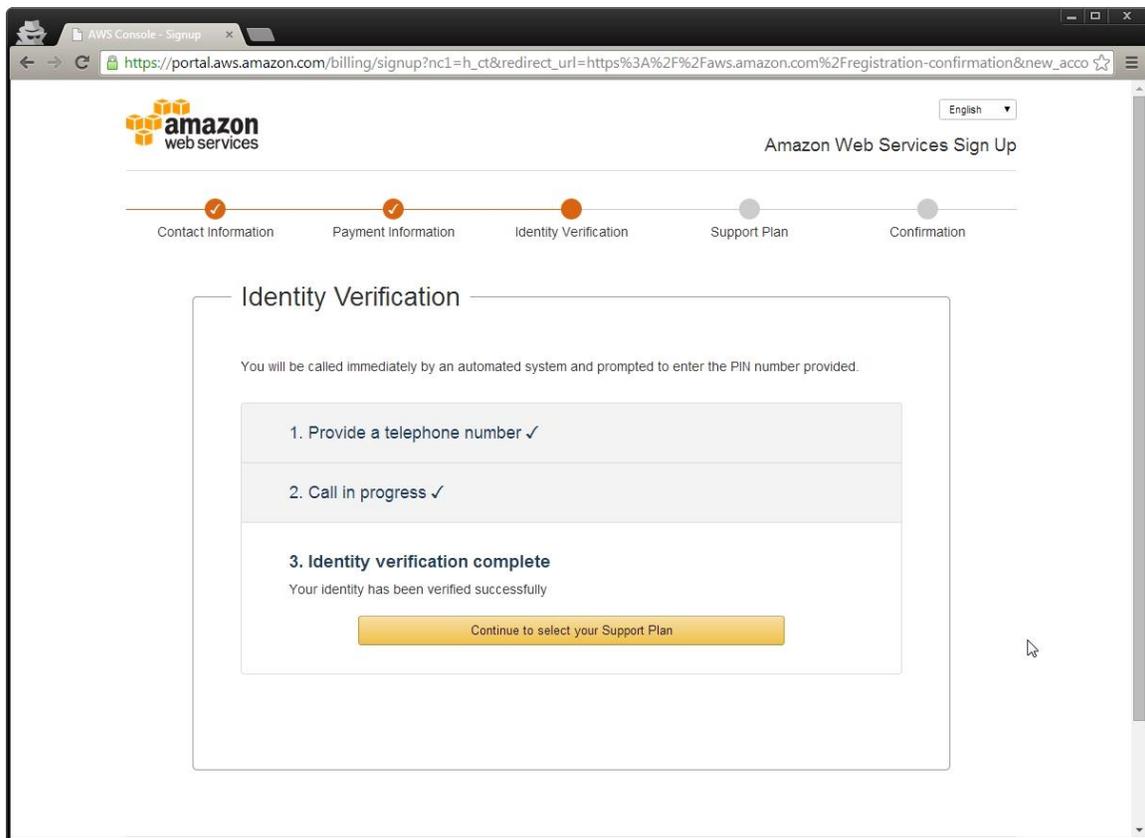
On the **Identify Verification** page, fill in your telephone number and click **Call Me Now**. You will receive an automated telephone call.



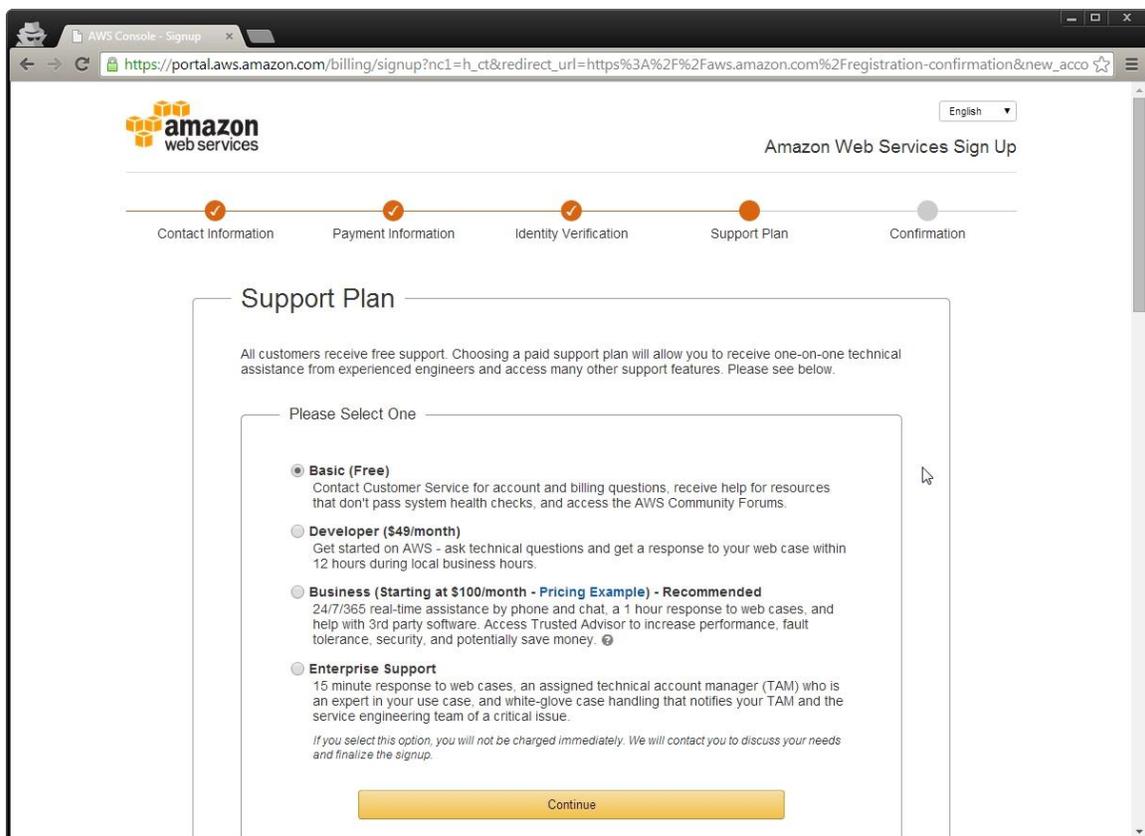
Once you answer the call, the automated voice will tell you to enter in your pin.



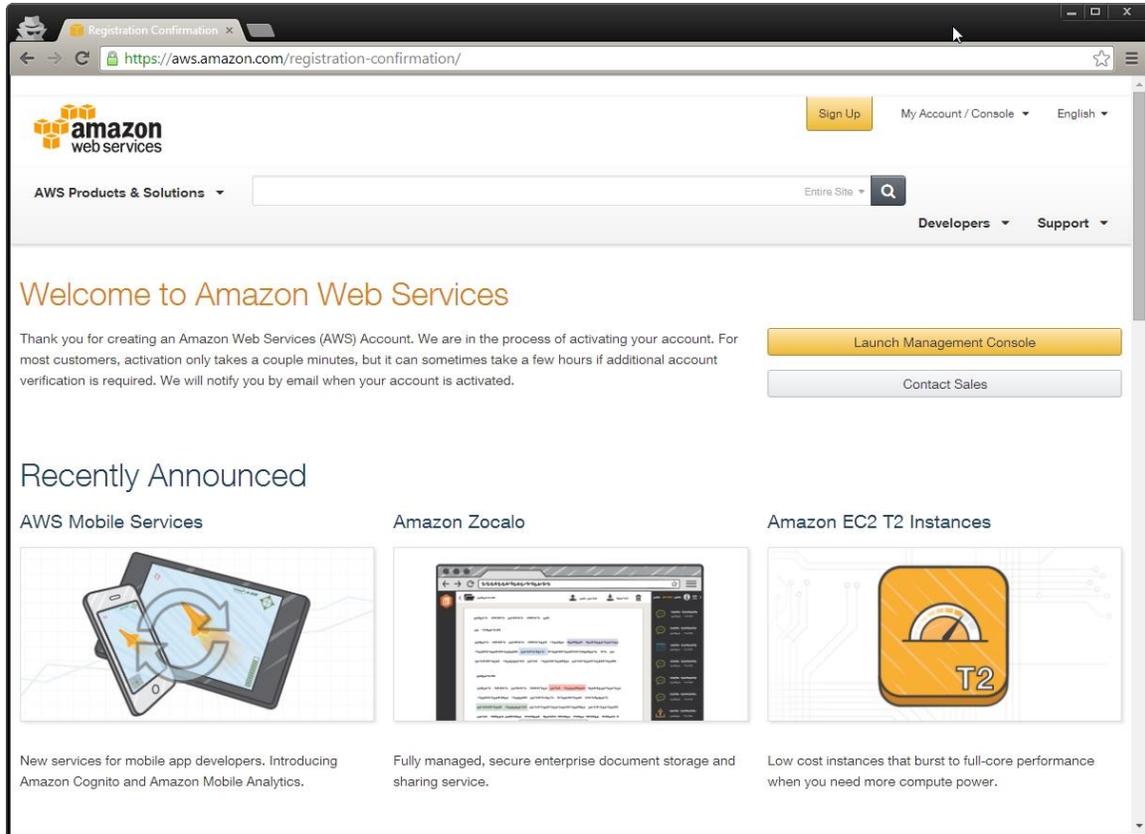
Once you finish with the call, click **Continue** to select your **Support Plan**.



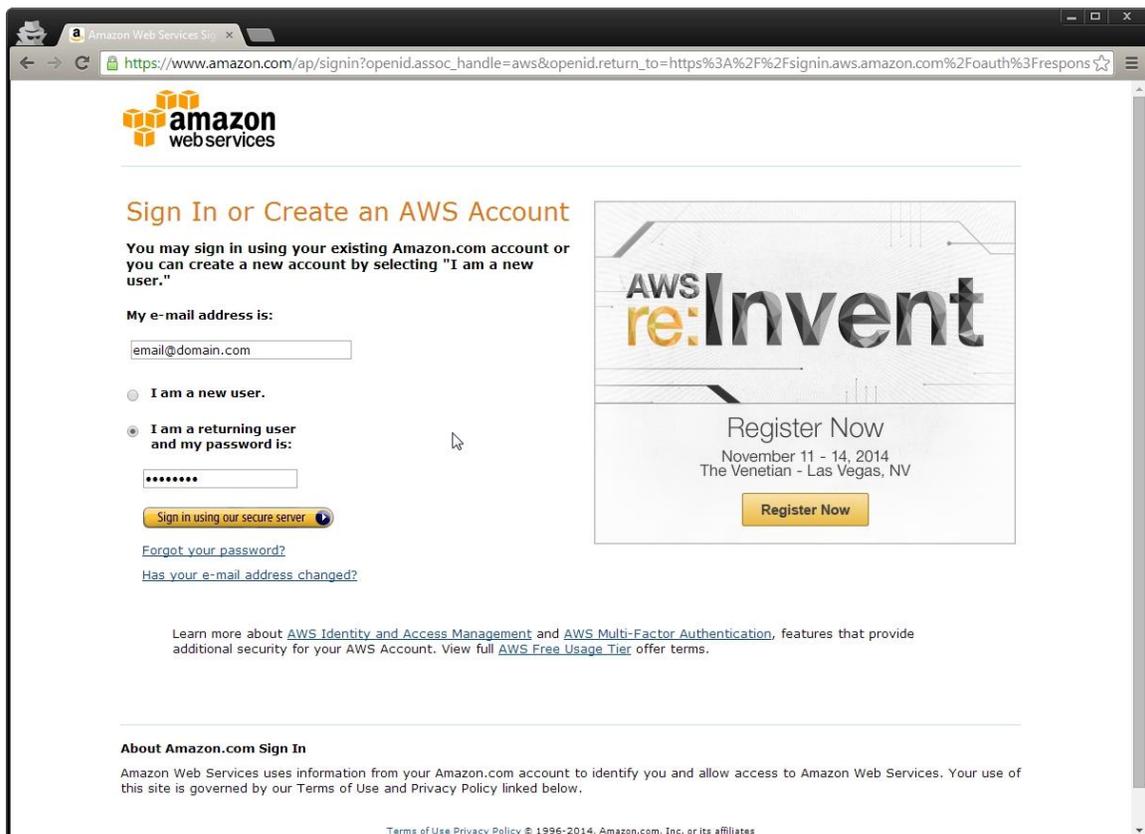
On the **Support Plan** page, select the type of support you would like to use. Click **Continue**.



You should see the **Welcome to Amazon Web Services** page once you finish registration. Click **Launch Management Console**.

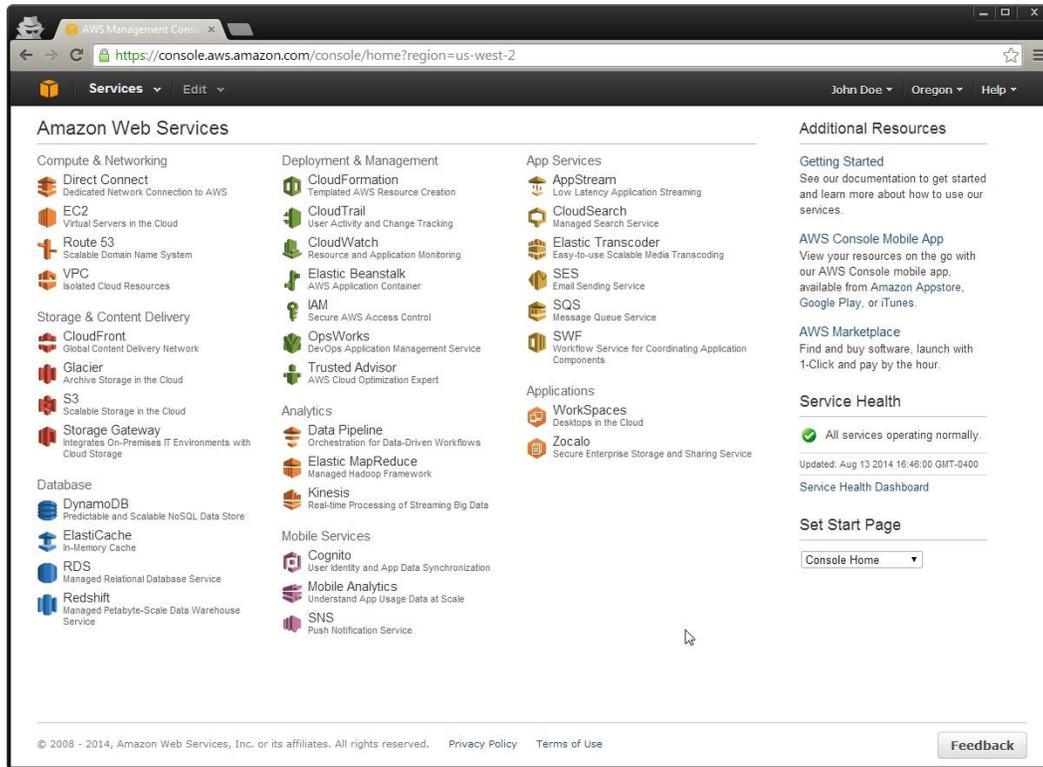


On the **Sign In** page, fill in your email and password. Click **Sign in using our secure server**.



5 Overview of AWS Management Console

You should now see the **AWS Management Console** page with the title **Amazon Web Services**. The page lists all of the available AWS services. To access the AWS Management Console directly, open your web browser to <https://console.aws.amazon.com>.



5.1 Shortcut Bar

At the top of the page, you'll see the shortcut bar. You can add the services you use most often to the shortcut bar for quicker navigation. To customize the shortcut bar, click **Edit** and then drag the services of your choice to the bar. On the top right of the shortcut bar, you'll see your name menu, a region menu, and the Help menu. The name menu provides access to your **Account** page, **Billing & Cost Management** page, and **Security Credentials** page. It also allows you to sign out. Amazon set the login sessions to expire after 12 hours so you'll have to log in again to continue using the services.

5.2 Categories and Services

Below the shortcut bar, you'll see all the Amazon services divided into categories. At the time of this publication, the categories are:

- Compute & Networking
- Storage & Content Delivery
- Database
- Deployment & Management
- Analytics
- Mobile Services
- App Services
- Application

5.3 Sidebar

On the right side of the page, you'll see the **Additional Resources**, **Service Health**, and **Set Start Page**. All of these sections provide you with additional tools to enhance your usage of AWS.

6 Launch an Instance using Amazon EC2

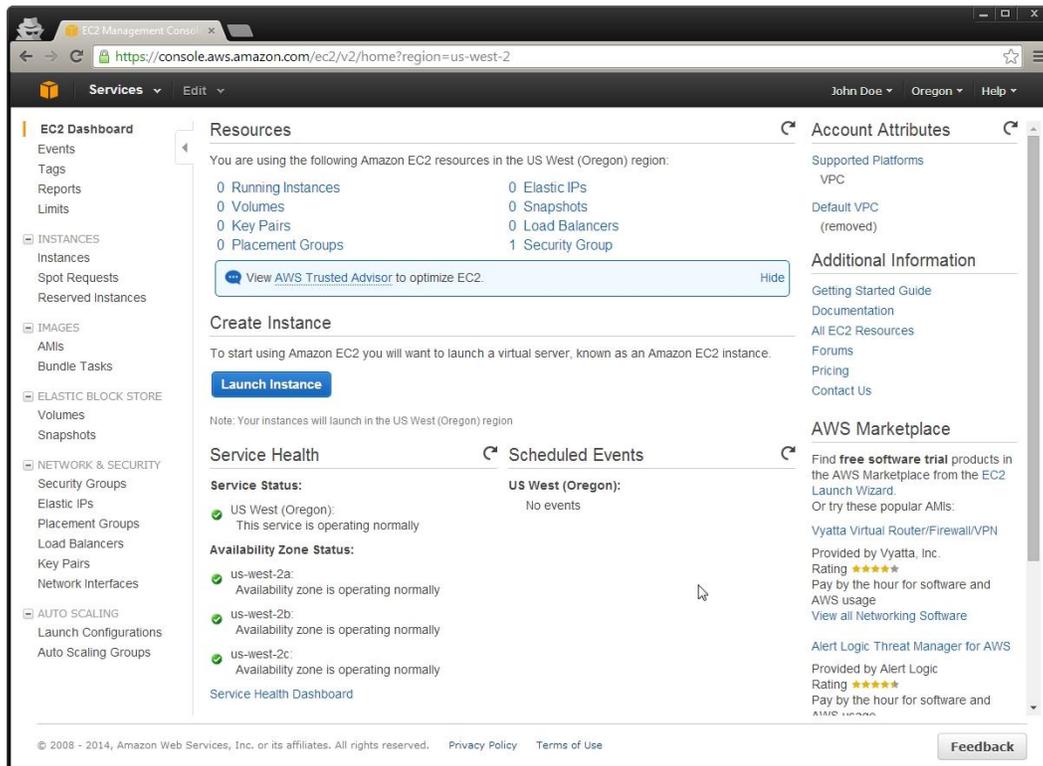
An instance in Amazon Elastic Compute Cloud (EC2) is a virtual server. It will boot up and behave just like a physical server, but it shares resources like CPU and RAM with other virtual servers. There are different instance sizes and each one costs a different amount per hour. You can launch either a Linux instance or a Windows instance. Windows instances typically cost twice the amount of a Linux instance because there are additional licensing fees associated with Windows. All licensing costs are included in the price per hour. Along with the price per hour which is charged when an instance is running, there are additional costs for each GB of data in and out as well as storage used. Be sure to look at the pricing model before continuing. For a micro instance running a typical WordPress website, it will cost around \$15 a month.

6.1 Linux Distributions vs Windows

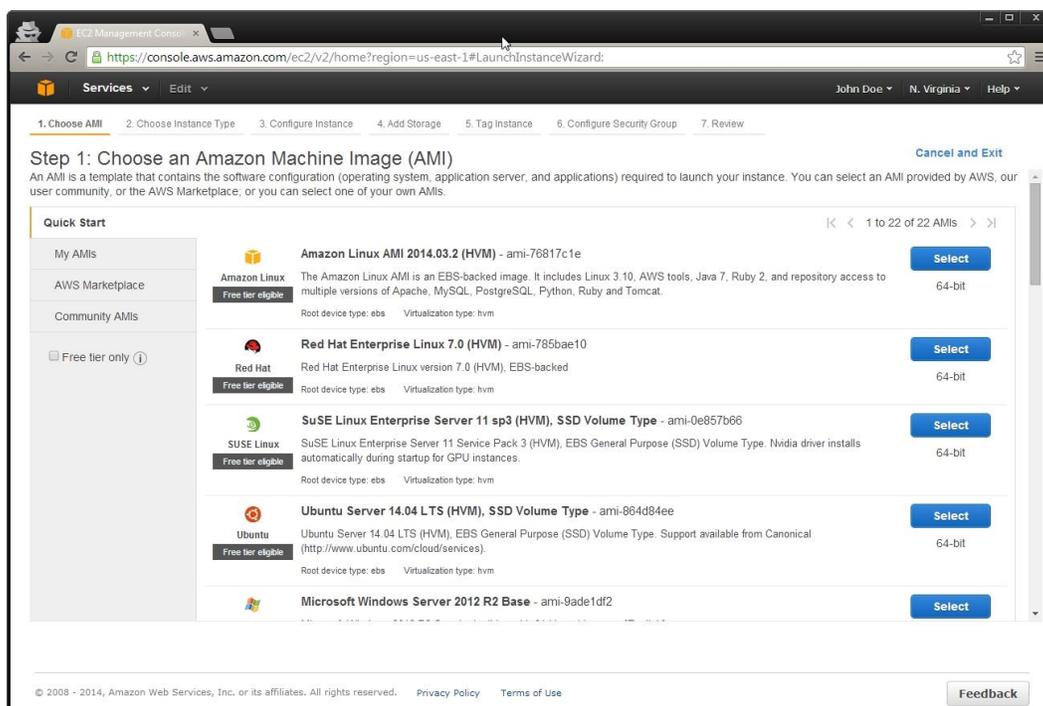
Linux is much different from Windows. Windows is created by Microsoft and every few years, a new major version is released like Windows 7 and Windows 8. For the most part, if a piece of software is created for Windows 7, it will also run on Windows 8. Linux distributions are created by many different companies. A few of the available Linux versions on Amazon are Amazon Linux, Red Hat Linux, SuSE Linux, and Ubuntu Server Linux. If a piece of software is developed for Red Hat Linux, it must be recompiled before use on any other version of Linux. This sounds complicated, but it's actually a process that developers have already done for you. Linux uses software called **package managers** to automate the compilation and installation of software. In Windows, you usually go to a website, download software, and then click through the installation process. In Linux, you type in a single command and then it installs the software. One of the other big differences to note is Linux is designed to run without a clickable graphical user interface (GUI). There are desktop versions of Linux with GUIs, but they will not be used in this tutorial because they require more resources and in the cloud, higher resource usage means more money.

6.2 Launch an Instance

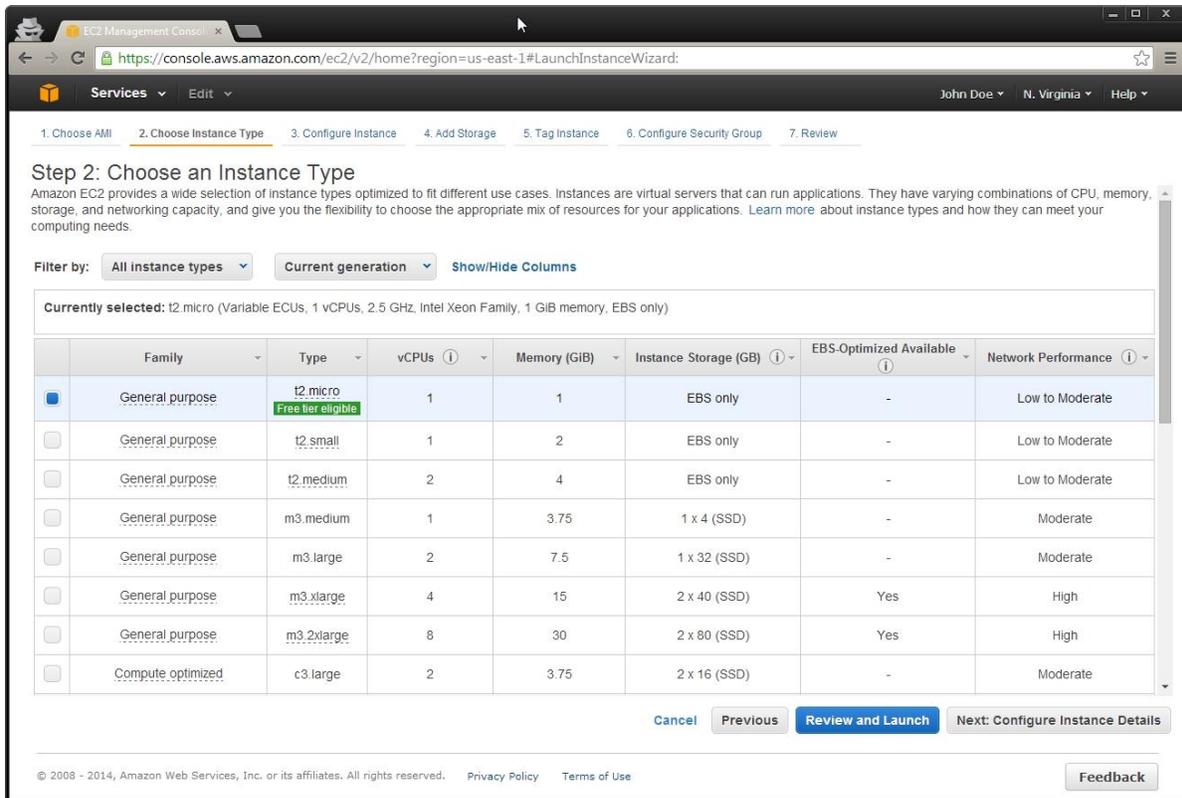
Navigate to the [AWS Management Console](https://console.aws.amazon.com/). Click on **EC2**. To access the EC2 service directly, open your web browser to <https://console.aws.amazon.com/ec2/>. Click **Launch Instance**.



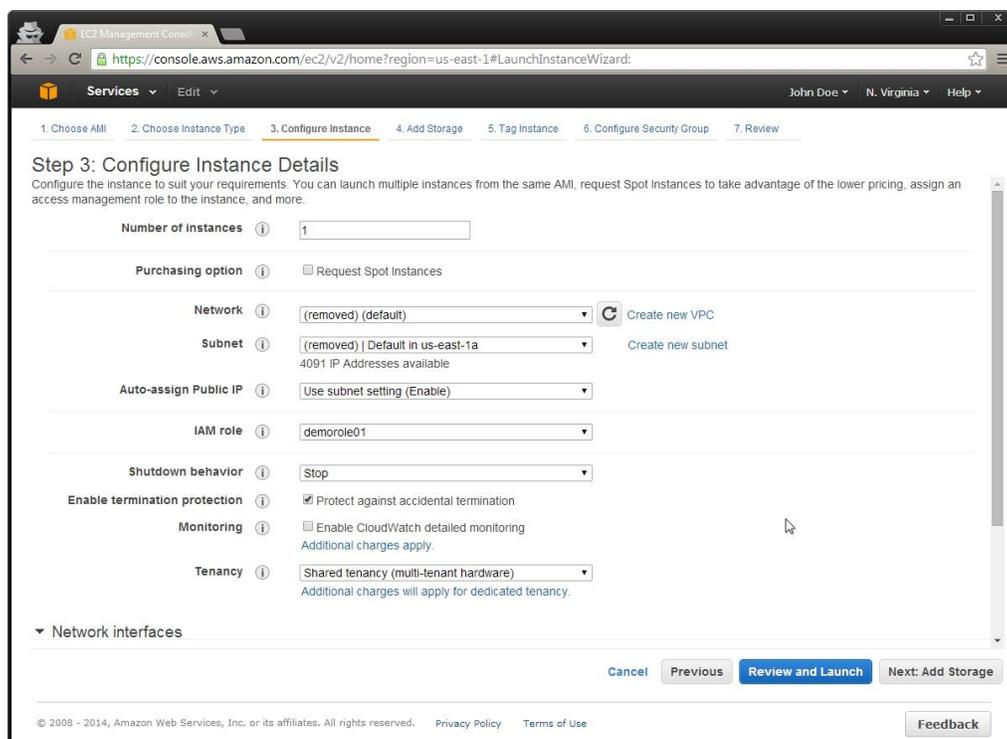
On the **Step 1: Choose an Amazon Machine Image** page, locate **Ubuntu Server 14.04 LTS (HVM)** and click **Select**. If you are more comfortable with a different version of Linux, you may select it, but it will make pieces of the guide difficult to follow.



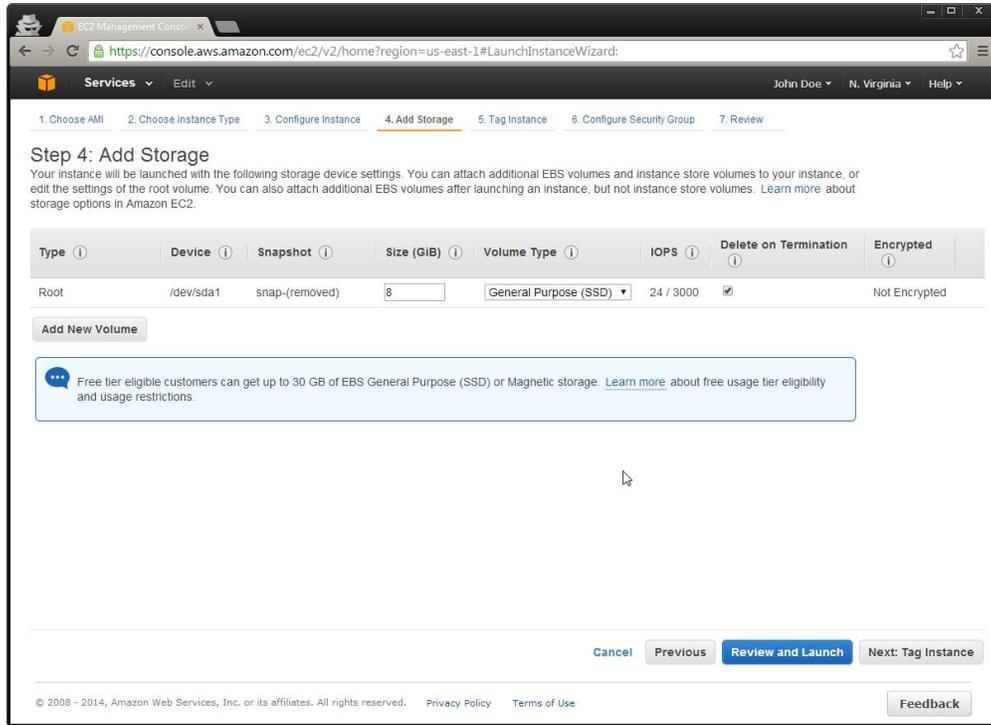
On the **Step 2: Choose and Instance Type** page, select the **t2.micro** instance. Click **Next: Configure Instance Details**.



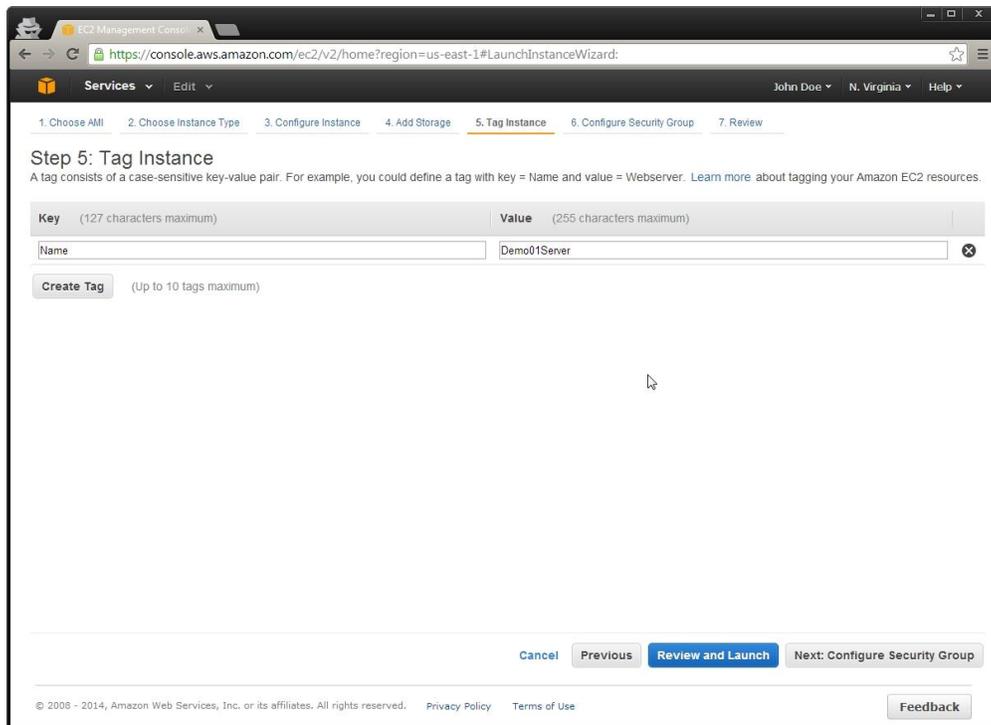
On the **Step 3: Configure Instance Details** page, select **Stop** from the **Shutdown behavior** dropdown. Check the box, **Enable termination protection**. If you would like the ability to make requests to the AWS API easily, you'll need to create an IAM Role and then select it the **IAM role** dropdown. This piece is not necessary to setup WordPress. Click **Next: Add Storage**.



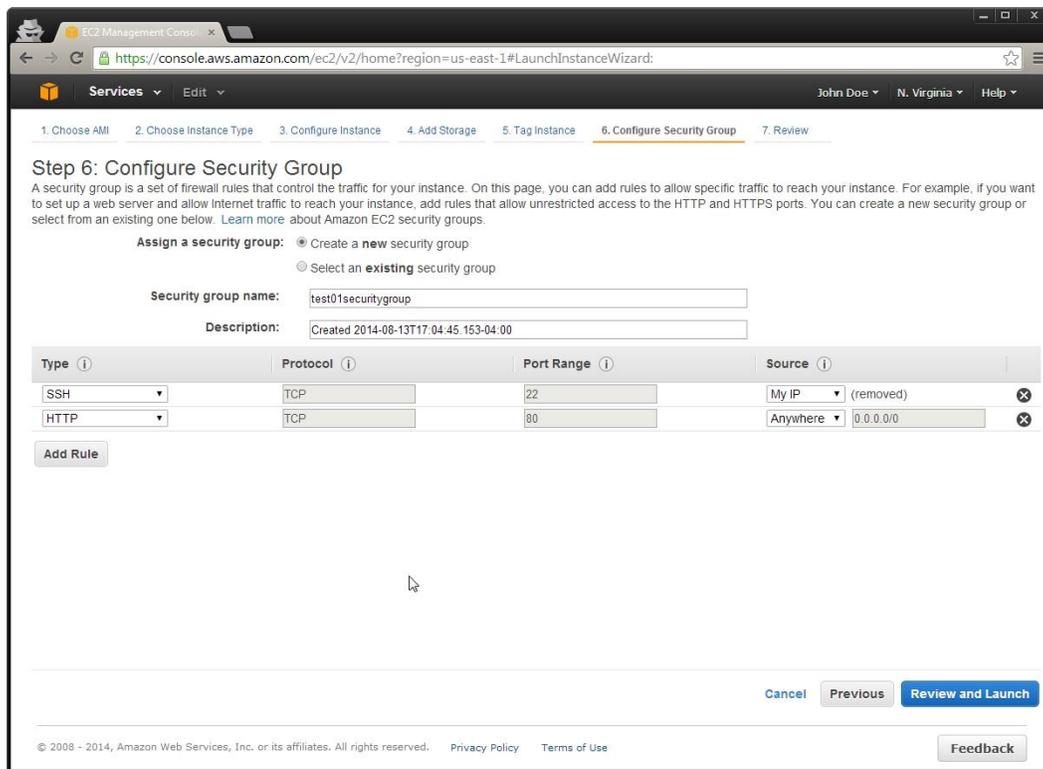
On the **Step 4: Add Storage** page, you can see the volume that will be created for the instance. The volume will essentially be the hard drive for your instance. All the files will be stored on the volume. No changes need to be made to this page unless you know you're going to have a very large amount of rich content like videos on your website. You can also change the size of the volume later, but it will require a few steps documented on <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-expand-volume.html>. To take maximum advantage of the **Free Tier**, you can change the size to 30 (GB). Click **Next: Tag instance**.



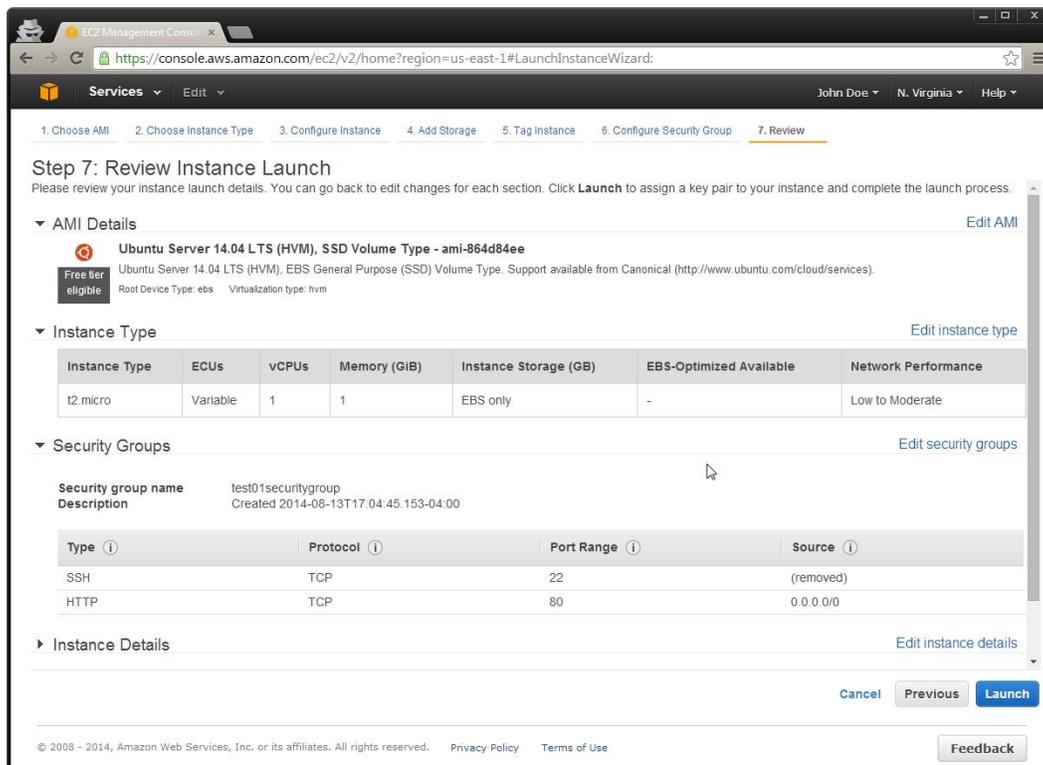
On the **Step 5: Tag instance** page, create a **Key** called Name with a **Value** of the name you would like to give to your instance. Click **Next: Configure Security Group**.



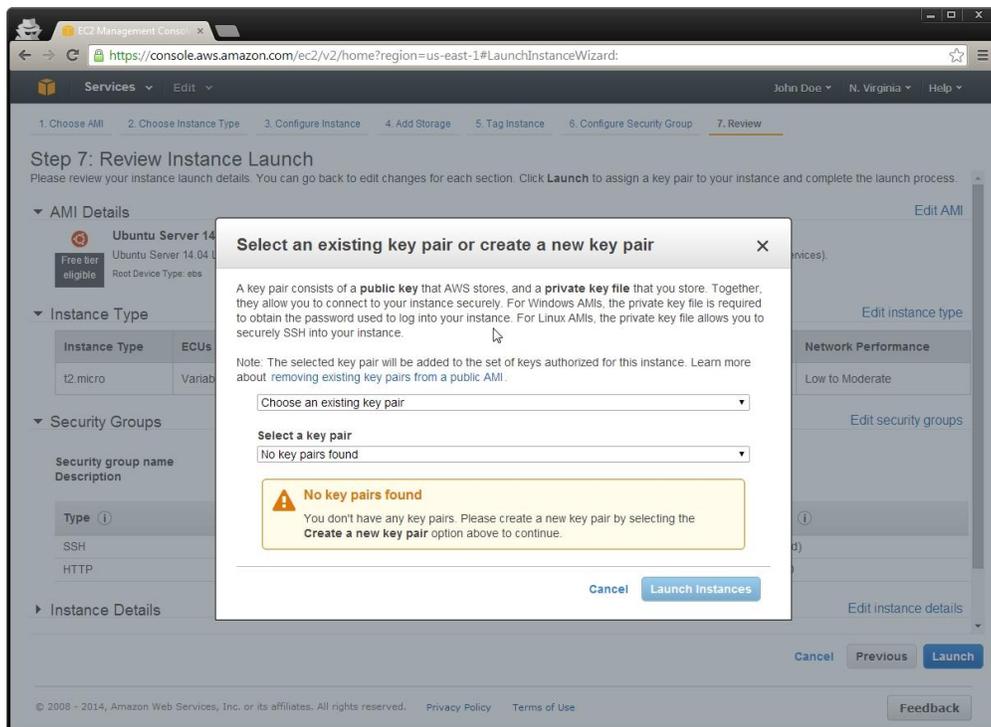
On the **Step 6: Configure Security Group** page, give the security group a name and a description. Click **Add Rule**, select **HTTP** from the **Type** dropdown, and select **Anywhere** from the **Source** dropdown. The security group controls the firewall for the instance. In order to access a website, you must connect to port 80 to retrieve the website. By default, Amazon disables access to all ports so you must explicitly define a rule that allows traffic to access port 80 (HTTP). Click **Review and Launch**.



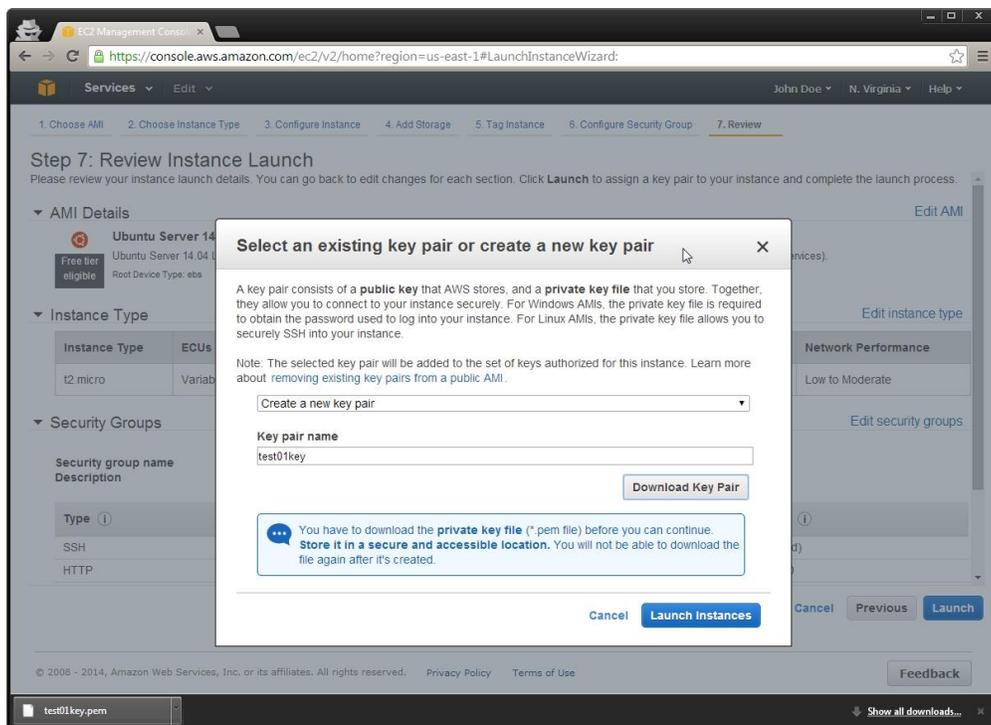
On the **Step 7: Review Instance Launch** page, verify all the information and click **Launch**.



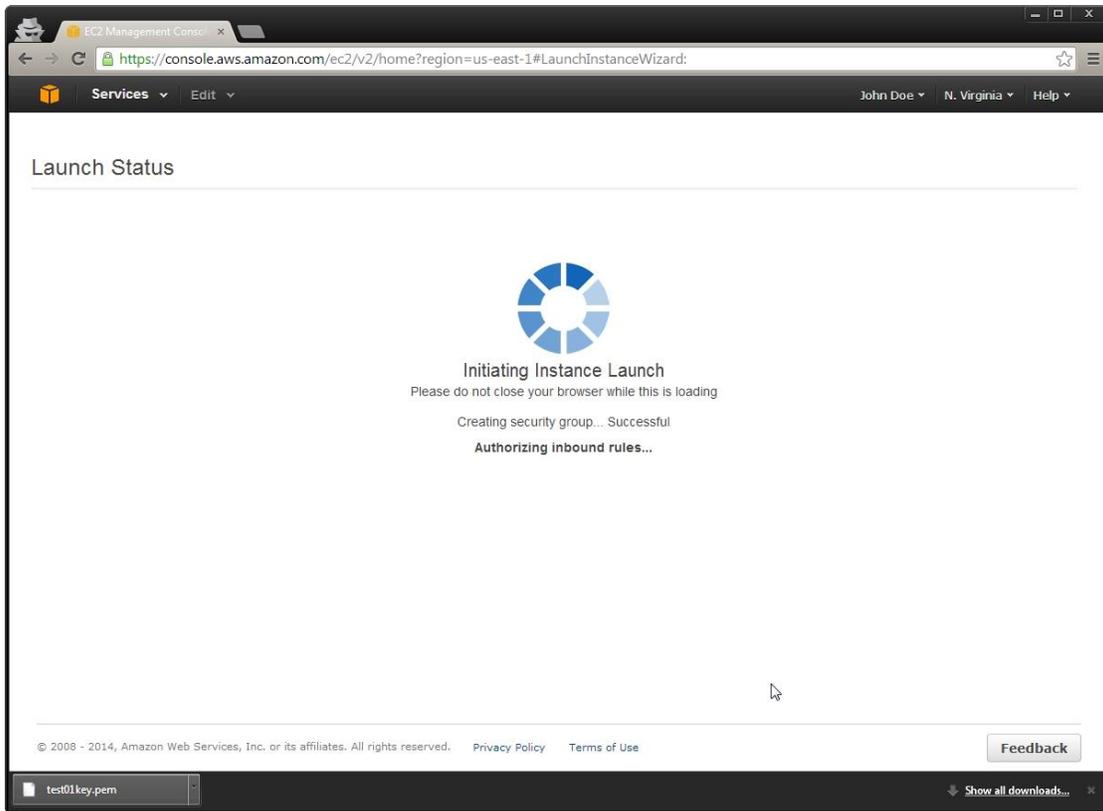
You should see a popup called **Select an existing key pair or create a new key pair**. A key pair is a public and private key that allows you to connect and log in to your instance. The private key should NEVER be shared with anyone else because it's the equivalent of a password. The public key can be given to anyone and will actually be stored on the instance in the `/home/ubuntu/.ssh/authorized_keys` file. If someone adds your public key to this file on their server, you will be able to log in to their server.



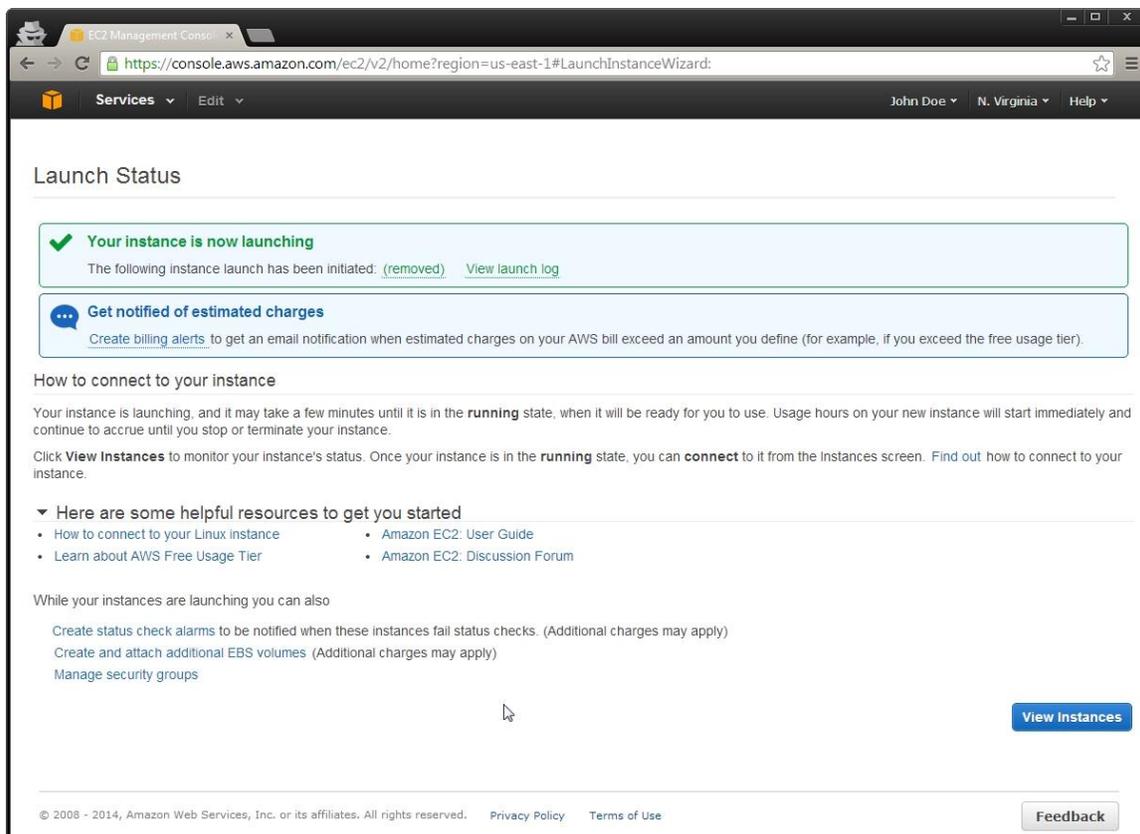
Select **Create a new key pair** from the first dropdown. Fill in the **Key pair name** textbox. Click **Download Key Pair**. Save the downloaded `.pem` file somewhere safe. If you've already created a key pair, you can select it from the first dropdown and use the same key pair for multiple instances. Click **Launch Instances**.



On the **Launch Status** page, you should see Amazon setting up the instance.



Once the instance is ready, you should see the message, **Your instance is now launching**. Click **View Instances**.



On the Instances page, you should see **Initializing** in the **Status Checks** column. Once the instance is fully started, it will change to **2/2 checks passed**.

The screenshot shows the AWS Management Console interface for EC2 instances. The left sidebar contains navigation options such as EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES, IMAGES, ELASTIC BLOCK STORE, NETWORK & SECURITY, and AUTO SCALING. The main content area displays a table of instances with columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. A single instance named 'Demo01Server' is listed with a state of 'running' and a status check of 'Initializing'. Below the table, the details for the selected instance are shown, including Instance ID, Instance state, Instance type, Private DNS, Private IPs, Public DNS, Public IP, Elastic IP, Availability zone, and Security groups.

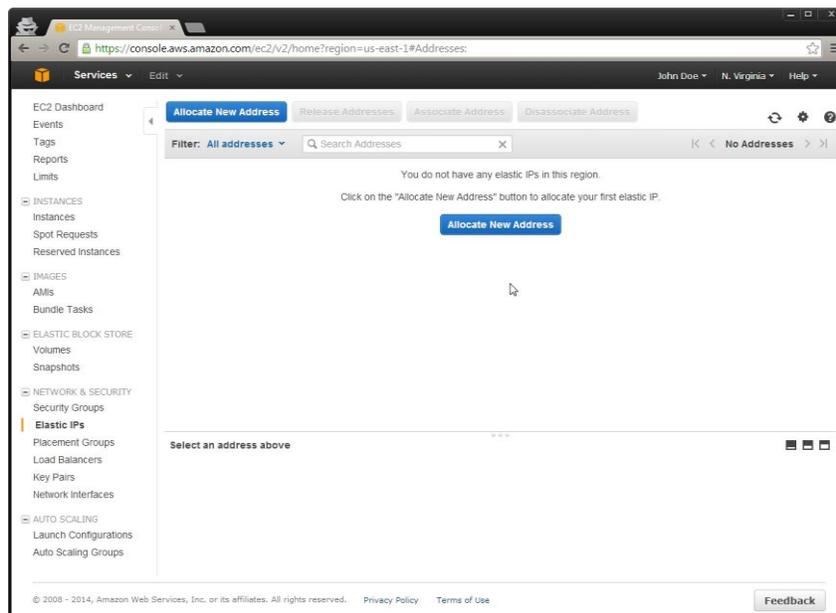
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
Demo01Server	(removed)	t2.micro	us-east-1a	running	Initializing	None	(removed)

Instance: (removed) (removed)			
Description	Status Checks	Monitoring	Tags
Instance ID	(removed)	Public DNS	(removed)
Instance state	running	Public IP	(removed)
Instance type	t2.micro	Elastic IP	-
Private DNS	(removed)	Availability zone	us-east-1a
Private IPs	(removed)	Security groups	test01securitygroup . view rules

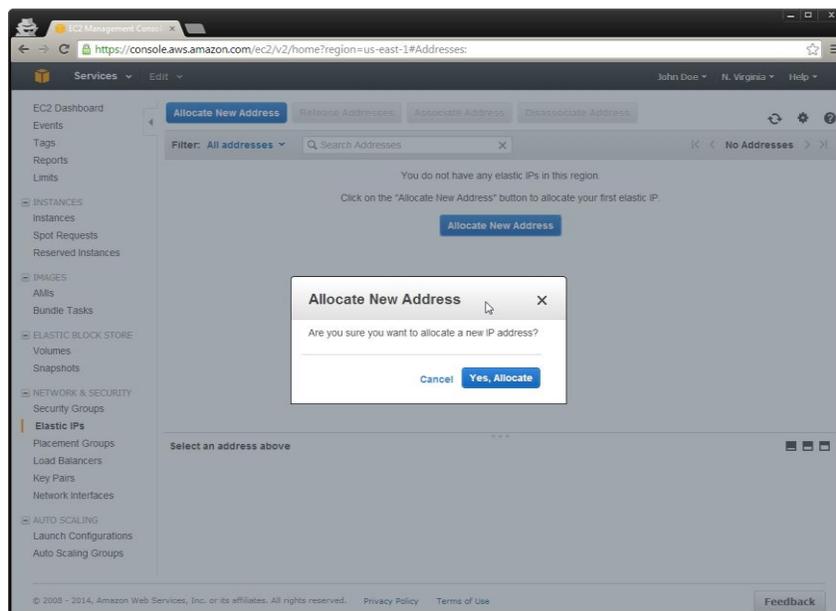
6.3 Assign an Elastic IP

IP stands for Internet Protocol. An IP address is the equivalent of a home address for a person. If I want to visit a friend, I'll use the home address to find out where he or she lives. An IP address is the location for a server on the internet. Every device on the internet has a public IP. The EC2 instance has two IPs: a private IP and a public IP. The private IP is used by other instances on the same network. The public IP is used by everything else on the internet. Amazon assigns a public IP to every instance, but if you restart your instance, the public IP will change. The domain name needs to be pointed at a public IP that does not change so Amazon a solution called Elastic IPs. Elastic IPs are public IPs that do not change. Every instance can have one Elastic IP free of charge.

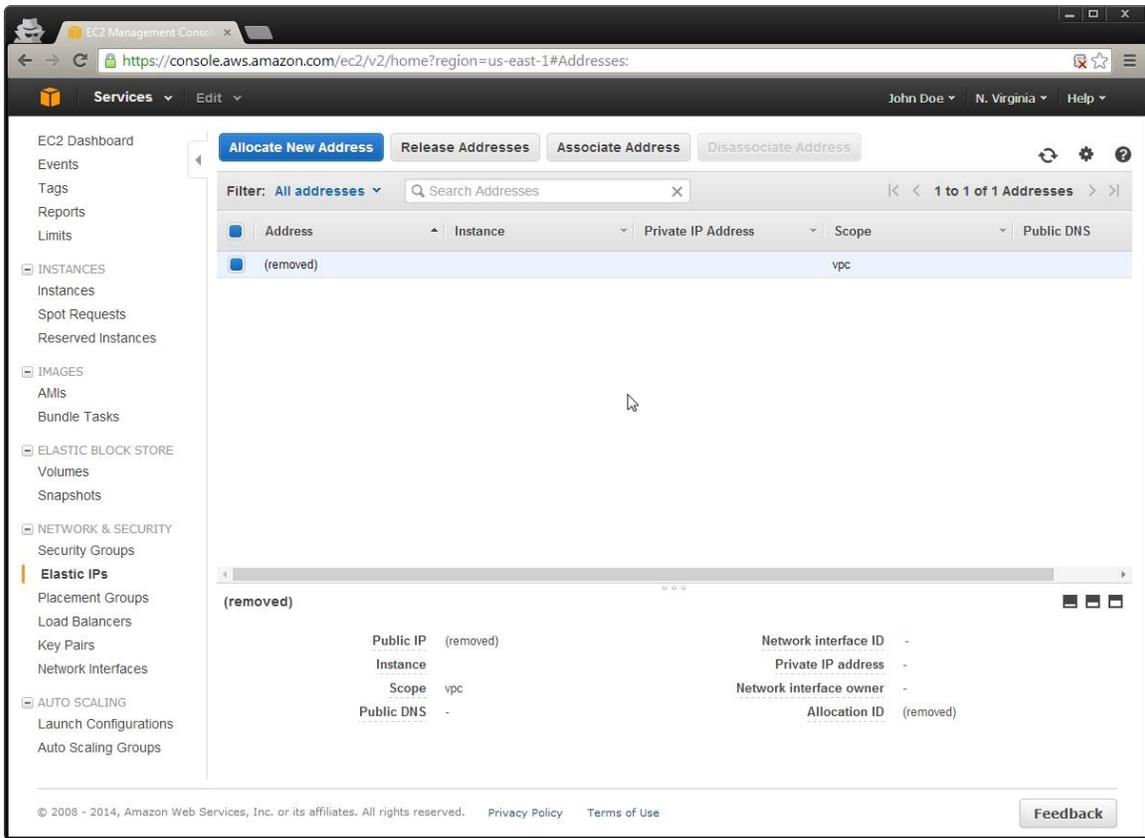
Navigate to the [AWS Management Console](https://console.aws.amazon.com/ec2/). Click on **EC2**. To access the Ec2 service directly, open your web browser to <https://console.aws.amazon.com/ec2/>. Click **Elastic IPs** on the left menu under the **Network & Security** menu. Click **Allocate New Address**.



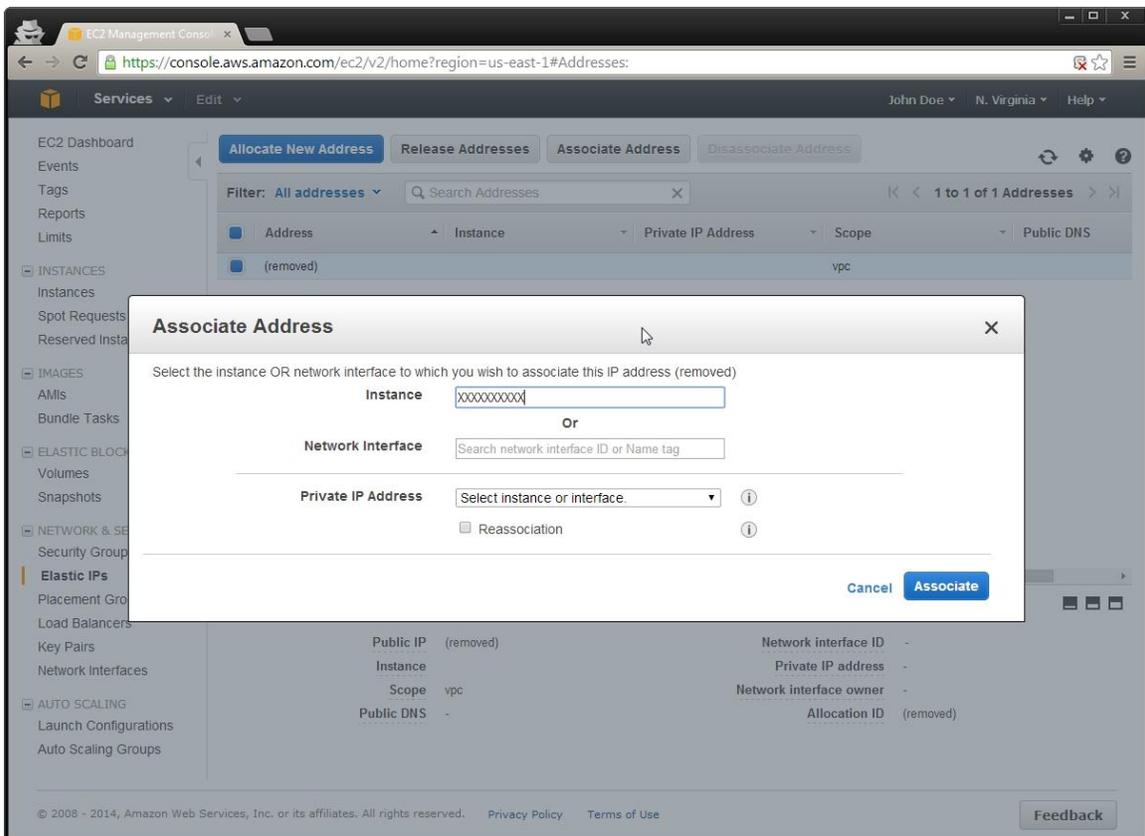
On the **Allocate New Address** dialog, click **Yes, Allocate**.



Click on the new IP address and click **Associate Address**. Save the IP so you can reference it later.



On the **Associate Address** dialog, select the instance to receive the Elastic IP and then click **Associate**.

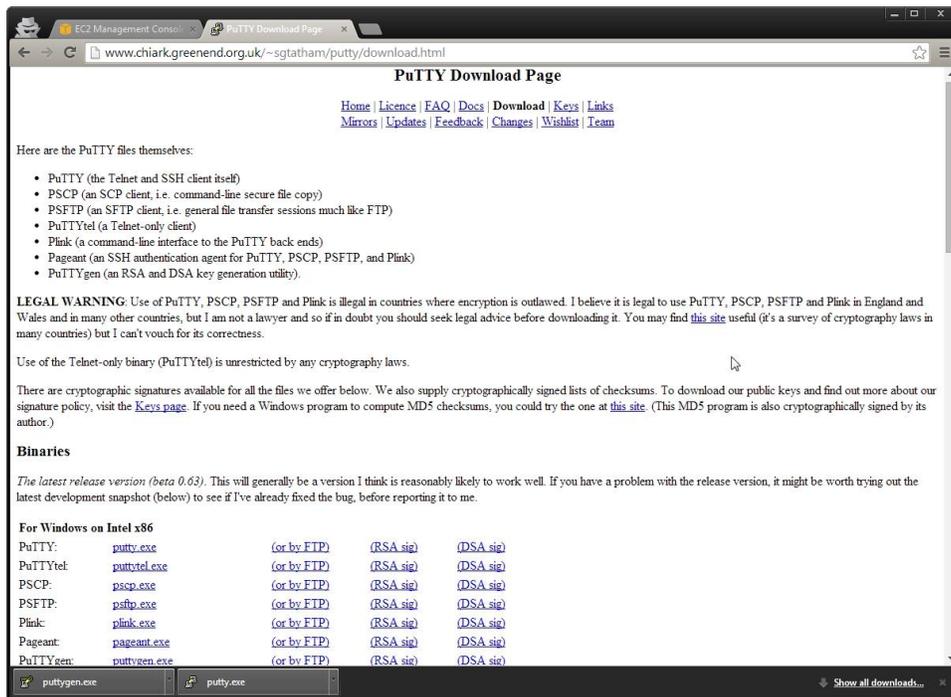


7 Access an Instance

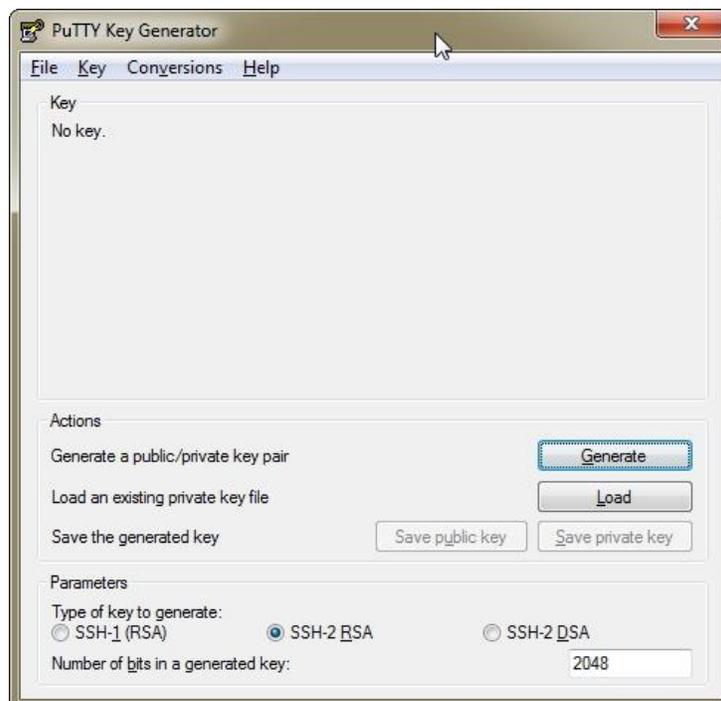
To connect to an AWS instance, you will need an SSH client. SSH stands for Secure Shell and allows you to send commands to a server through an encrypted connection. PuTTY, a free SSH client, will be used for this tutorial. You'll also need PuTTYgen to convert the .pem file into a PuTTY compatible private key.

7.1 Convert .PEM Key to .PPK PuTTY Key using PuTTYgen

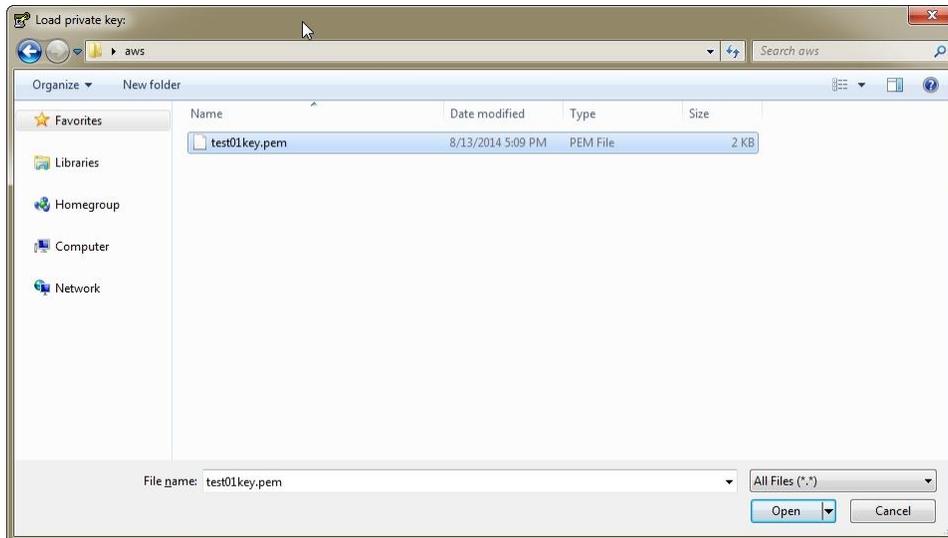
Open your web browser to <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html> and click on putty.exe and puttygen.exe to download both applications.



Run **puttygen.exe**. Click **Load** to load a private key.



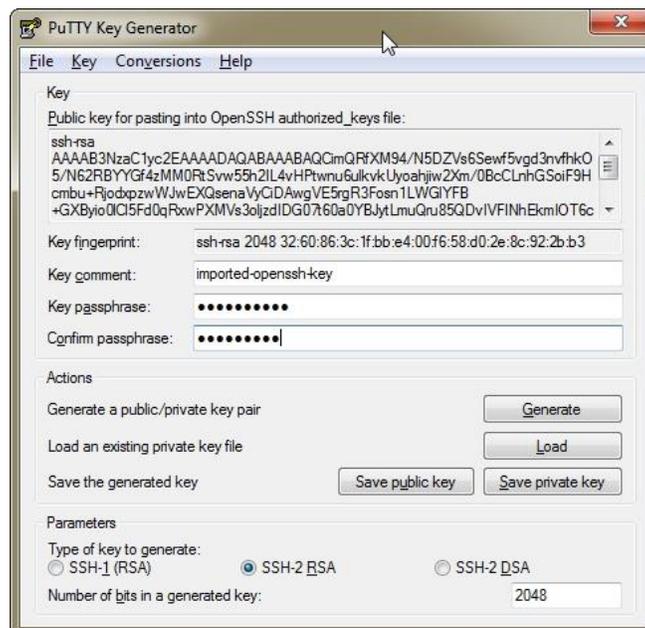
On the **Load private key** screen, navigate to the .pem file you downloaded from Amazon and click **Open**.



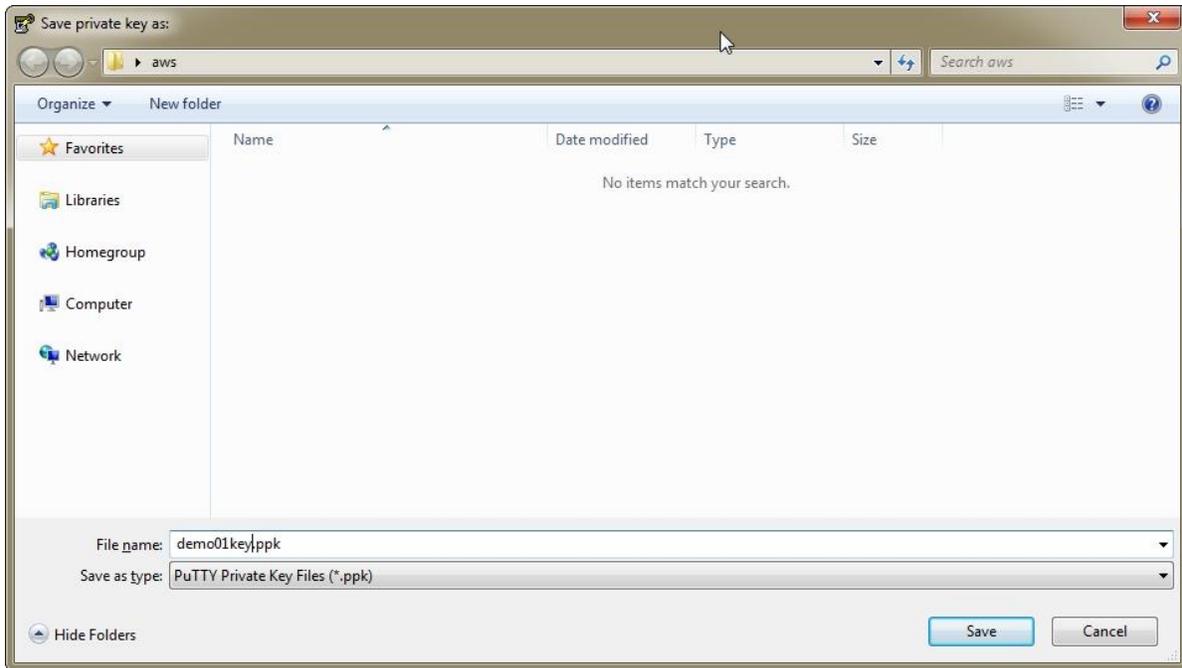
You should receive a notice that says, **Successfully imported foreign key**. Click **OK**.



On the **PuTTY Key Generator** form, you should see the public key at the top. This is the public key that is already stored in the /home/ubuntu/.ssh/authorized_keys file on the EC2 instance. Fill in the **Key passphrase** and **Confirm Passphrase** textboxes with a new password that you will type in every time you connect to your EC2 instance. Click **Save private key**.



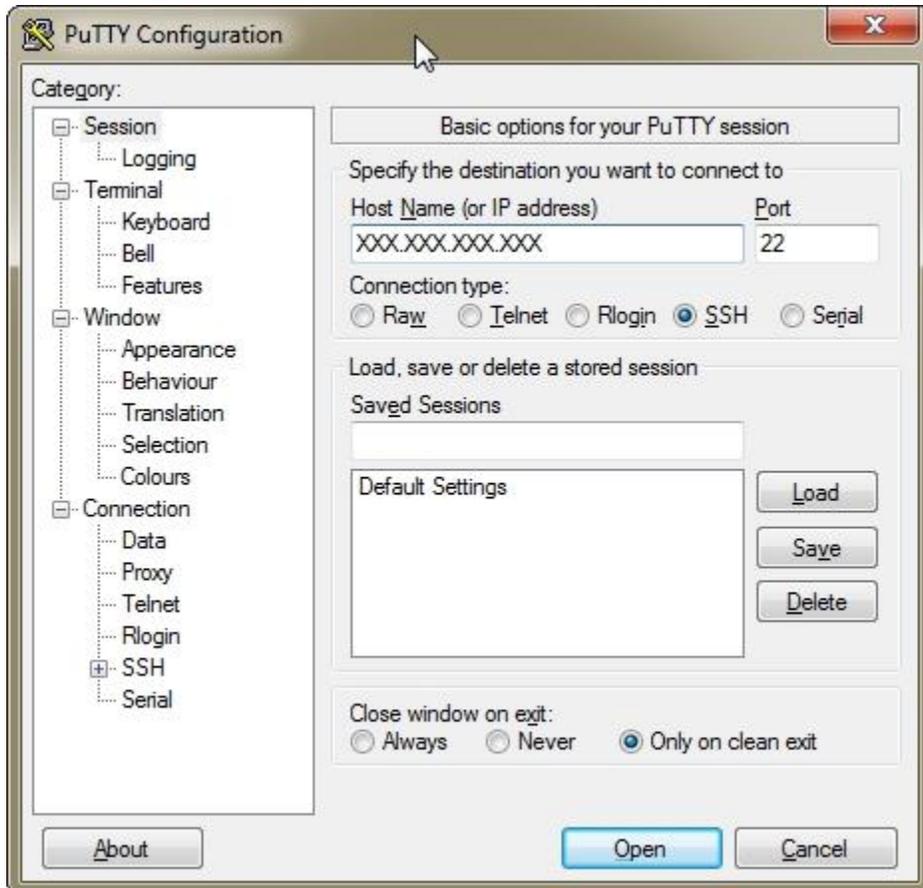
On the **Save private key as** dialog, navigate to a secure location for the key and type in a name. Click **Save**.



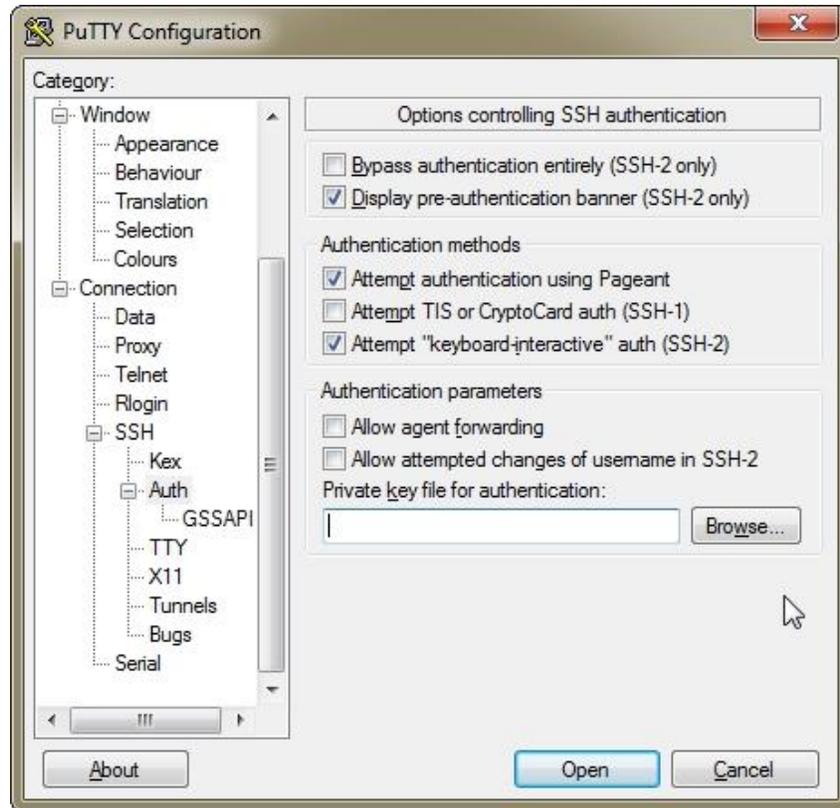
You can exit out of PuTTYgen.

7.2 Connect to an Instance using PuTTY

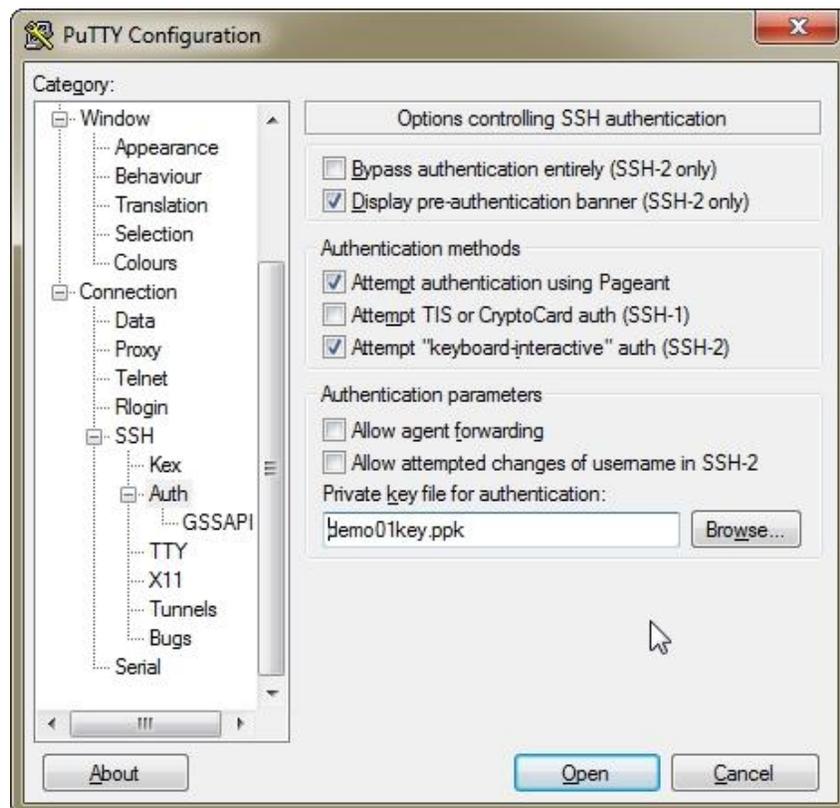
Run `putty.exe`. Type in the Elastic IP (public IP) or domain name for the instance into the **Host Name** textbox.



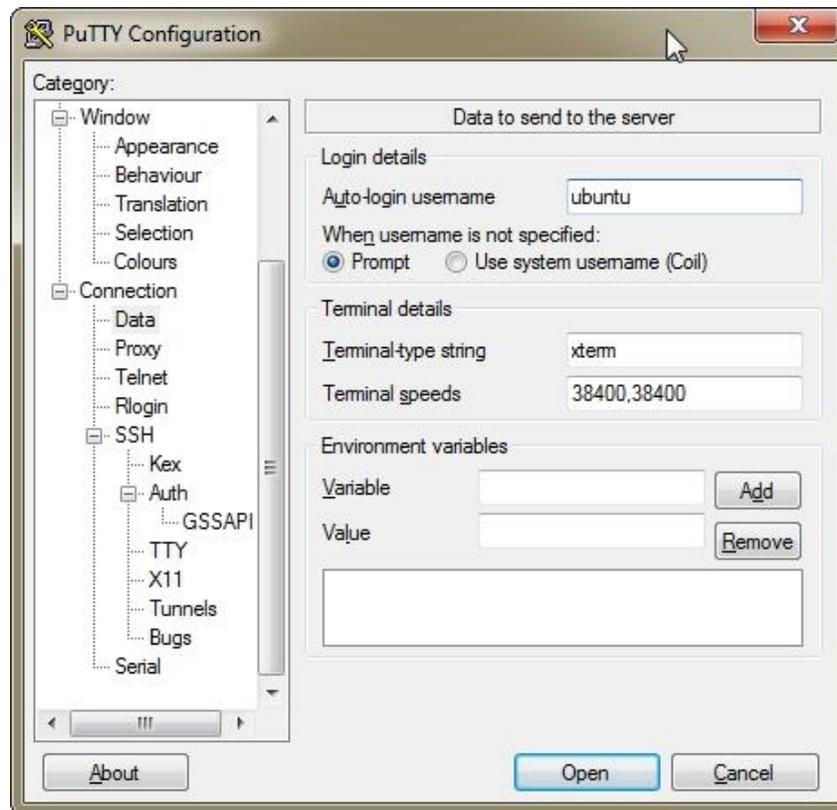
Using the left menus, navigate to **Connection** -> **SSH** -> **Auth**. Click the **Browse** button to the right of the **Private key file for authentication** textbox.



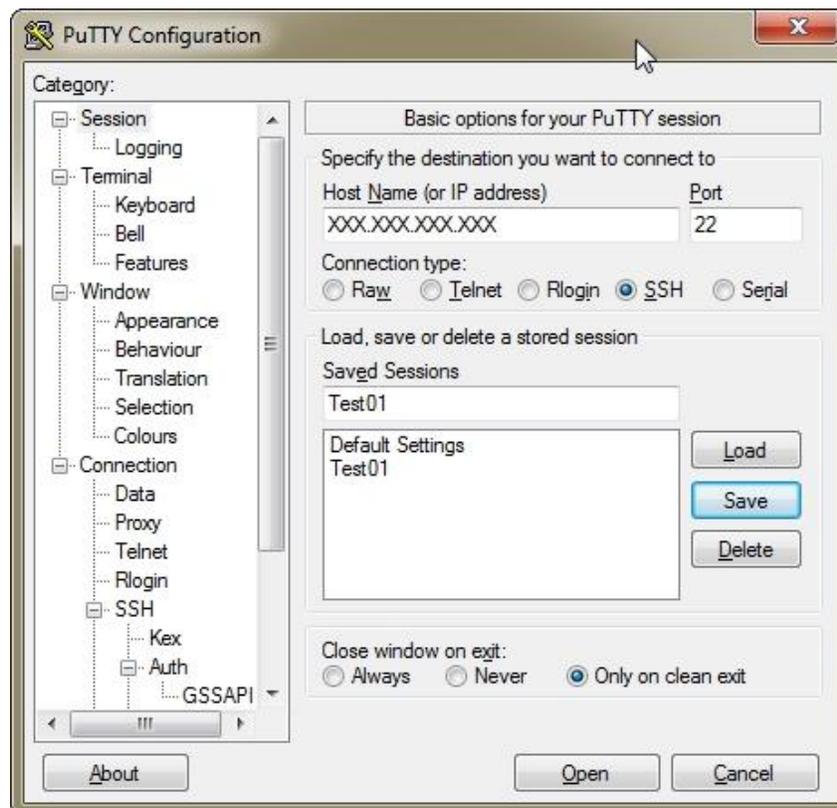
On the **Select private key file** dialog, select the .PPK private key and click **Open** so it shows up in the **PuTTY Configuration** form.



Using the left menus, navigate to **Connection** -> **Data**. Fill in the **Auto-login username** textbox with **ubuntu**.



Using the left menus, navigate to **Session**. Fill in the **Saved Sessions** textbox with the name of the server so you can recognize it. Click **Save**. The next time you open PuTTY, you can easily click on the session name and it will reload all the settings. Click **Open** to connect to the server.



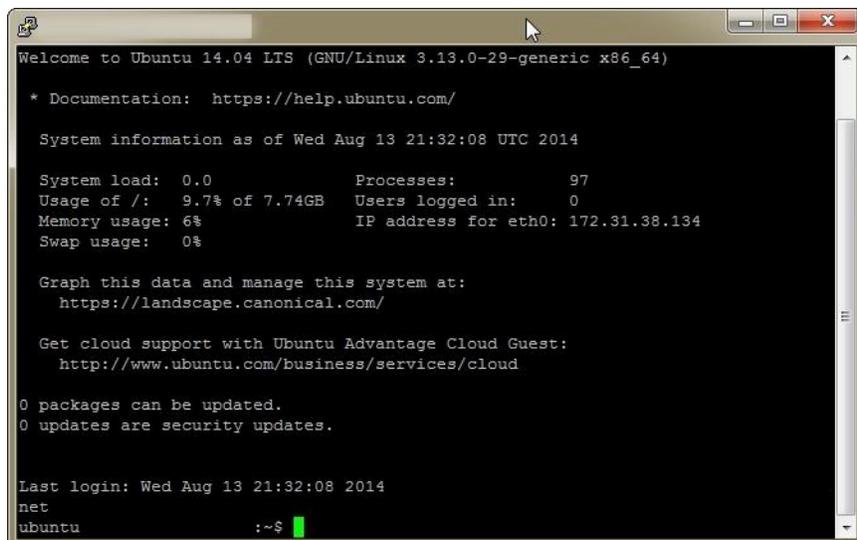
If you are prompted with a PuTTY Security Alert, just click **Yes**. You will see this alert the first to you connect to every server. If this prompt appears more than one, it may mean you're not connecting to the correct instance.



You should get a terminal window that asks for the passphrase you created in PuTTYgen. Type in the passphrase and press **Enter**.



You should see a terminal window that says **Welcome to Ubuntu** at the top. You've successfully connected to the Amazon EC2 instance.



8 Buy a Domain Name using Amazon Route 53

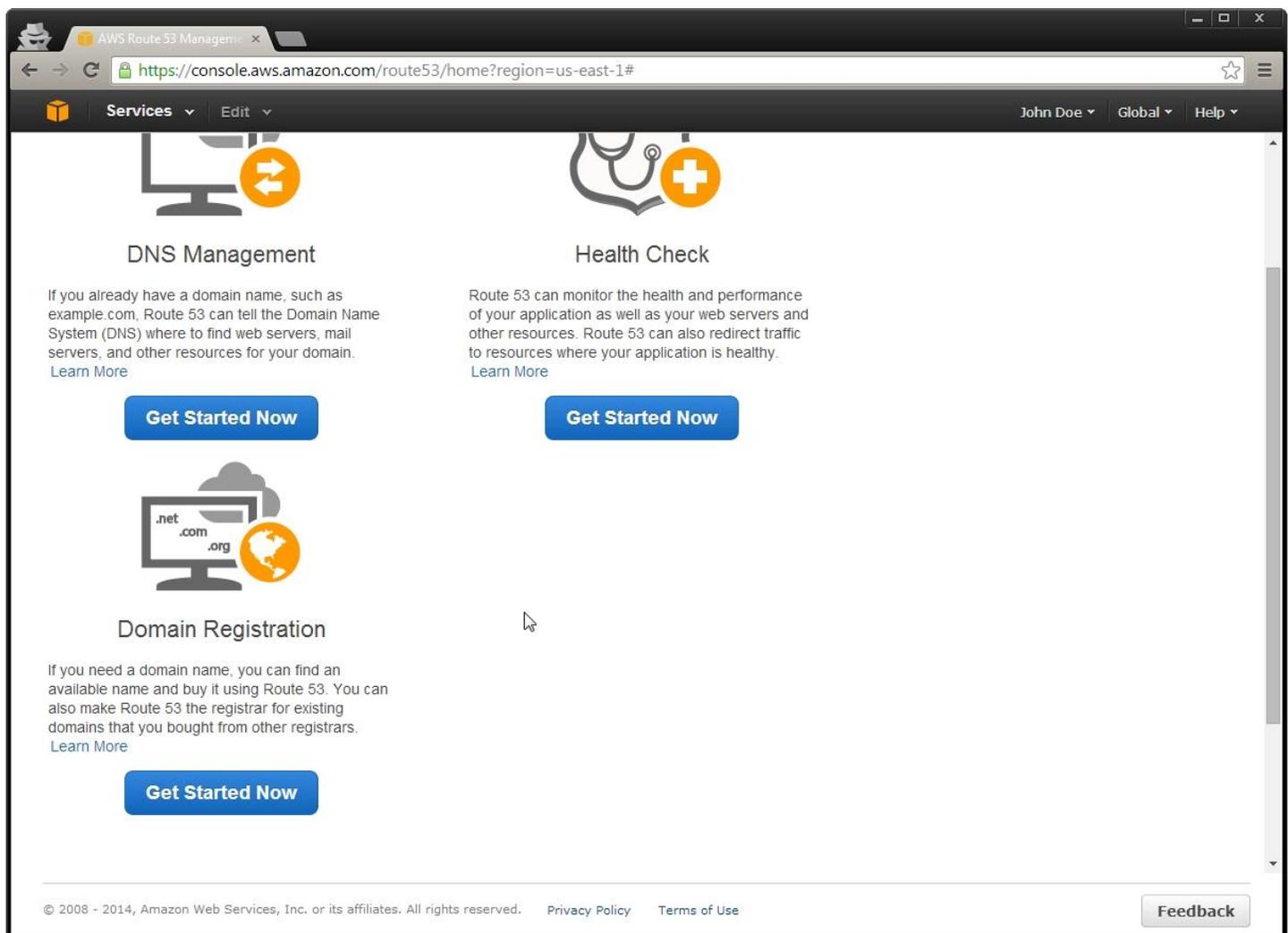
You don't actually need a .com or .net domain name to setup WordPress. You can run it directly from an IP address if you would like. Domain names just make it much easier for you and other people to access your website. The .com domains are \$12 a year, .net domains are \$10 a year, and .org domains are \$12 a year. There are also many other top-level domains (TLDs) (.me, .name, etc) available for purchase and range in price. The DNS will cost about \$0.90 per month for a typical website.

8.1 Domain WHOIS Record Privacy Protection

Amazon also offers privacy protection for your WHOIS record at no additional cost. When you buy a domain name, you have to provide information like a name, address, email, and phone number that is publically available. If you use your home address when you register for your domain and you don't want other people to see that information, you can enable privacy protection. Many other domain registrars charge an additional fee for privacy protection.

8.2 Register a Domain

Navigate to the [AWS Management Console](#). Click on **Route 53**. To access the Route 53 service directly, open your web browser to <https://console.aws.amazon.com/route53>. Scroll to the bottom of the page so you can see the **Domain Registration** section. Click **Get Started Now**.

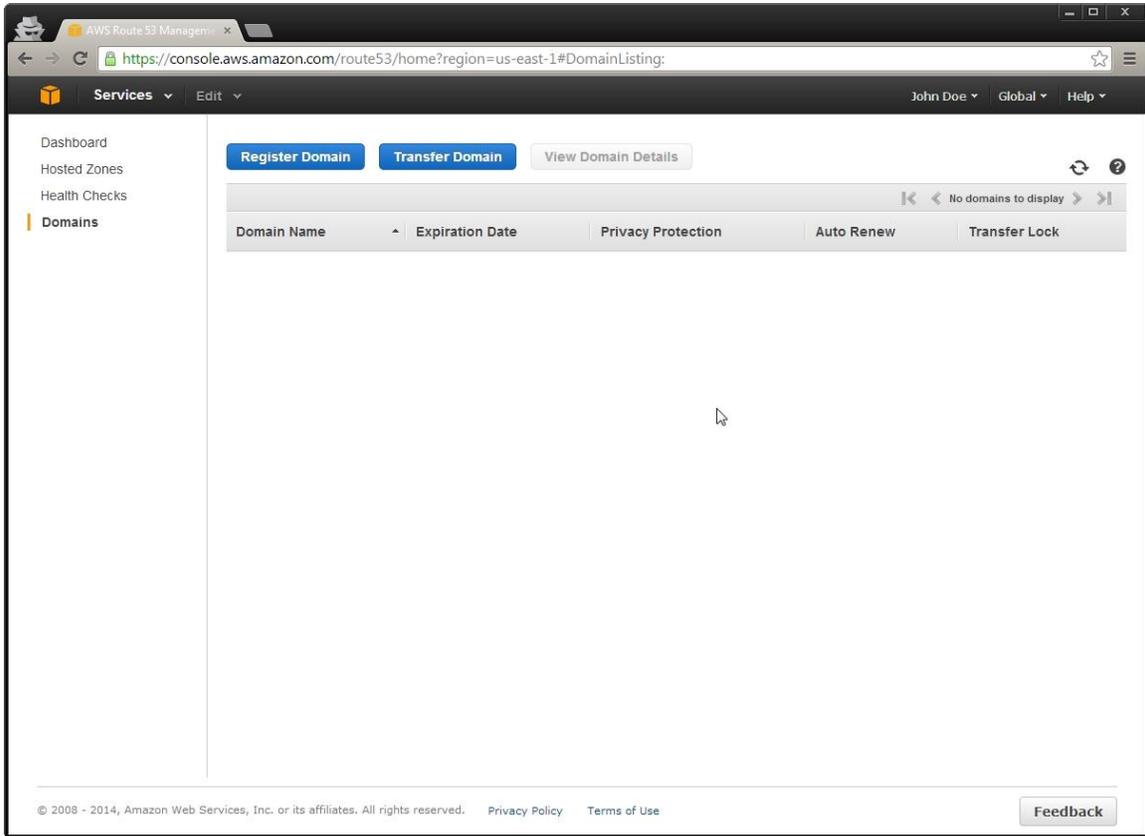


The screenshot shows the AWS Route 53 Management Console interface. The browser address bar displays <https://console.aws.amazon.com/route53/home?region=us-east-1#>. The page features a navigation bar with 'Services' and 'Edit' dropdowns, and a user profile for 'John Doe' with 'Global' and 'Help' options. The main content area is divided into three sections:

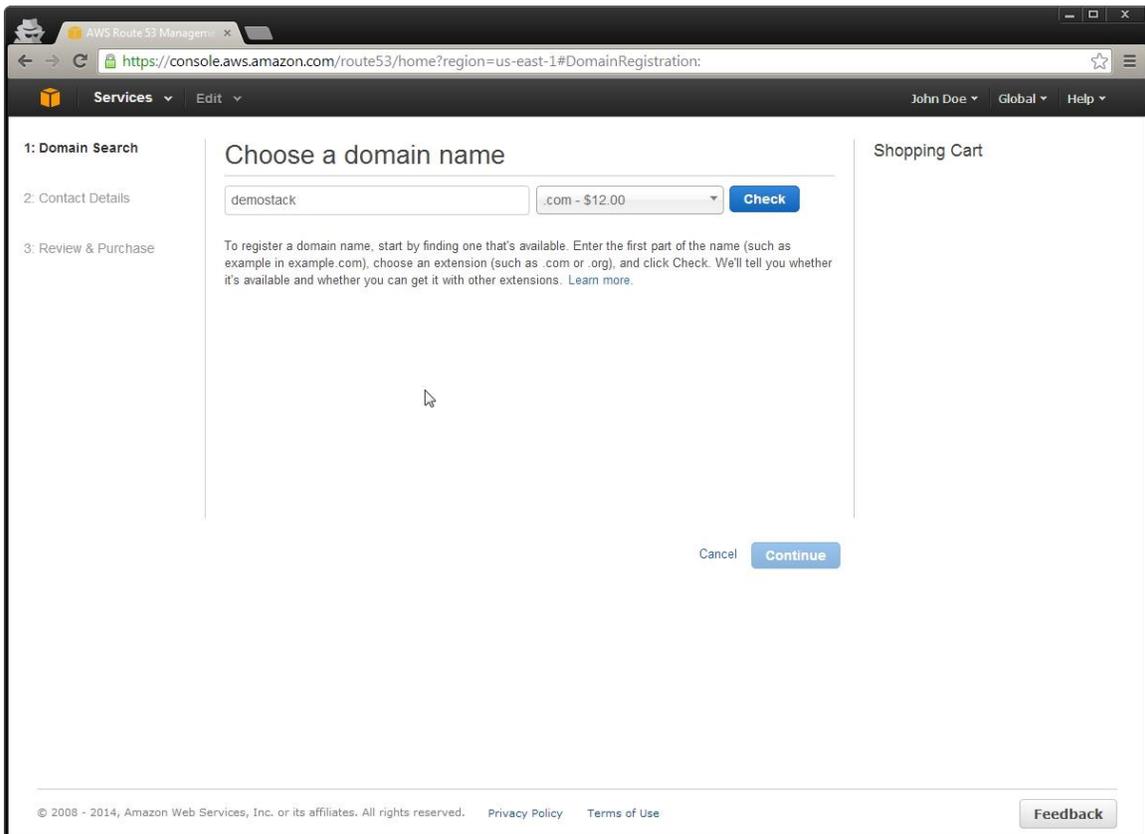
- DNS Management**: Includes an icon of a monitor with a double-headed arrow. Text: "If you already have a domain name, such as example.com, Route 53 can tell the Domain Name System (DNS) where to find web servers, mail servers, and other resources for your domain. [Learn More](#)". A blue "Get Started Now" button is present.
- Health Check**: Includes an icon of a stethoscope with a plus sign. Text: "Route 53 can monitor the health and performance of your application as well as your web servers and other resources. Route 53 can also redirect traffic to resources where your application is healthy. [Learn More](#)". A blue "Get Started Now" button is present.
- Domain Registration**: Includes an icon of a monitor with ".net" and ".org" labels and a globe. Text: "If you need a domain name, you can find an available name and buy it using Route 53. You can also make Route 53 the registrar for existing domains that you bought from other registrars. [Learn More](#)". A blue "Get Started Now" button is present.

At the bottom of the page, there is a footer with copyright information: "© 2008 - 2014, Amazon Web Services, Inc. or its affiliates. All rights reserved." and links for "Privacy Policy" and "Terms of Use". A "Feedback" button is located in the bottom right corner.

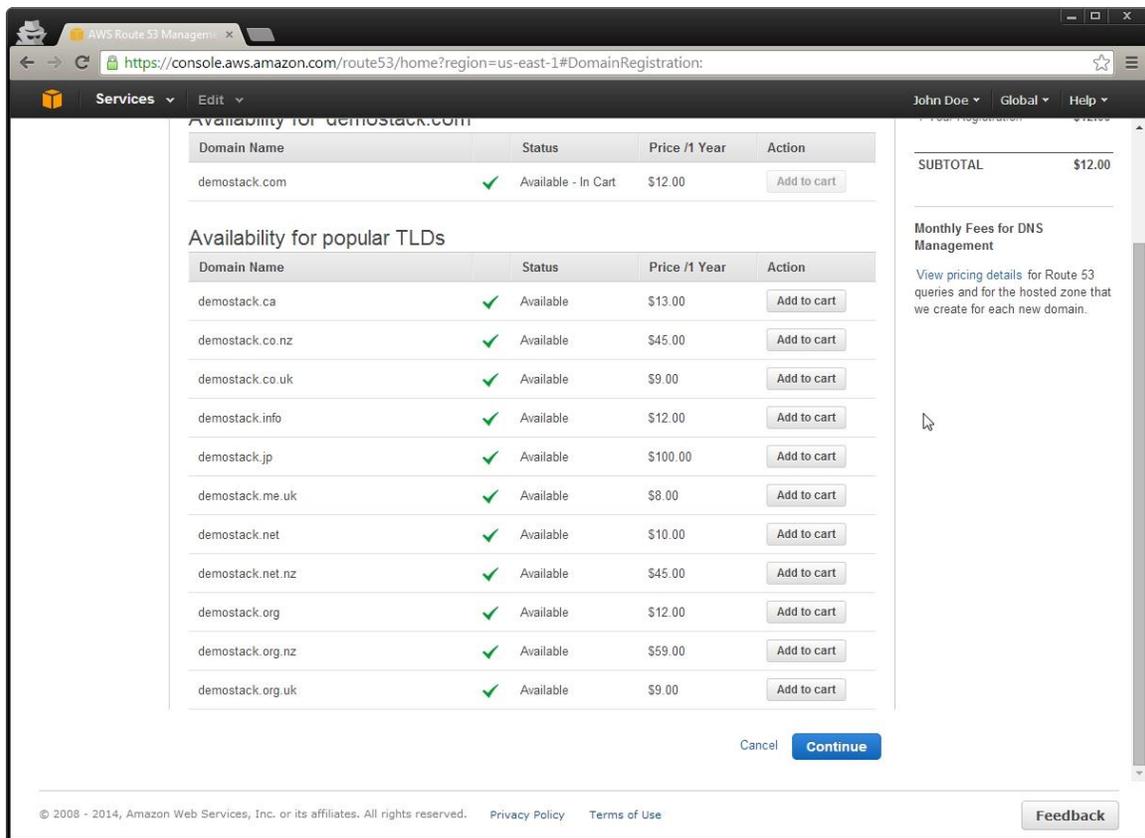
On the **Domains** page, click **Register Domain**. You also have the ability to transfer a domain to Amazon if you own one.



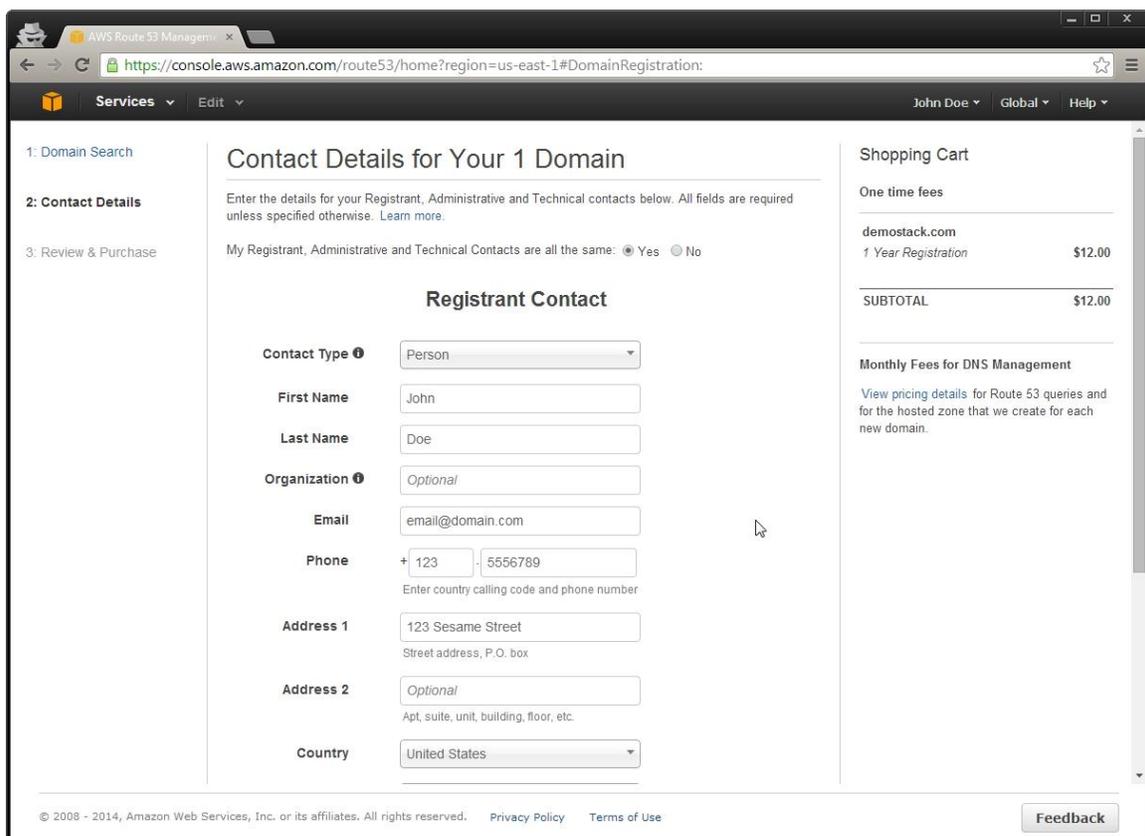
On the **Choose a domain name** page, fill in the textbox with the domain name you would like to purchase. Choose the TLD from the dropdown. Click **Check**.



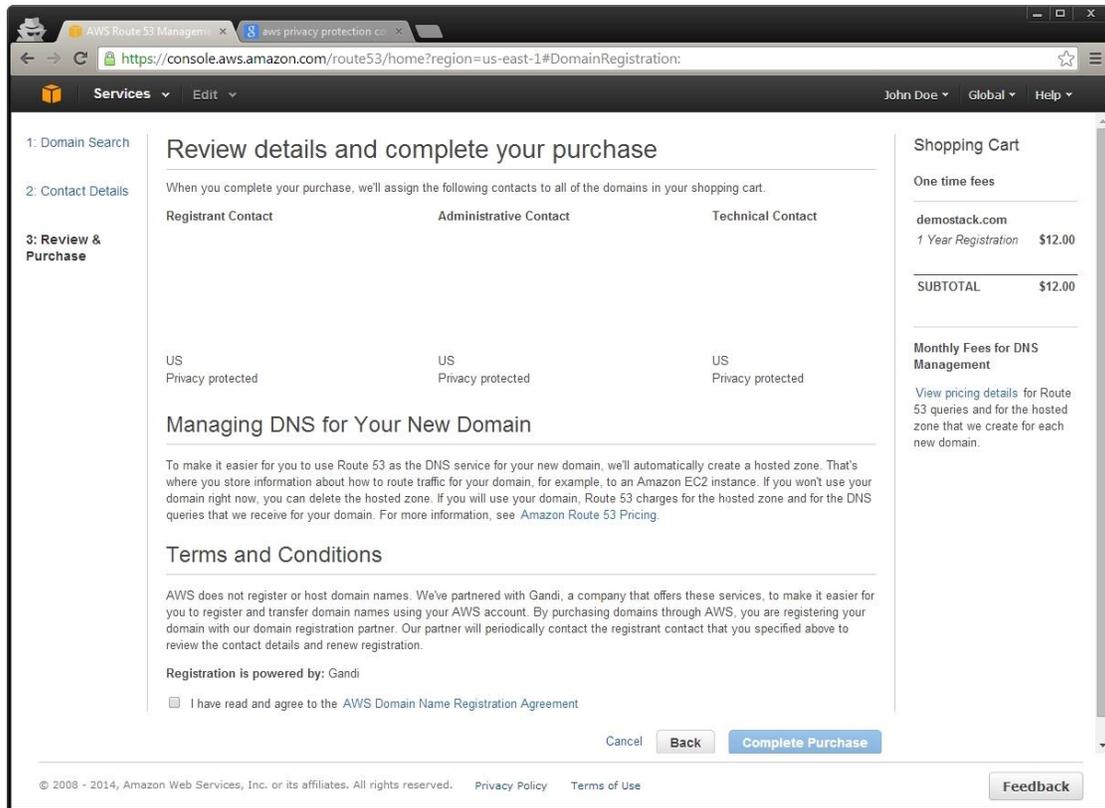
If the domain name you want is available, click **Add to cart**. Scroll to the bottom of the page and click **Continue**.



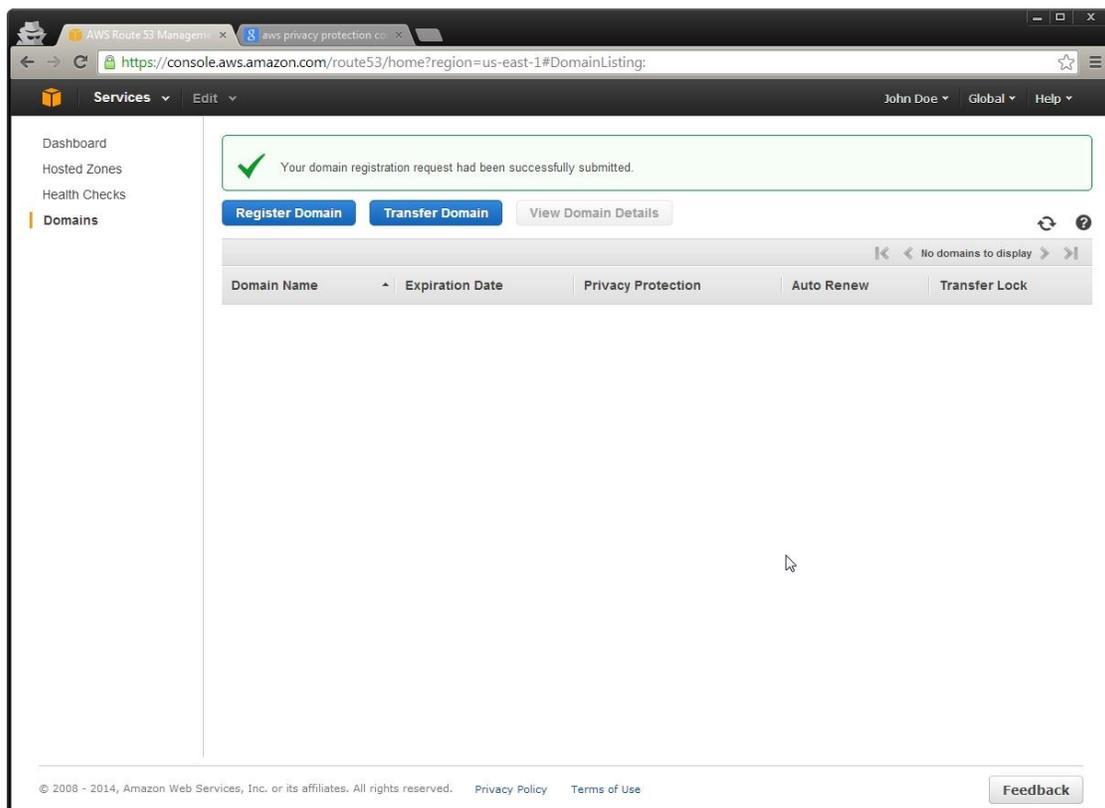
On the **Contact Details** page, fill in the contact information. You have the option to enable **Privacy Protection** on this page. Click **Continue**.



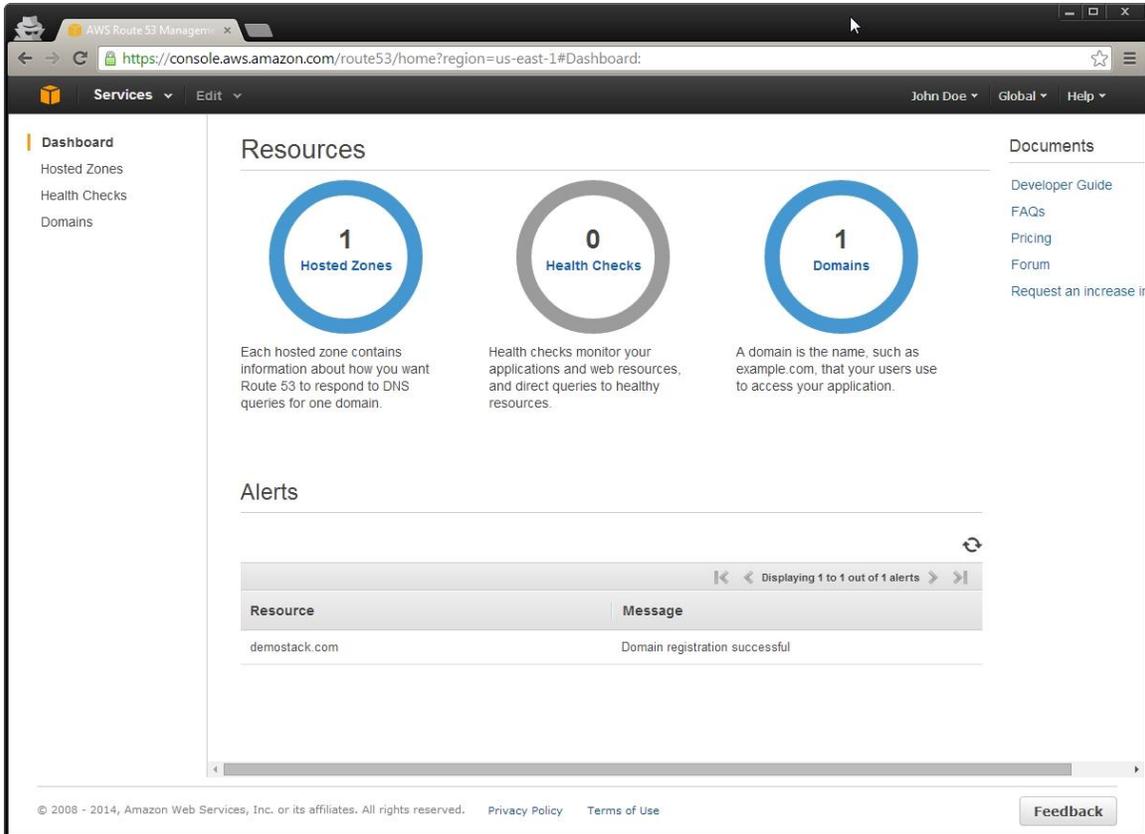
On the Review details and complete your purchase page, verify all your information and read the **AWS Domain Name Registration Agreement**. Once you are ready, click **Complete Purchase**.



On the **Domains** page, you should see a notice at the top that says **Your domain registration request has been successfully submitted**.



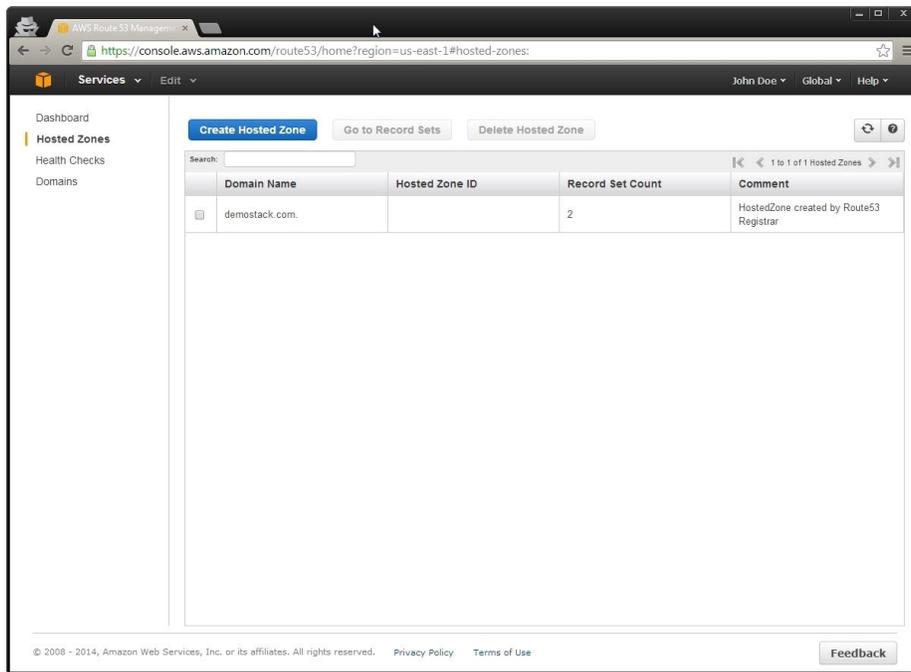
It may take a few minutes for the domain to appear in Route 53, but once it does, you should see the message, **Domain registration successful**.



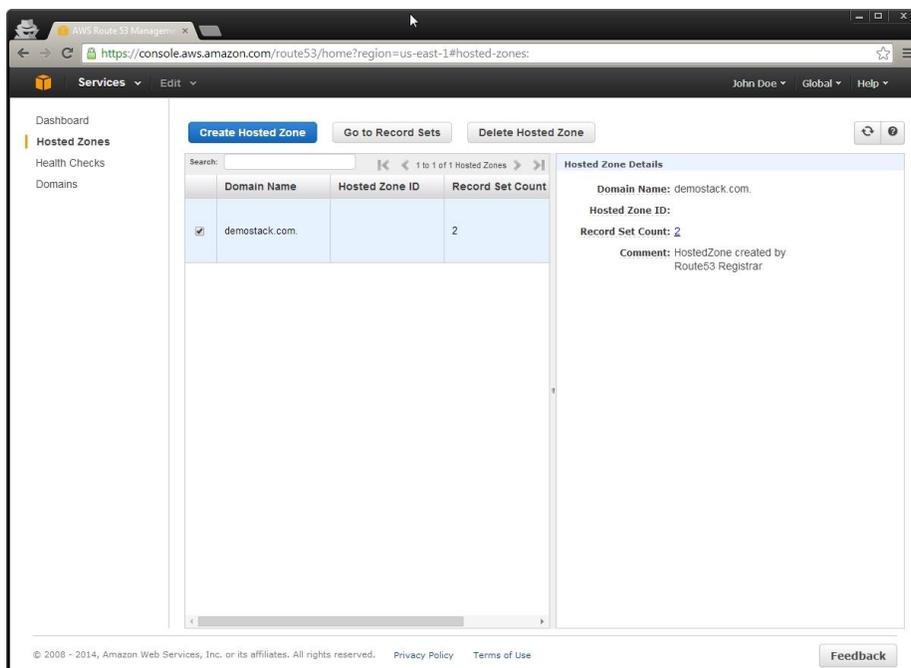
9 Set up DNS using Amazon Route 53

Once the domain name is available in Route 53, you can add different types of records. The most common records are name server (NS), address (A), and mail exchanger (MX) records. The NS records point to the server that holds all the other records. When you register a domain, the NS records are usually automatically set. The A records point a domain or subdomain to an IP address. The MX records point to your email server so services like Gmail and Hotmail know where to deliver your email.

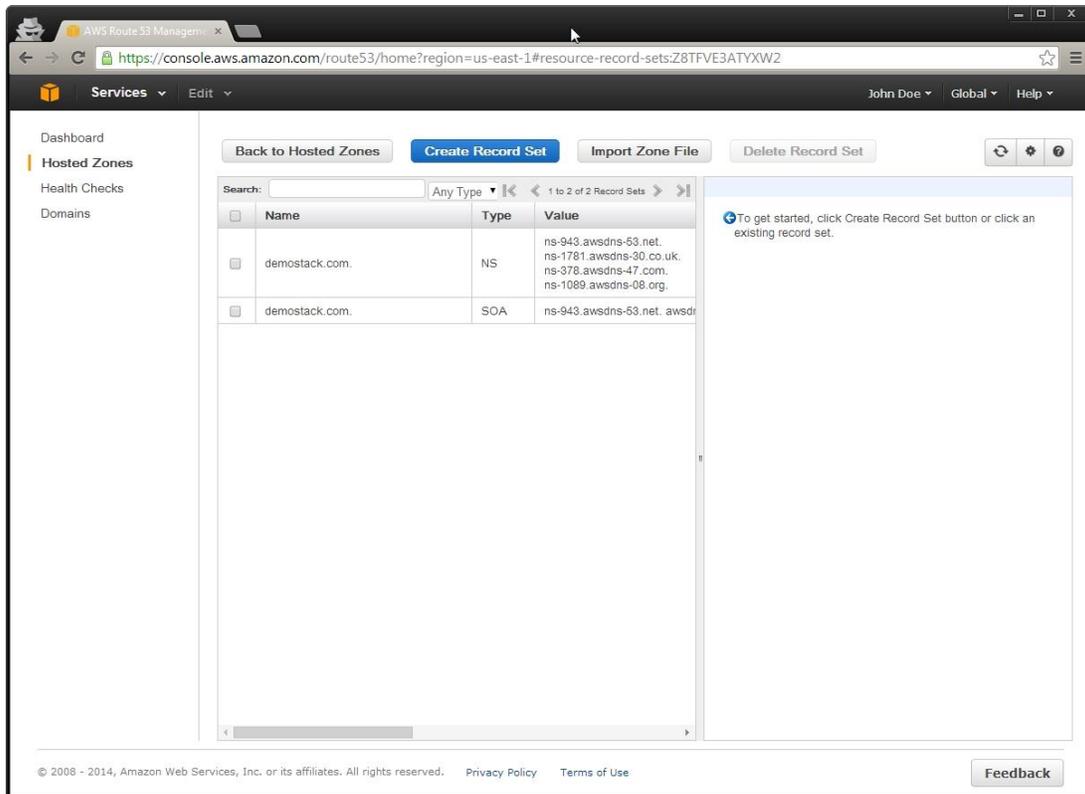
Navigate to the [AWS Management Console](#). Click on **Route 53**. To access the Route 53 service directly, open your web browser to <https://console.aws.amazon.com/route53>. Click **Hosted Zones** which is in left menu.



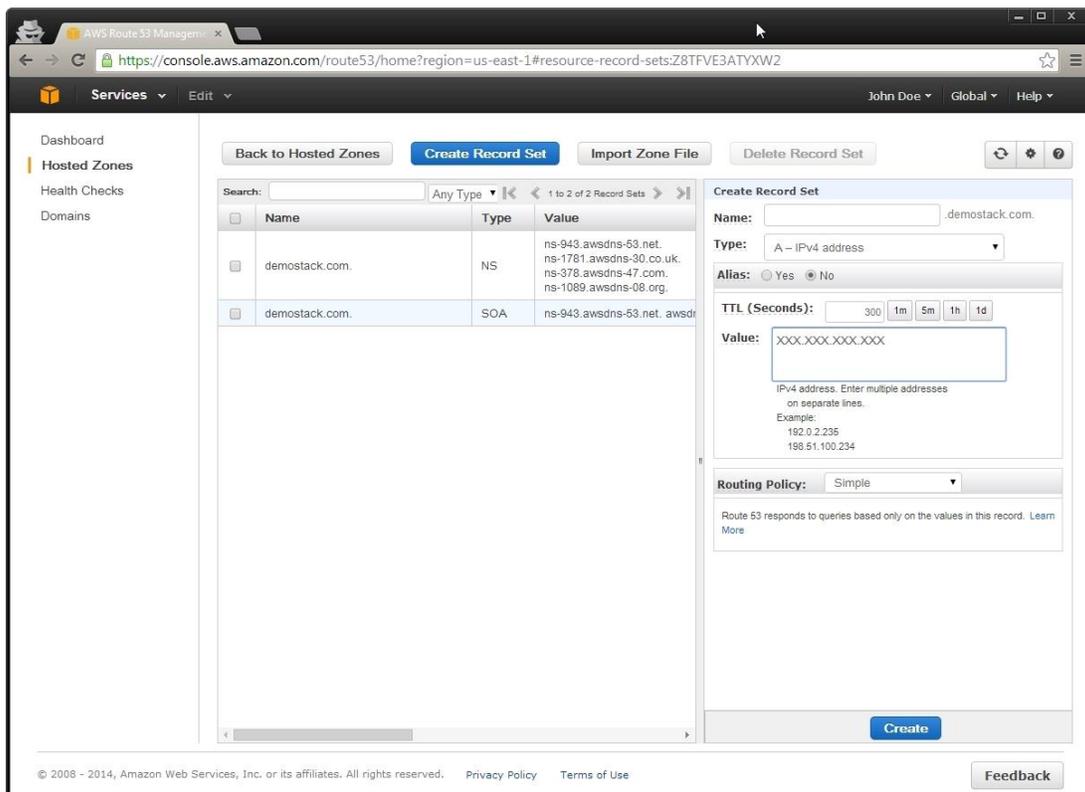
Click on your domain. Click **Go To Record Sets**.



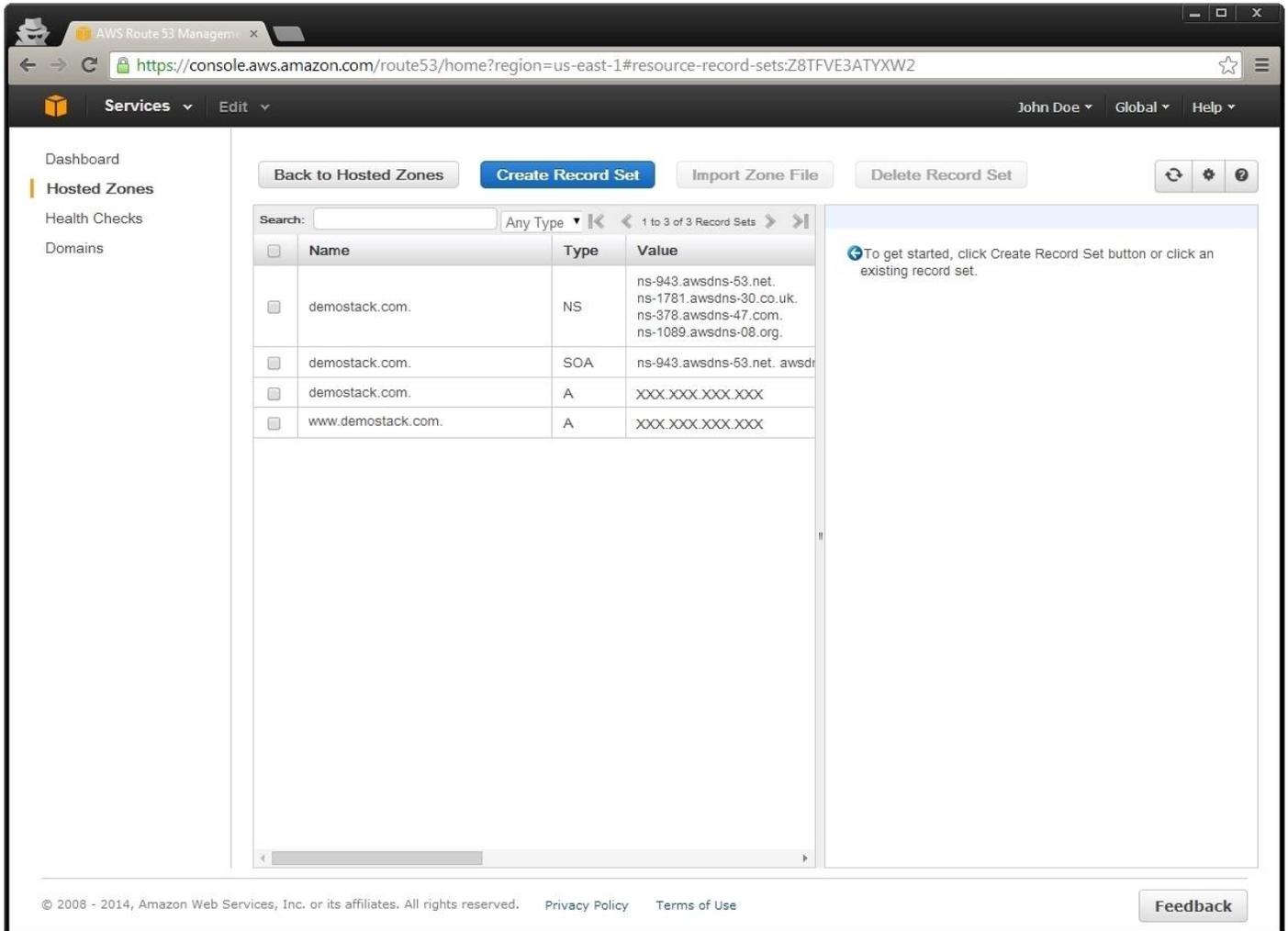
By default, your name server (NS) record points to Amazon which. Click **Create Record Set**.



Leave the **Name** textbox empty. Select **A – IPv4 addresses** in the **Type** dropdown. Fill in your server IP address into the **Value** text area. The IP address should be the Elastic IP assigned to the EC2 instance. Leave all the other settings at the defaults. Click **Create**.



You should see your new record. Repeat the same process again, but instead of leaving the name textbox empty, enter in **www**. That way, if someone goes to example.com or www.example.com, the same page will display. WordPress will take care of the redirection. In a few minutes, you should be able to type the domain name in your web browser and your website will appear.



10 Configure an Instance

Log in to the instance using PuTTY. You should see a shell prompt that allows you to type in commands. Your terminal will look a little different. The default shell for Ubuntu and many other Linux distributions is **Bash**.

Since this section consists of commands to type in, the guide will use this format:

Description of command:

Command to type in or paste

[Key to press]

<Screenshot of command before execution>

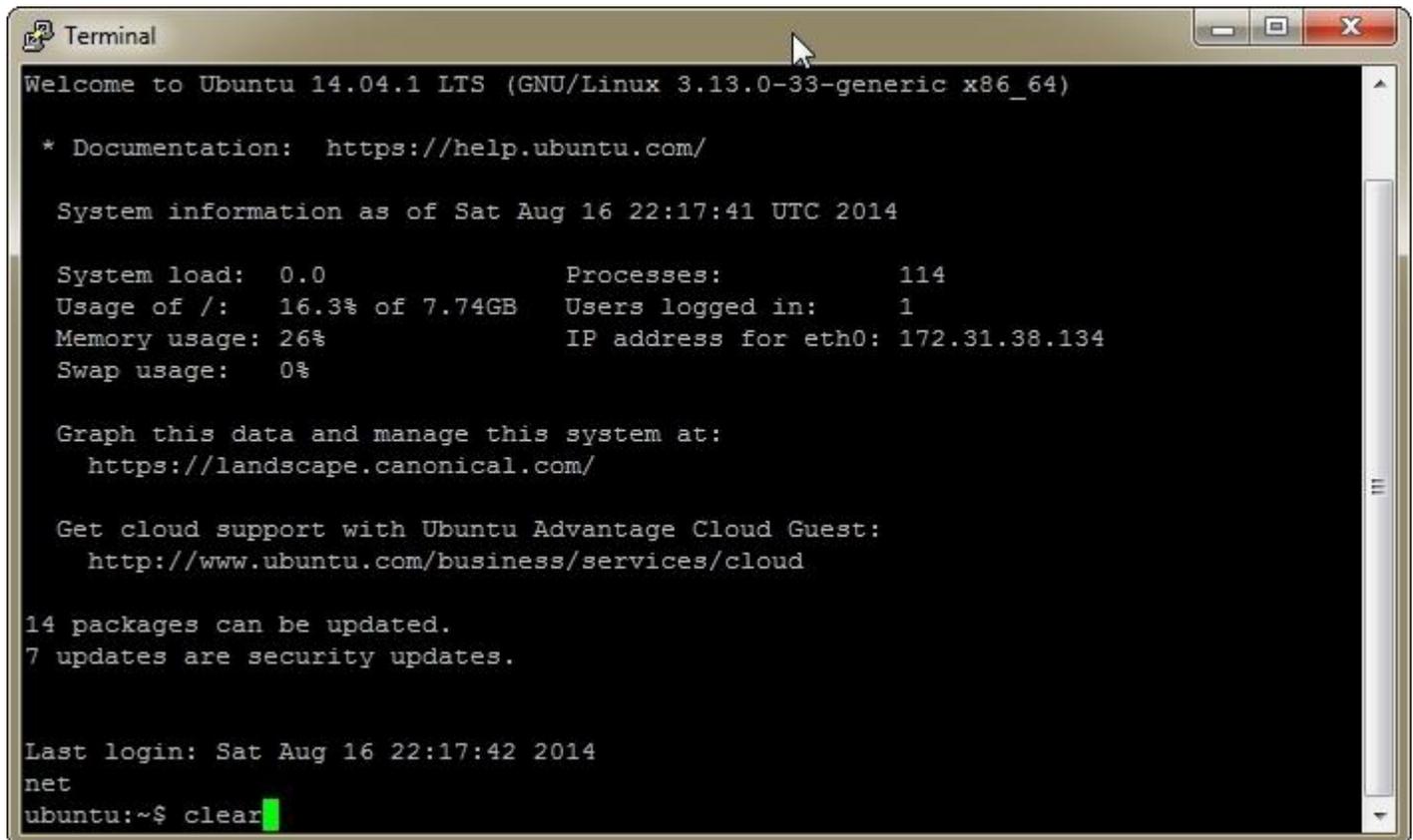
<Screenshot of command after execution>

10.1 Update the Software on an Instance

Let's clear the screen so it's easy to read:

`clear`

[Enter]



```
Terminal
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-33-generic x86_64)

* Documentation:  https://help.ubuntu.com/

System information as of Sat Aug 16 22:17:41 UTC 2014

System load:  0.0                Processes:           114
Usage of /:   16.3% of 7.74GB     Users logged in:    1
Memory usage: 26%                IP address for eth0: 172.31.38.134
Swap usage:   0%

Graph this data and manage this system at:
  https://landscape.canonical.com/

Get cloud support with Ubuntu Advantage Cloud Guest:
  http://www.ubuntu.com/business/services/cloud

14 packages can be updated.
7 updates are security updates.

Last login: Sat Aug 16 22:17:42 2014
net
ubuntu:~$ clear
```



```
Terminal
ubuntu:~$
```

Next, we'll update the package lists from the repositories so the package manager has access to the latest software. The `sudo` command tells the shell to run the command as a superuser or a user with administrative privileges. The `apt-get` command is a command line interface for downloading and installing packages.

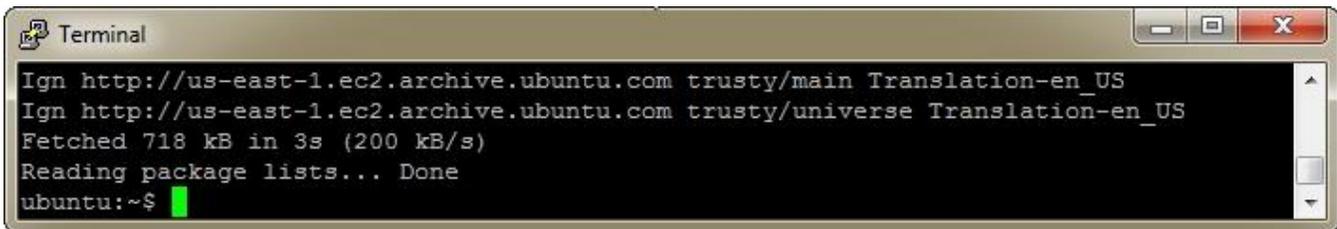
Update the package lists from the repositories:

```
sudo apt-get update
```

[Enter]



```
Terminal
ubuntu:~$ sudo apt-get update
```



```
Terminal
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/main Translation-en_US
Ign http://us-east-1.ec2.archive.ubuntu.com trusty/universe Translation-en_US
Fetched 718 kB in 3s (200 kB/s)
Reading package lists... Done
ubuntu:~$
```

Update all the installed packages (applications):

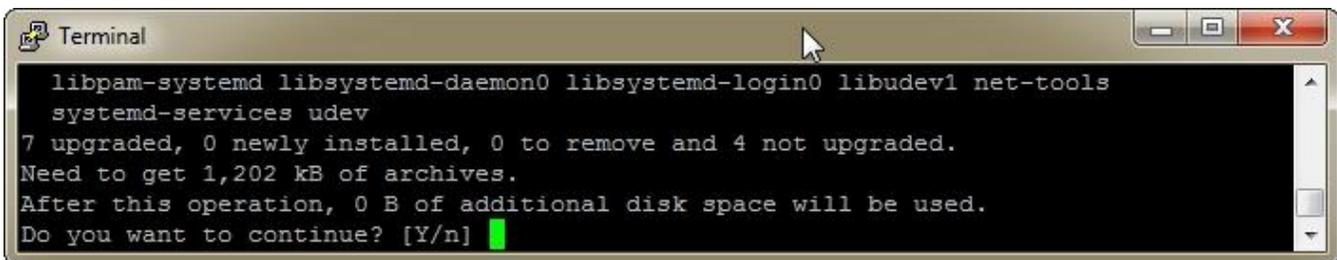
```
sudo apt-get upgrade
```

[Enter]

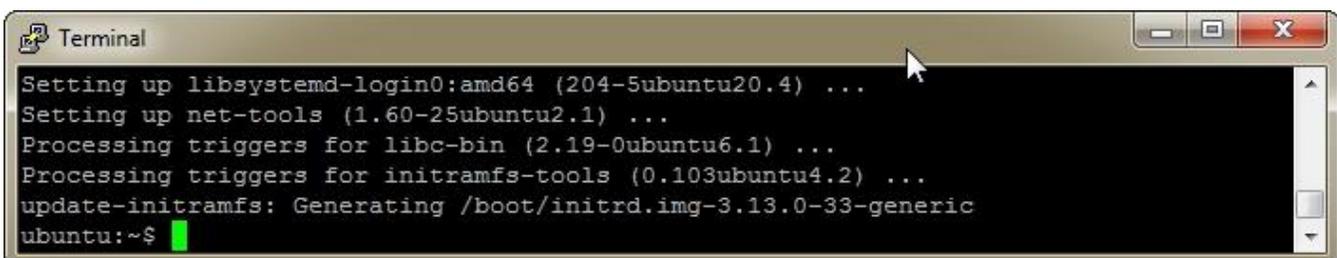


```
Terminal
ubuntu:~$ sudo apt-get upgrade
```

Press **Y** to upgrade.



```
Terminal
libpam-systemd libsystemd-daemon0 libsystemd-login0 libudev1 net-tools
systemd-services udev
7 upgraded, 0 newly installed, 0 to remove and 4 not upgraded.
Need to get 1,202 kB of archives.
After this operation, 0 B of additional disk space will be used.
Do you want to continue? [Y/n]
```



```
Terminal
Setting up libsystemd-login0:amd64 (204-5ubuntu20.4) ...
Setting up net-tools (1.60-25ubuntu2.1) ...
Processing triggers for libc-bin (2.19-0ubuntu6.1) ...
Processing triggers for initramfs-tools (0.103ubuntu4.2) ...
update-initramfs: Generating /boot/initrd.img-3.13.0-33-generic
ubuntu:~$
```

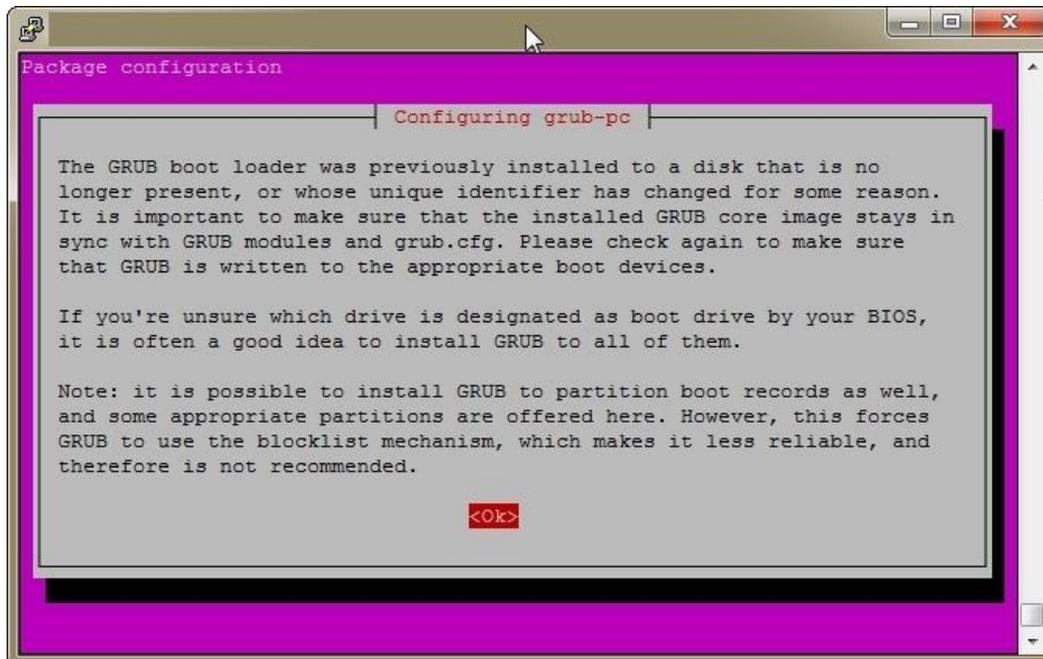
10.2 Updating GRUB on an Instance

If GRUB is updated when you run `sudo apt-get upgrade` and the **Configuring grub-pc** screen appears, follow the steps below to update it correctly. If you don't see the screen, you can skip this section.

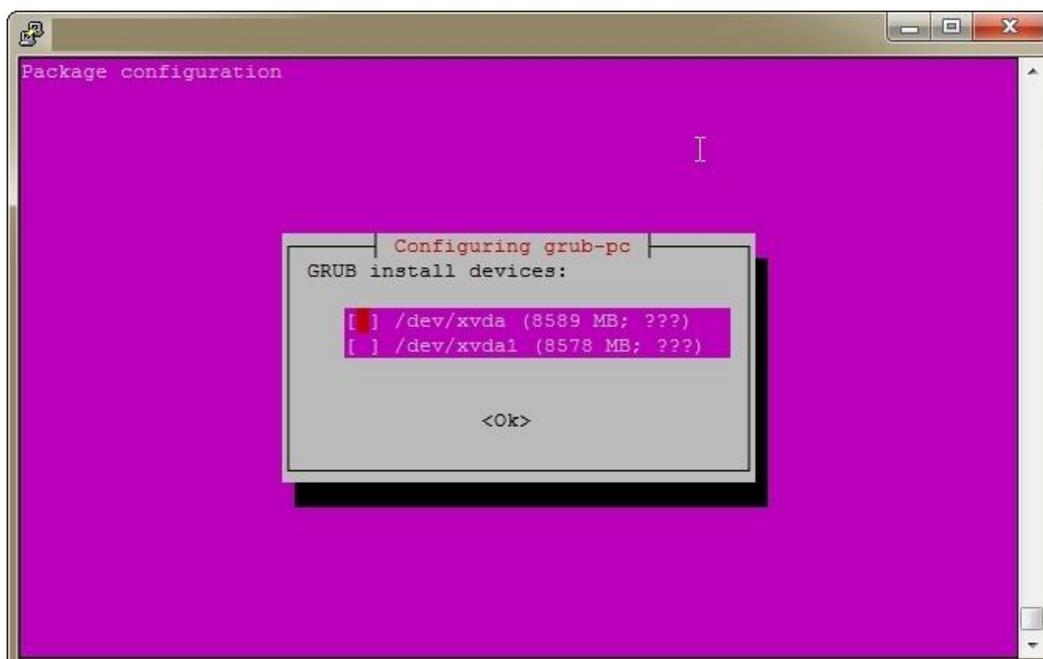
GRUB stands for GRand Unified Bootloader. It is the piece of software that tells the computer to load an operating systems. In this case, it tells the instance to load Ubuntu Linux. If GRUB is not configured correctly, the instance will not boot correctly and you will not be able to access the instance.

Use **Space** to select options. Use **Tab** to move between controls. Use **Enter** to execute controls.

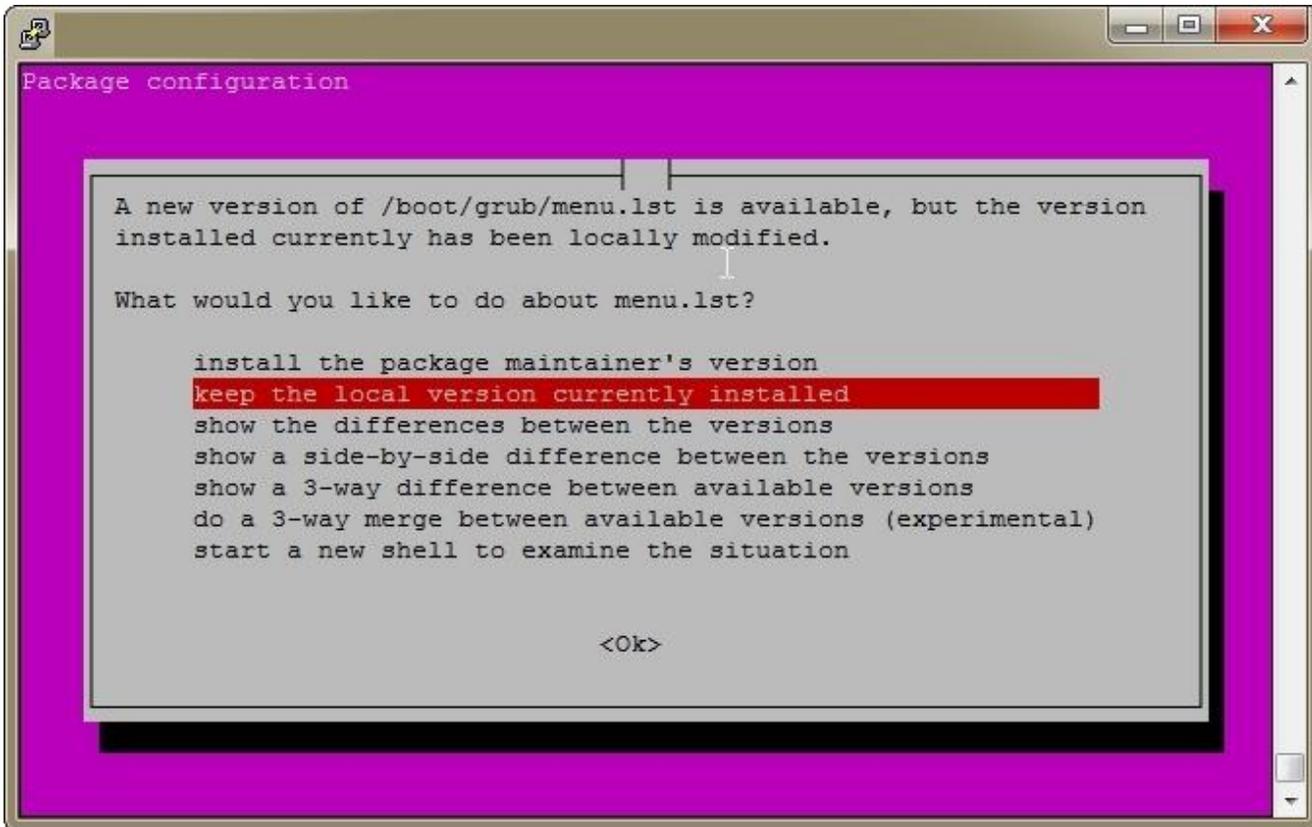
On the **Configuring grub-pc** screen, press **Tab** and then press **Enter**.



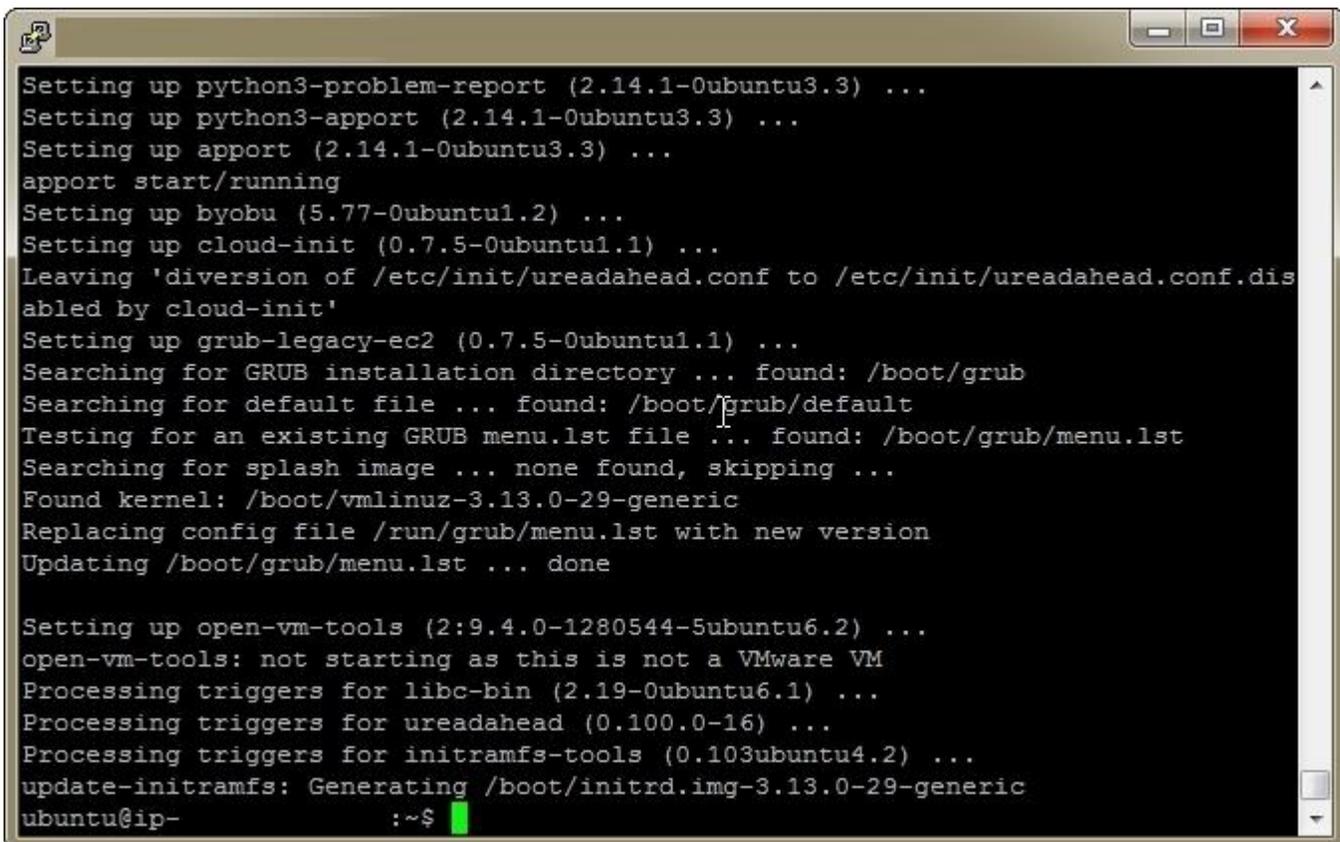
On the GRUB install devices screen, press **Space** to select the top option. Press **Tab** and then press **Enter**.



On the **Menu Selection** screen, press **Arrow Up** to select **install the package maintainer's version**. Press **Tab** and then press **Enter**.



Once installation completes, you should see the **update-initramfs** command at the bottom of the screen.



10.3 Reboot an Instance

One of the nice features of Linux is the ability to run for long periods of time without requiring a restart. There are Linux servers that have run over a year without a single reboot. It is a good practice to ensure the instance reboots successfully before installing additional software.

Let's reboot the instance:

```
sudo reboot
```

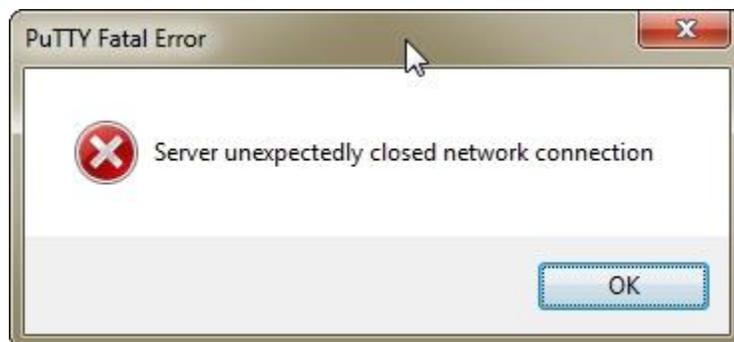
[Enter]



You will see the message, **The system is going down for reboot NOW!**



You will then see the **PuTTY Fatal Error** dialog that says, **Server unexpectedly closed network connection**. The message is completely normal and expected. Click **OK** and then close the inactive PuTTY terminal.



After 30 seconds, run PuTTY.exe again and log in to verify the machine rebooted successfully.

10.4 Install MySQL Database

MySQL is one of the most popular relational database management systems. MySQL is free, open-source software and is extremely fast. WordPress requires and only supports MySQL.

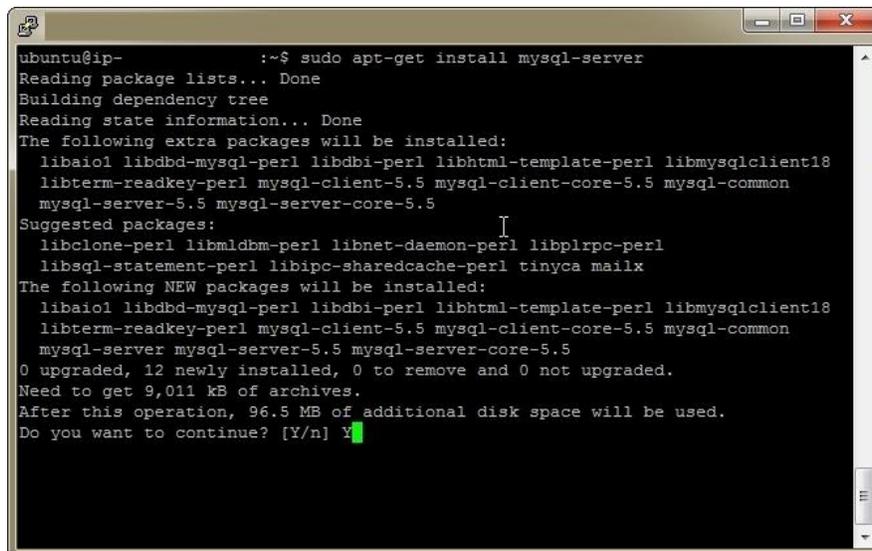
Install MySQL:

```
sudo apt-get install mysql-server
```

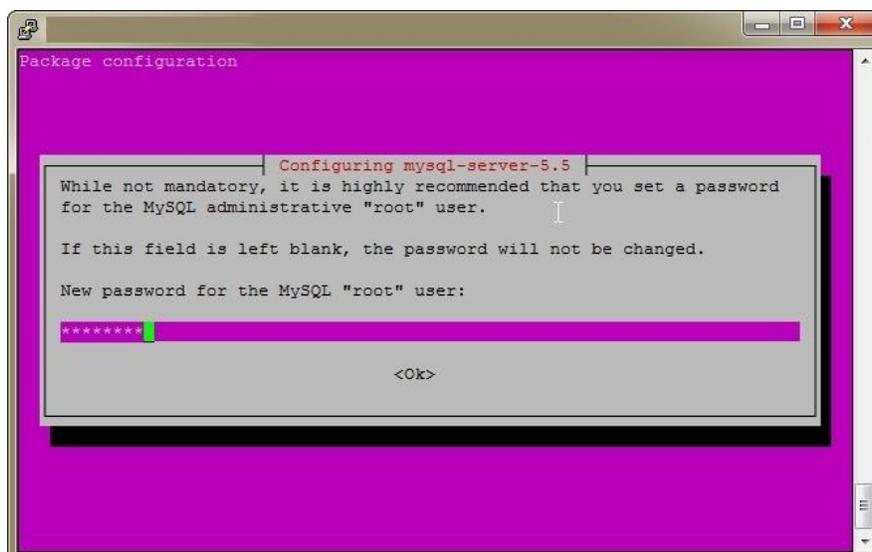
[Enter]



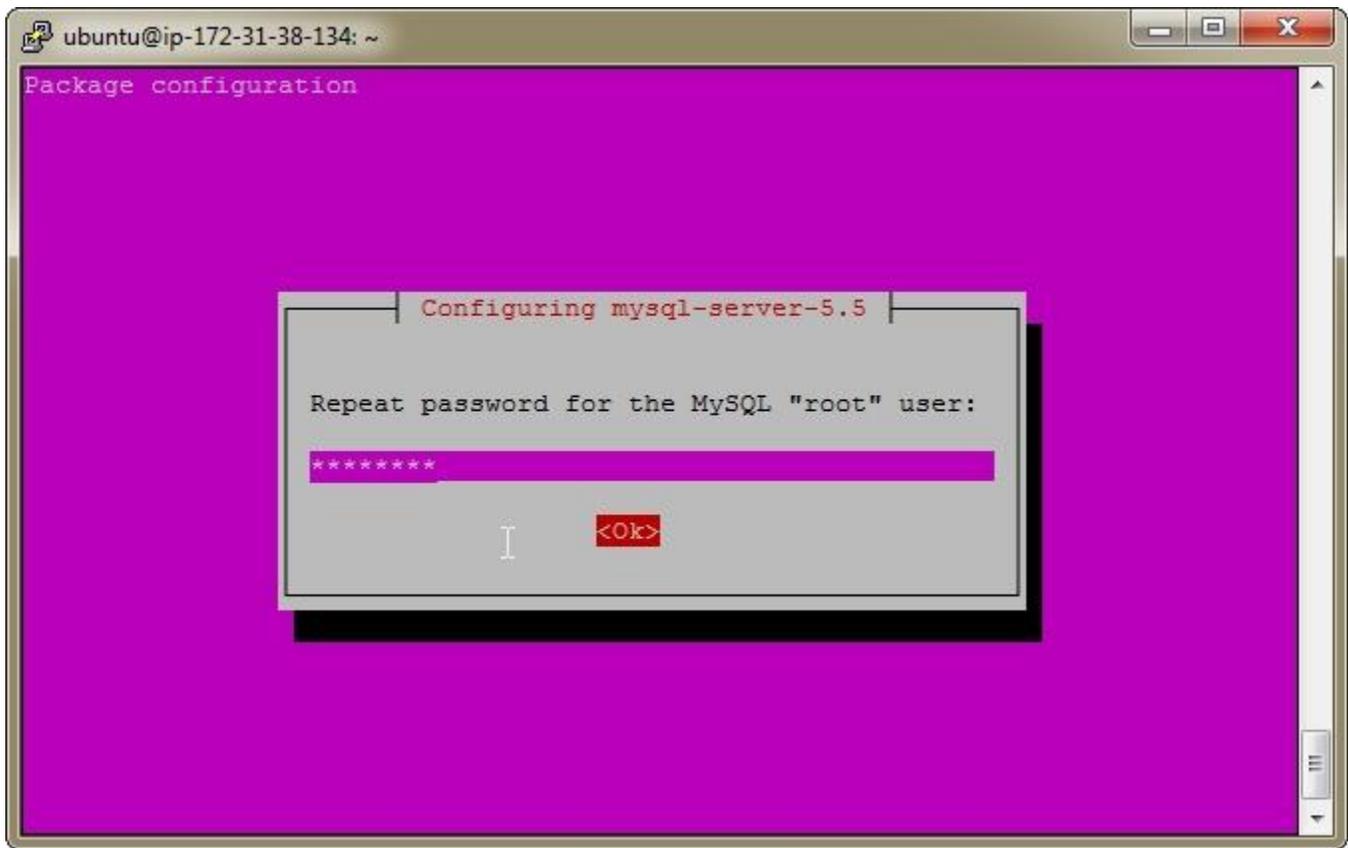
Press **Y** to install.



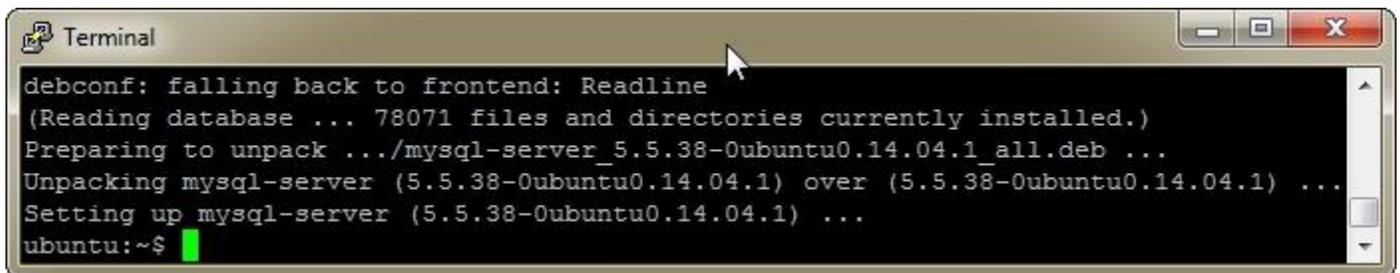
On the **Configuring mysql-server-5.5** screen, type in a password for the root account. Press **Tab** and then press **Enter**.



On the next screen, type in the same password again for the root account. Press **Tab** and then press **Enter**.



MySQL installation should finish successfully.



10.5 Install PHP and Apache HTTP Server

PHP is a server-side scripting language that functions as both a web language and a scripting language. WordPress is written primarily in PHP along with HTML, CSS, and JavaScript.

Apache HTTP Server is one of the most widely used web servers.

Install PHP 5 and it should install Apache:

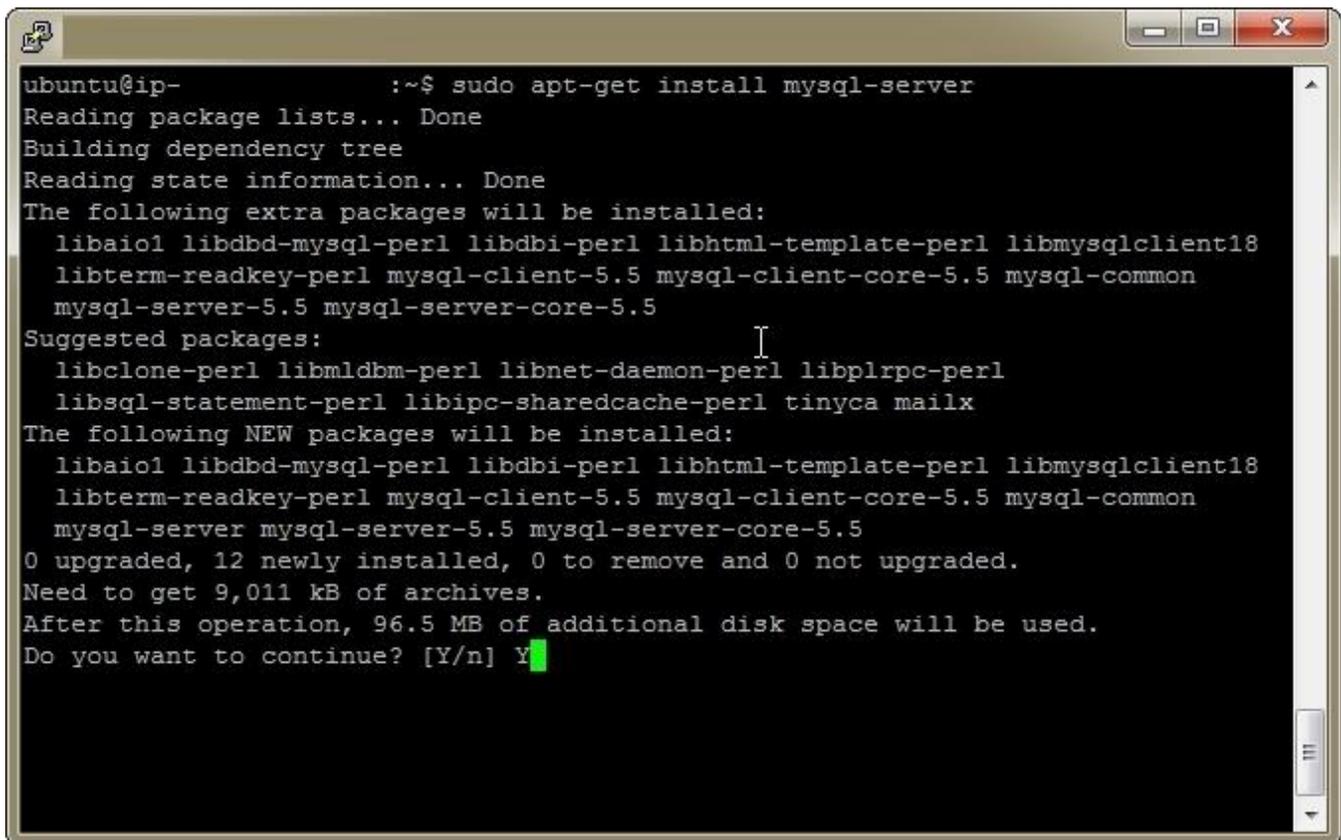
```
sudo apt-get install php5
```

[Enter]



```
Terminal
ubuntu:~$ sudo apt-get install php5
```

Press **Y** to install.



```
ubuntu@ip-:~$ sudo apt-get install mysql-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
 libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
 libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
 mysql-server-5.5 mysql-server-core-5.5
Suggested packages:
 libclone-perl libmldbm-perl libnet-daemon-perl libplrpc-perl
 libsql-statement-perl libipc-sharedcache-perl tinyca mailx
The following NEW packages will be installed:
 libaio1 libdbd-mysql-perl libdbi-perl libhtml-template-perl libmysqlclient18
 libterm-readkey-perl mysql-client-5.5 mysql-client-core-5.5 mysql-common
 mysql-server mysql-server-5.5 mysql-server-core-5.5
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 9,011 kB of archives.
After this operation, 96.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

Enable the `mod_rewrite` module in Apache. WordPress requires the module so it can use pretty permalinks. You can see more information in the WordPress Codex: http://codex.wordpress.org/Using_Permalinks

```
sudo a2enmod rewrite
```

[Enter]



```
Terminal
ubuntu:~$ sudo a2enmod rewrite
```



```
Terminal
ubuntu:~$ sudo a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  service apache2 restart
ubuntu:~$
```

Restart Apache to activate the new configuration:

```
sudo service apache2 restart
```

[Enter]

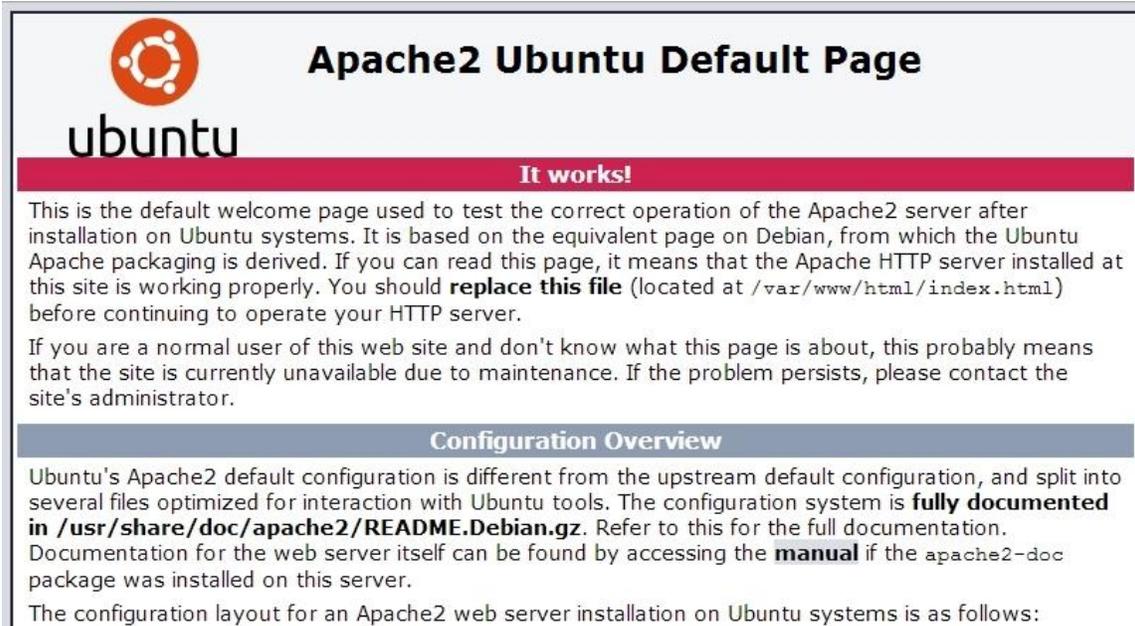


```
Terminal
ubuntu:~$ sudo service apache2 restart
```



```
Terminal
ubuntu:~$ sudo service apache2 restart
* Restarting web server apache2
ubuntu:~$ [ OK ]
```

Open your web browser to the Elastic IP (public IP) of the instance and you should see the **Apache 2 Ubuntu Default Page**.



The screenshot shows the Apache2 Ubuntu Default Page. At the top left is the Ubuntu logo (a red circle with white dots) and the word "ubuntu" in a bold, lowercase font. To the right of the logo is the title "Apache2 Ubuntu Default Page" in a bold, black font. Below the title is a red horizontal bar with the text "It works!" in white. The main content area contains two paragraphs of text. The first paragraph explains that this is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It mentions that the page is based on the equivalent page on Debian and that if you can read this page, it means the Apache HTTP server is working properly. It also notes that you should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server. The second paragraph explains that if you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. It suggests contacting the site's administrator. Below the text is a blue horizontal bar with the title "Configuration Overview" in white. The final paragraph explains that Ubuntu's Apache2 default configuration is different from the upstream default configuration and is split into several files optimized for interaction with Ubuntu tools. It states that the configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz and refers to this for the full documentation. It also mentions that documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed on this server. The last line of the screenshot states: "The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:"

11 Transfer Files to an Instance

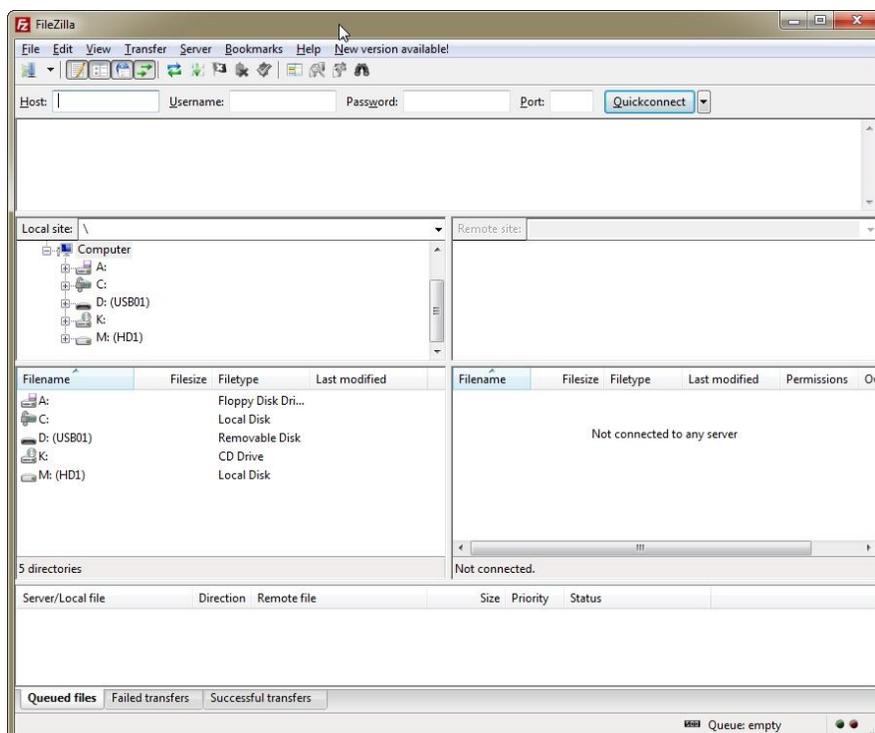
SFTP stands for Secure File Transfer Protocol. It is a secure method to transfer and manage files. It's an alternative to FTP which is insecure since it transfers data in plaintext that is not encrypted. FileZilla, a free SFTP client, will be used for this tutorial.

11.1 Connect to Instance using FileZilla

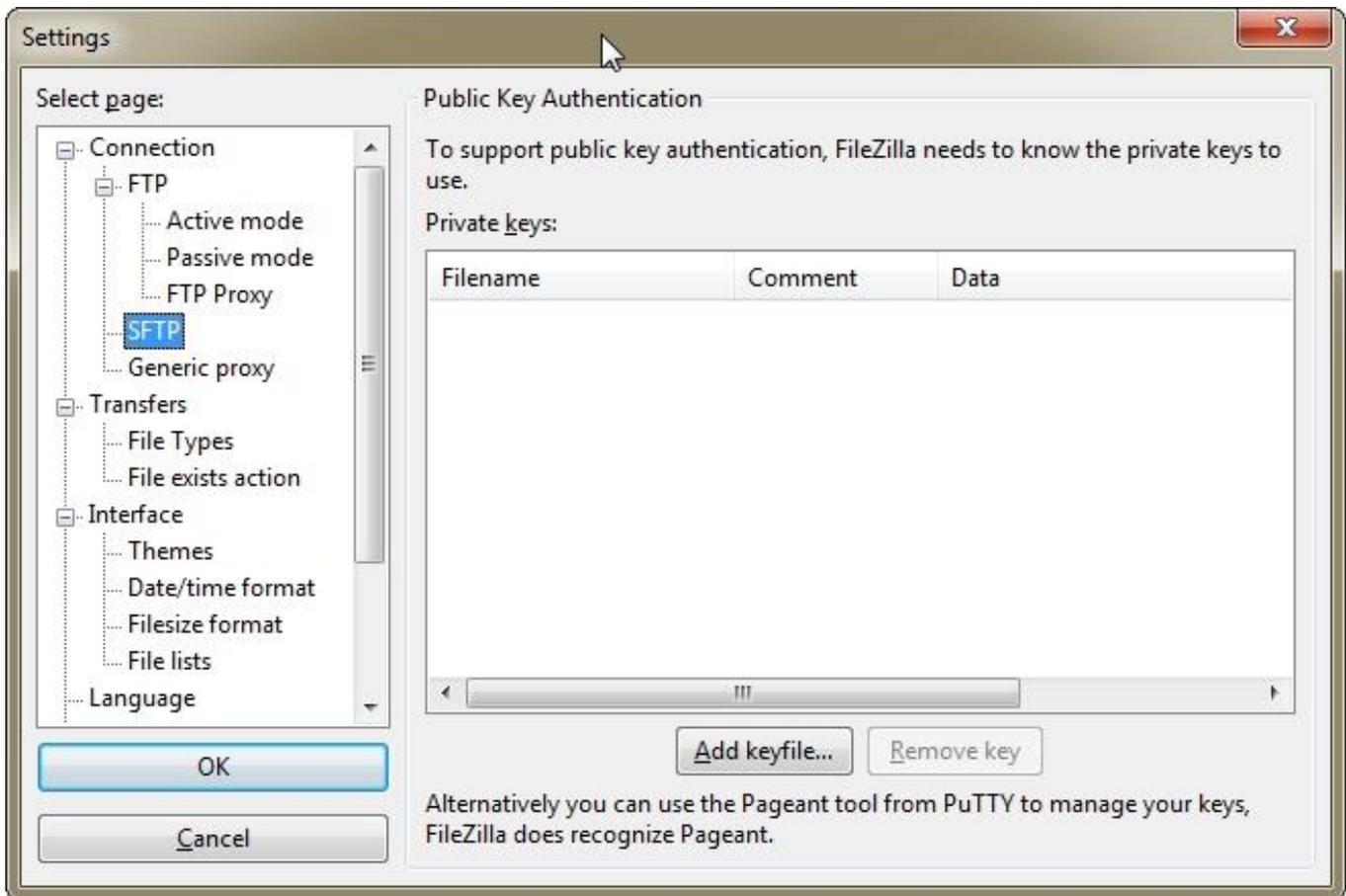
Open your web browser to <https://filezilla-project.org/download.php?type=client> and click **Download Now**.



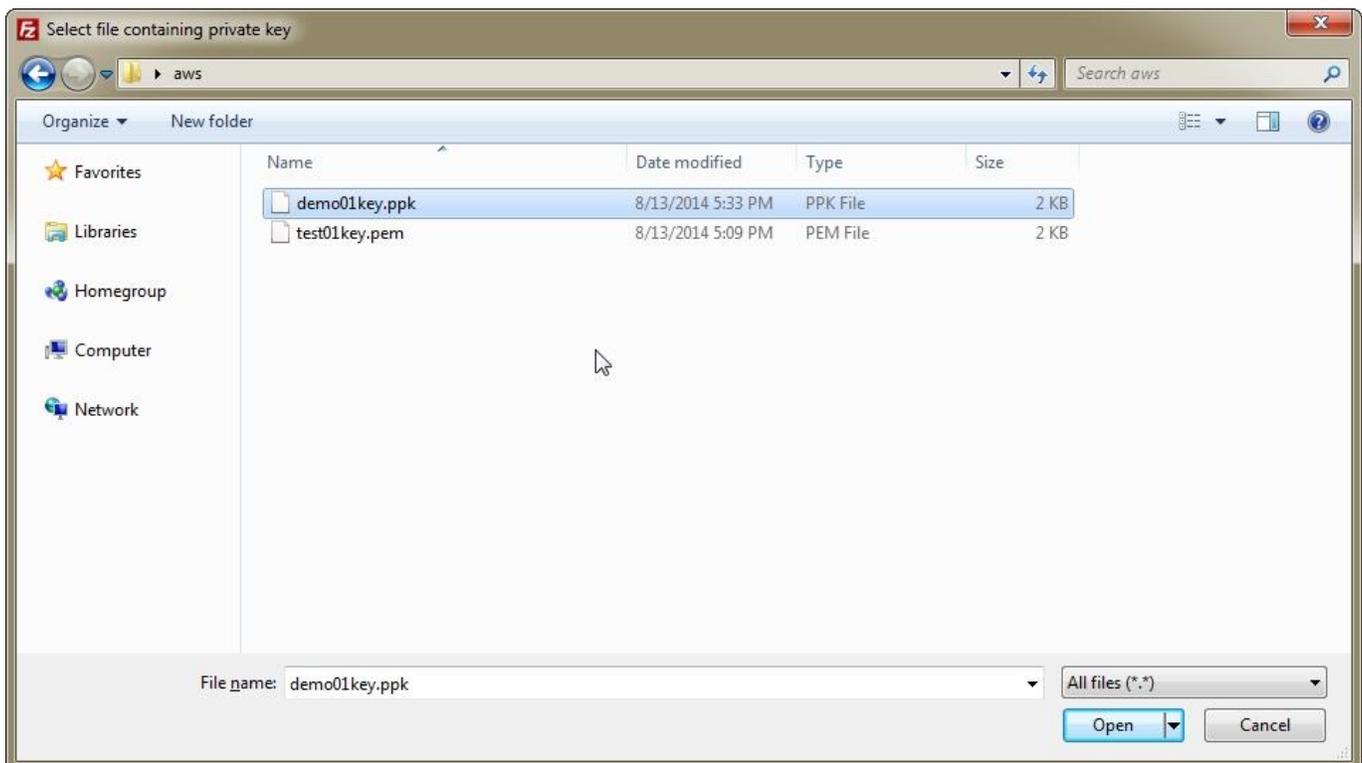
Install FileZilla and then run filezilla.exe. You should see the FileZilla application appear. Click **File** -> **Settings**. Navigate to **Connection** -> **SFTP**.



Click **Add keyfile**.



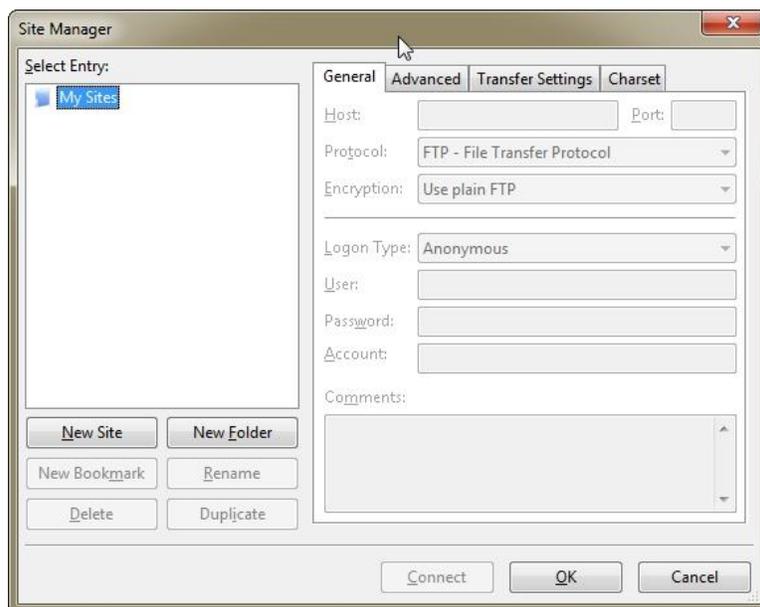
On the **Select file containing private key** dialog, select the .PPK private key and click **Open**.



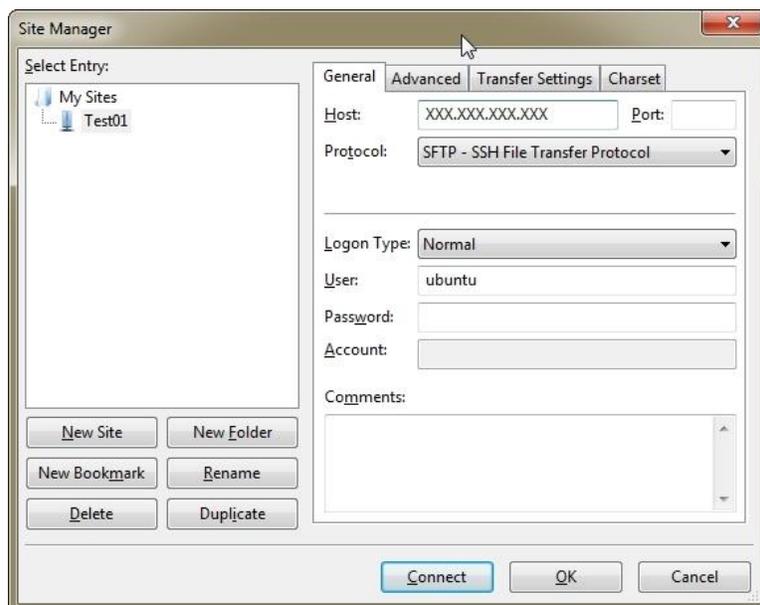
If you receive a **Convert keyfile** dialog, click **Yes** to convert the key into a file without a password.



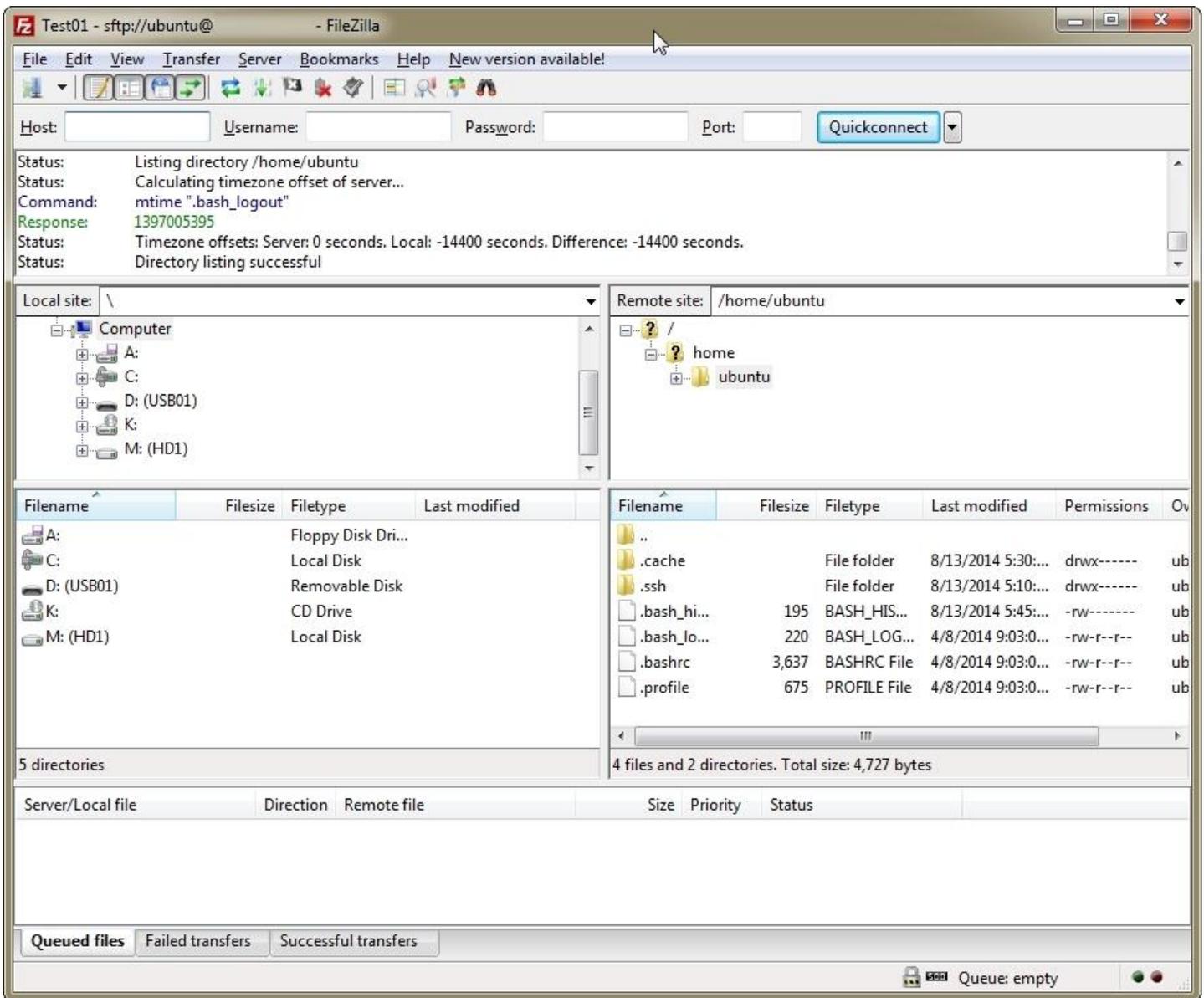
Click **OK** to close the **Setting** dialog. Click **File- > Site Manager**. On the **Site Manager** form, click **New Site**.



In the **Host** textbox, fill in your Elastic IP (public IP). Select **SFTP – SSH File Transfer Protocol** from the **Protocol** dropdown. Select **Normal** from the **Logon Type** dropdown. Type **ubuntu** in the **User** textbox. Click **Connect**.



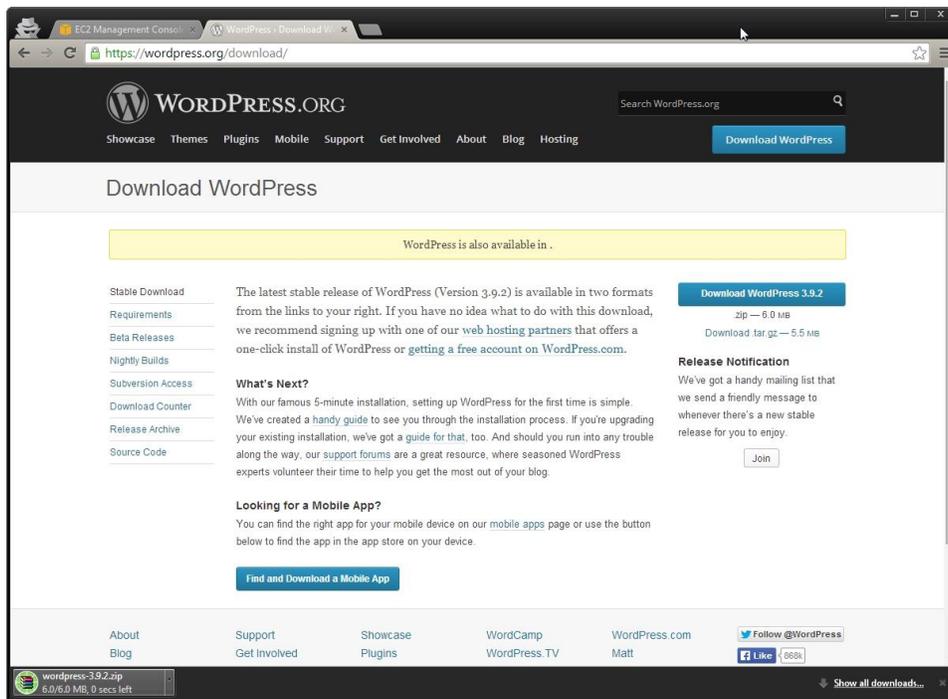
You should see the **Remote Site** panel on the right side with the server folders at **/home/Ubuntu**.



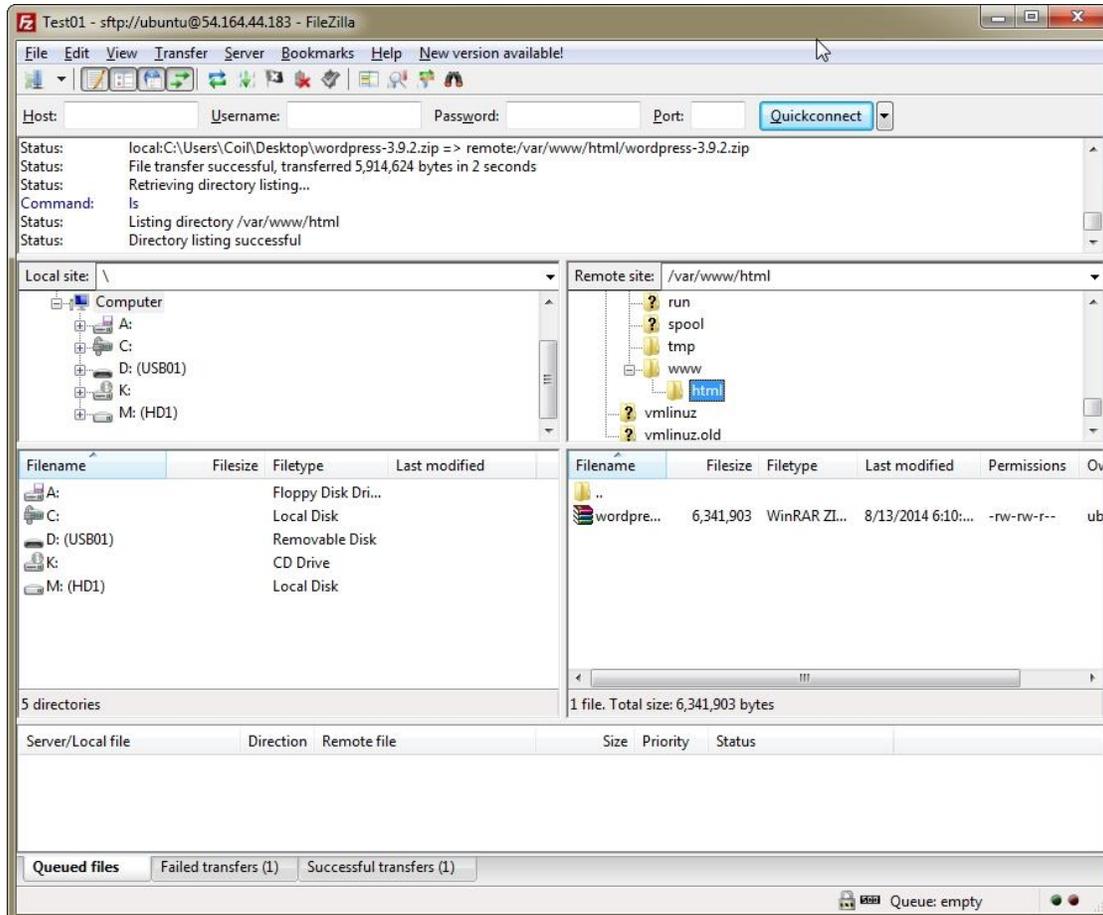
You can now transfer files between your computer (left panel) and the instance (right panel).

11.2 Transfer WordPress Files to an Instance

Open your web browser to <https://wordpress.org/download/> and click **Download WordPress**.



In FileZilla, navigate to `/var/www/html` in the **Remote Site** panel on the right side. Locate the WordPress zip file you downloaded and drag it into the **Remote Site** panel to transfer the file to the instance.



11.3 Extract WordPress Files to an Instance

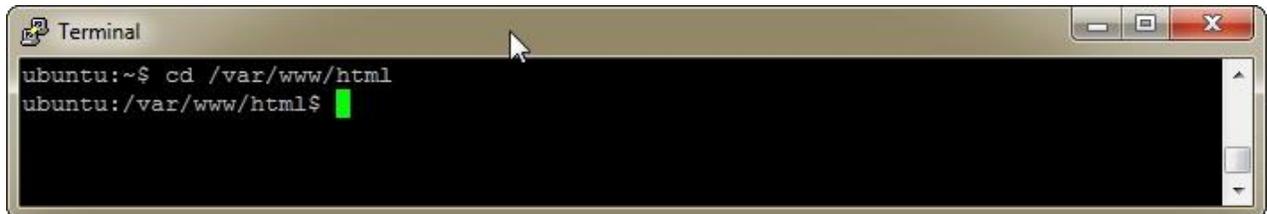
Change to the Apache html directory:

```
cd /var/www/html
```

[Enter]



```
Terminal
ubuntu:~$ cd /var/www/html
```

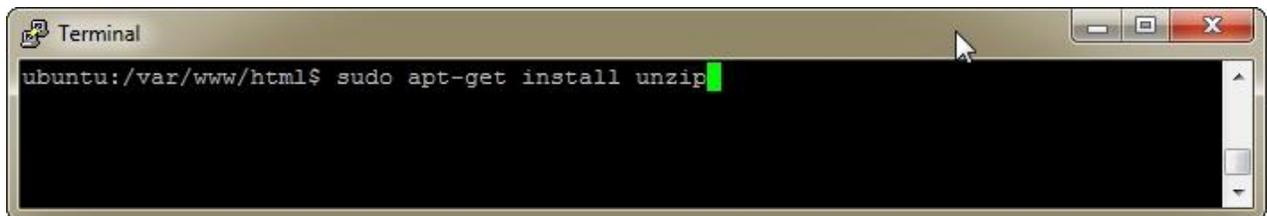


```
Terminal
ubuntu:~$ cd /var/www/html
ubuntu:/var/www/html$
```

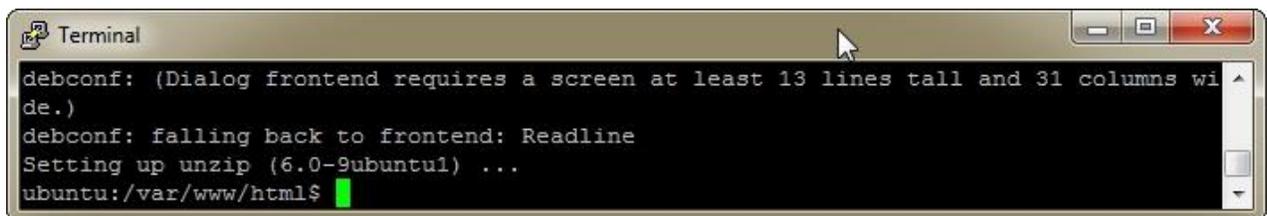
Install Unzip:

```
sudo apt-get install unzip
```

[Enter]



```
Terminal
ubuntu:/var/www/html$ sudo apt-get install unzip
```

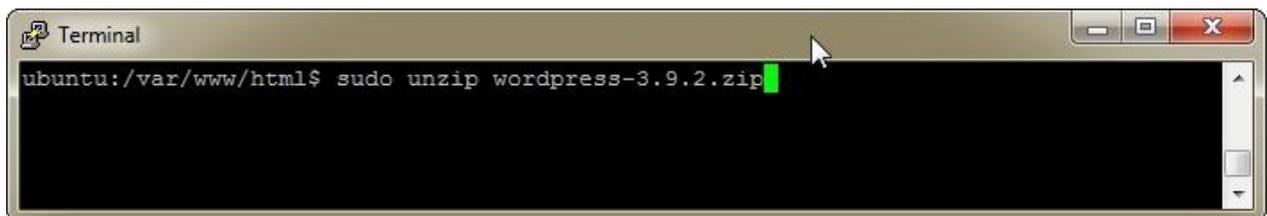


```
Terminal
debconf: (Dialog frontend requires a screen at least 13 lines tall and 31 columns wide.)
debconf: falling back to frontend: Readline
Setting up unzip (6.0-9ubuntu1) ...
ubuntu:/var/www/html$
```

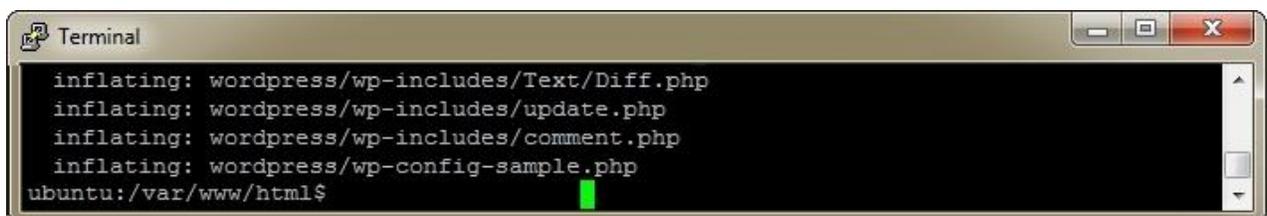
Extract the Wordpress files:

```
sudo unzip wordpress-3.9.2.zip
```

[Enter]



```
Terminal
ubuntu:/var/www/html$ sudo unzip wordpress-3.9.2.zip
```



```
Terminal
inflating: wordpress/wp-includes/Text/Diff.php
inflating: wordpress/wp-includes/update.php
inflating: wordpress/wp-includes/comment.php
inflating: wordpress/wp-config-sample.php
ubuntu:/var/www/html$
```

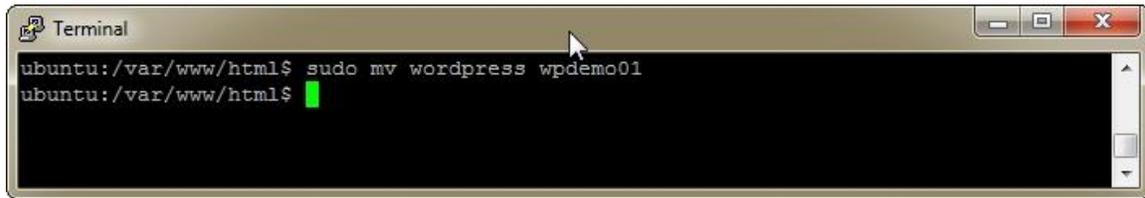
Rename the Wordpress folder to a name of your choice:

```
sudo mv wordpress wpdemo01
```

[Enter]



```
Terminal
ubuntu:/var/www/html$ sudo mv wordpress wpdemo01
```

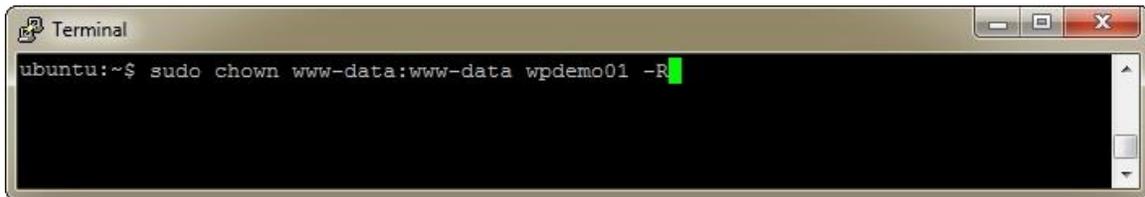


```
Terminal
ubuntu:/var/www/html$ sudo mv wordpress wpdemo01
ubuntu:/var/www/html$
```

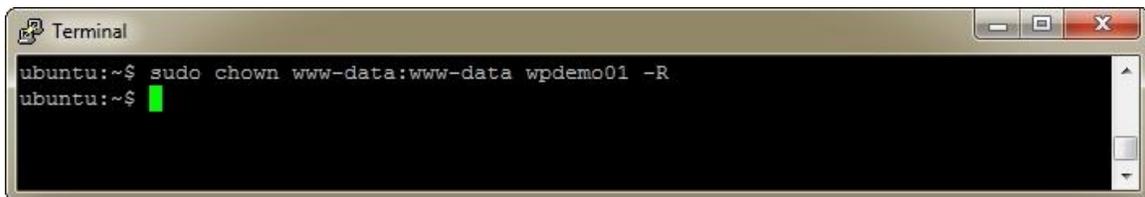
Change the ownership of the files so Apache can access them:

```
sudo chown www-data:www-data wpdemo01 -R
```

[Enter]



```
Terminal
ubuntu:~$ sudo chown www-data:www-data wpdemo01 -R
```



```
Terminal
ubuntu:~$ sudo chown www-data:www-data wpdemo01 -R
ubuntu:~$
```

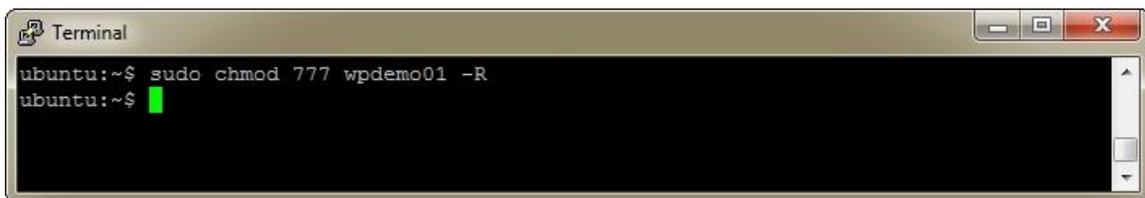
Change permissions of the files temporarily. Be sure to update the permissions noted in the **Final Notes** section once WordPress is configured:

```
sudo chmod 777 wpdemo01 -R
```

[Enter]



```
Terminal
ubuntu:~$ sudo chmod 777 wpdemo01 -R
```



```
Terminal
ubuntu:~$ sudo chmod 777 wpdemo01 -R
ubuntu:~$
```

The WordPress files are now located at `/var/www/html/wpdemo01`.

12 Configure Apache Web Server

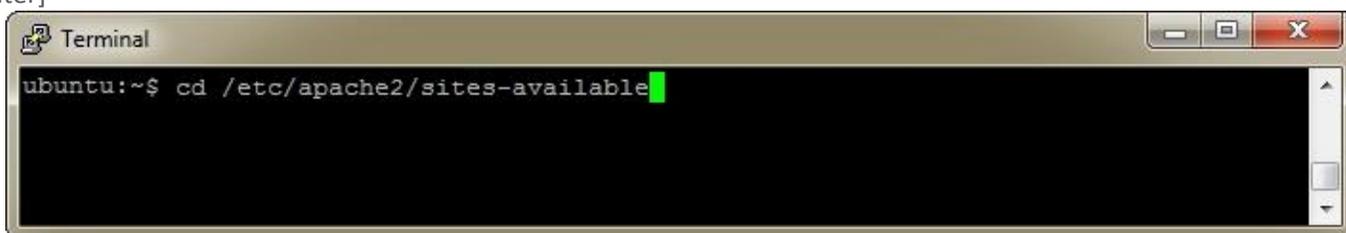
By default, Apache serves files that are located in the `/var/www/html` directory to any request whether it be for the public IP address of the server (ex. 172.16.0.0) or by the A record in DNS (`www.example.com`) that points to the public IP address. Apache has a feature called virtual hosts which allows running multiple websites from Apache at the same time. There are name-based virtual hosts and IP-based virtual hosts. Name-based virtual hosts allow for more than one website per IP address which is very common and utilized by many of the shared hosting companies since IPv4 addresses are so scarce. IP-based virtual hosts provides an IP address for a single website. In this guide, you'll disable the default website and create a new default configuration file. This will enable you to add more websites in the future.

Apache stores all of the **available** website configuration files in the `/etc/apache2/sites-available` directory and all the enabled website configuration files in the `/etc/apache2/sites-enabled` directory. The files in the `sites-enabled` directory are actually symlinks or pointers to the files in the `/etc/apache2/sites-available` directory. You can enable any site using **`sudo a2ensite site-name`** or disable any site using **`sudo a2dissite site-name`** (ignore the `.conf` file extension). For instance, to disable the default site specified in configuration file, `000-default.conf`, type **`sudo a2dissite 000-default`**.

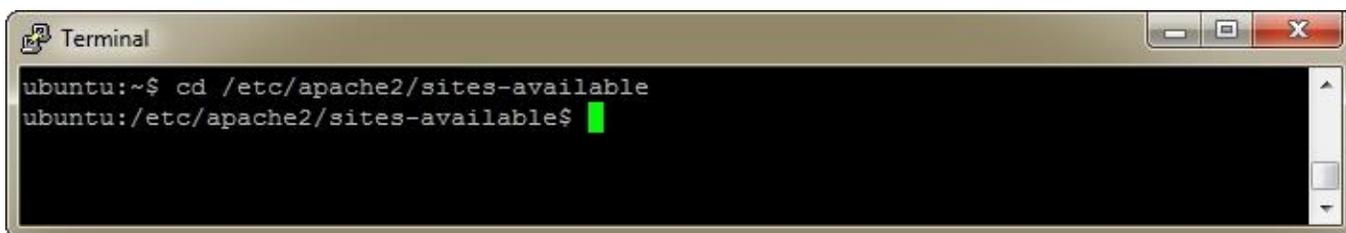
Navigate to the Apache configuration directory:

```
cd /etc/apache2/sites-available
```

[Enter]



```
Terminal
ubuntu:~$ cd /etc/apache2/sites-available
```

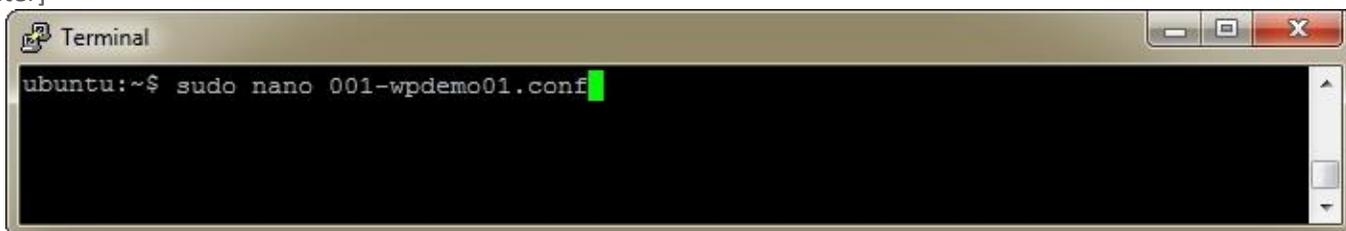


```
Terminal
ubuntu:~$ cd /etc/apache2/sites-available
ubuntu:/etc/apache2/sites-available$
```

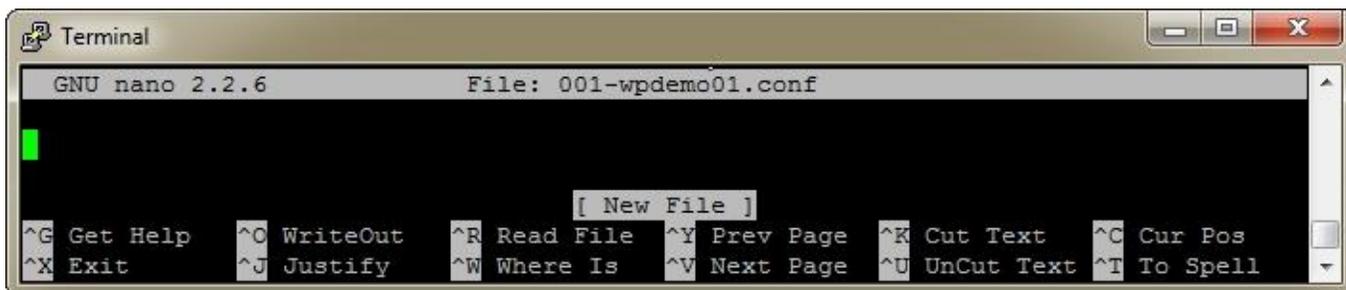
Open the Nano editor to a new configuration file:

```
sudo nano 001-wpdemo01.conf
```

[Enter]



```
Terminal
ubuntu:~$ sudo nano 001-wpdemo01.conf
```



```
Terminal
GNU nano 2.2.6 File: 001-wpdemo01.conf
[ New File ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Below is a secure configuration file that follows many of the best practices when hardening or securing Apache. The only Apache directives you need change when setting up a new site are below:

- ServerName www.example.com
- ServerAlias example.com
- DocumentRoot /var/www/html/wpdemo01
- <Directory "/var/www/html/wpdemo01">

Update the bolded fields so they match the domain name and folder containing the WordPress files. Paste in the following code by simply right clicking in the PuTTY window:

```
### START OF FILE
```

```
#  
# Remove Apache version from response to mitigate  
#  
ServerSignature Off  
ServerTokens Prod
```

```
#  
# Use the ServerName directive to set the name of the server  
#  
UseCanonicalName On
```

```
#  
# Disable the ability for clients to send an HTTP TRACE  
#  
TraceEnable Off
```

```
<VirtualHost *:80>
```

```
    # The ServerName directive sets the request scheme, hostname and port that  
    # the server uses to identify itself. This is used when creating  
    # redirection URLs. In the context of virtual hosts, the ServerName  
    # specifies what hostname must appear in the request's Host: header to  
    # match this virtual host. For the default virtual host (this file) this  
    # value is not decisive as it is used as a last resort host regardless.  
    # However, you must set it for any further virtual host explicitly.
```

```
    ServerName www.example.com  
    ServerAlias example.com
```

```
    ServerAdmin webmaster@localhost  
    DocumentRoot /var/www/html/wpdemo01
```

```
    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,  
    # error, crit, alert, emerg.  
    # It is also possible to configure the loglevel for particular  
    # modules, e.g.  
    #LogLevel info ssl:warn
```

```
    ErrorLog ${APACHE_LOG_DIR}/error.log  
    CustomLog ${APACHE_LOG_DIR}/access.log combined
```

```
    # For most configuration files from conf-available/, which are  
    # enabled or disabled at a global level, it is possible to
```

```

# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
</VirtualHost>

<Directory />
Order Deny,Allow
Deny from all
Options None
AllowOverride None
Require all denied
</Directory>

<Directory "/var/www/html/wpdemo01">
#
# Possible values for the Options directive are "None", "All",
# or any combination of:
# Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
# The Options directive is both complicated and important. Please see
# http://httpd.apache.org/docs/2.2/mod/core.html#options
# for more information.
#
Options None

#
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
# Options FileInfo AuthConfig Limit
#
AllowOverride All

#
# Controls who can get stuff from this server.
#
Order allow,deny
Allow from all

#
# Limit the number of bytes that are allowed in a request body
# 1048576 is 1 MB. 10485760 is 10 MB.
#
LimitRequestBody 10485760

</Directory>

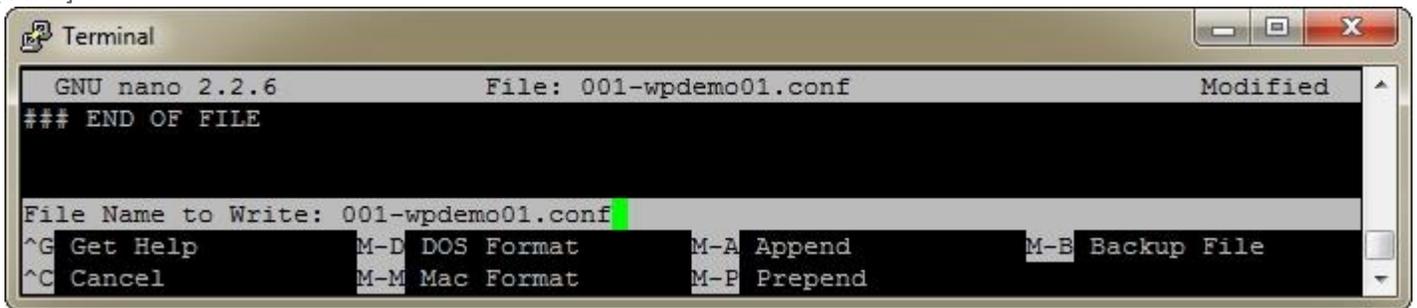
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
### END OF FILE

```

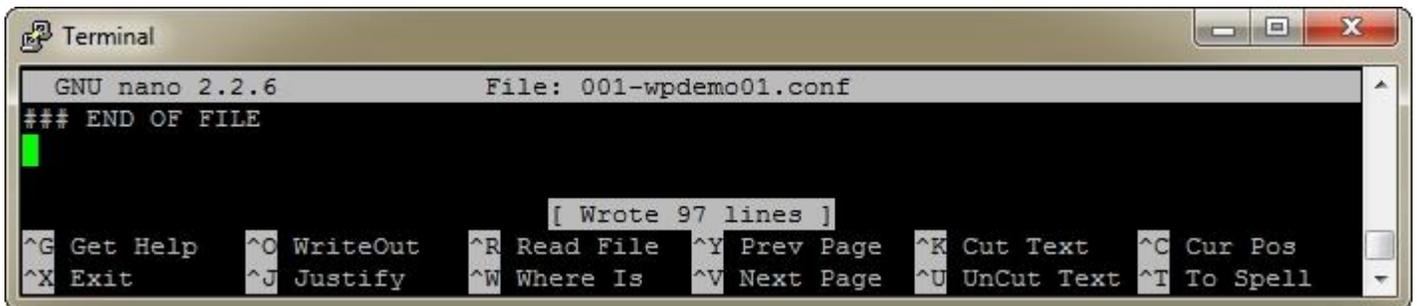
Save the configuration file:

[Ctrl + O]

[Enter]



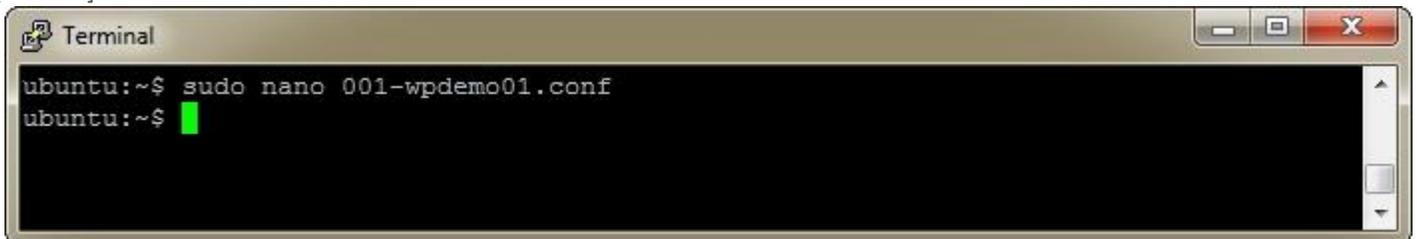
```
Terminal
GNU nano 2.2.6 File: 001-wpdemo01.conf Modified
### END OF FILE
File Name to Write: 001-wpdemo01.conf
^G Get Help M-D DOS Format M-A Append M-B Backup File
^C Cancel M-M Mac Format M-P Prepend
```



```
Terminal
GNU nano 2.2.6 File: 001-wpdemo01.conf
### END OF FILE
[ Wrote 97 lines ]
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Exit Nano:

[Ctrl + X]



```
Terminal
ubuntu:~$ sudo nano 001-wpdemo01.conf
ubuntu:~$
```

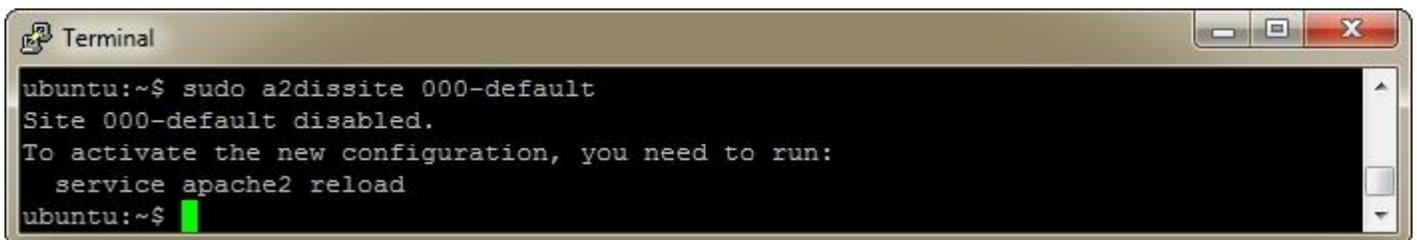
Disable the default Apache website:

`sudo a2dissite 000-default`

[Enter]



```
Terminal
ubuntu:~$ sudo a2dissite 000-default
```



```
Terminal
ubuntu:~$ sudo a2dissite 000-default
Site 000-default disabled.
To activate the new configuration, you need to run:
  service apache2 reload
ubuntu:~$
```

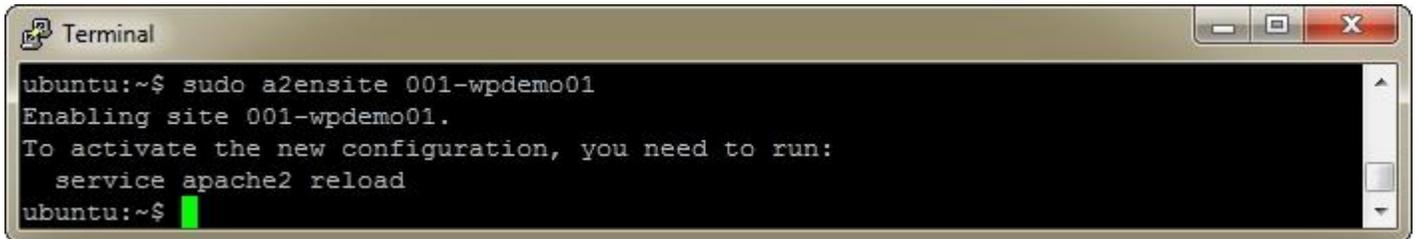
Enable the new Apache website created above:

```
sudo a2ensite 001-wpdemo01
```

[Enter]



```
Terminal
ubuntu:~$ sudo a2ensite 001-wpdemo01
```

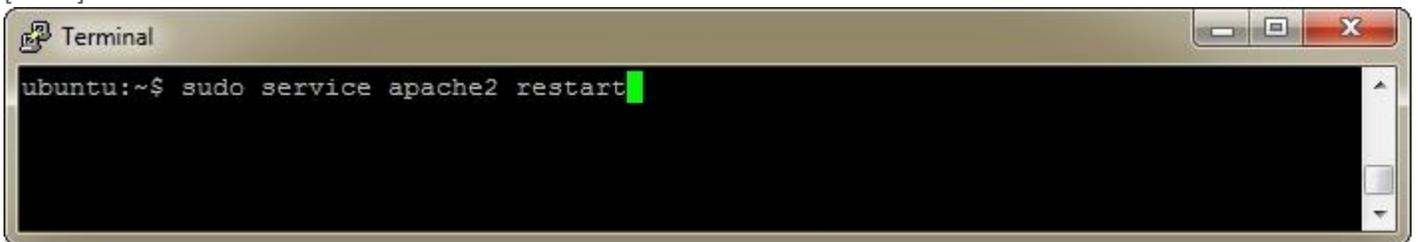


```
Terminal
ubuntu:~$ sudo a2ensite 001-wpdemo01
Enabling site 001-wpdemo01.
To activate the new configuration, you need to run:
  service apache2 reload
ubuntu:~$
```

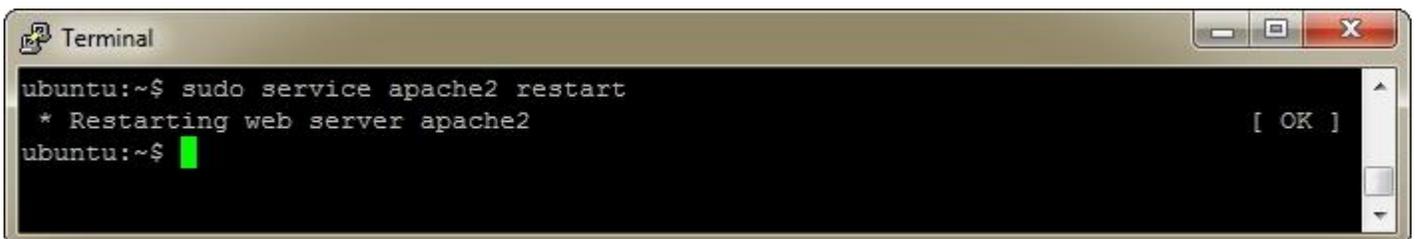
Restart Apache:

```
sudo service apache2 restart
```

[Enter]

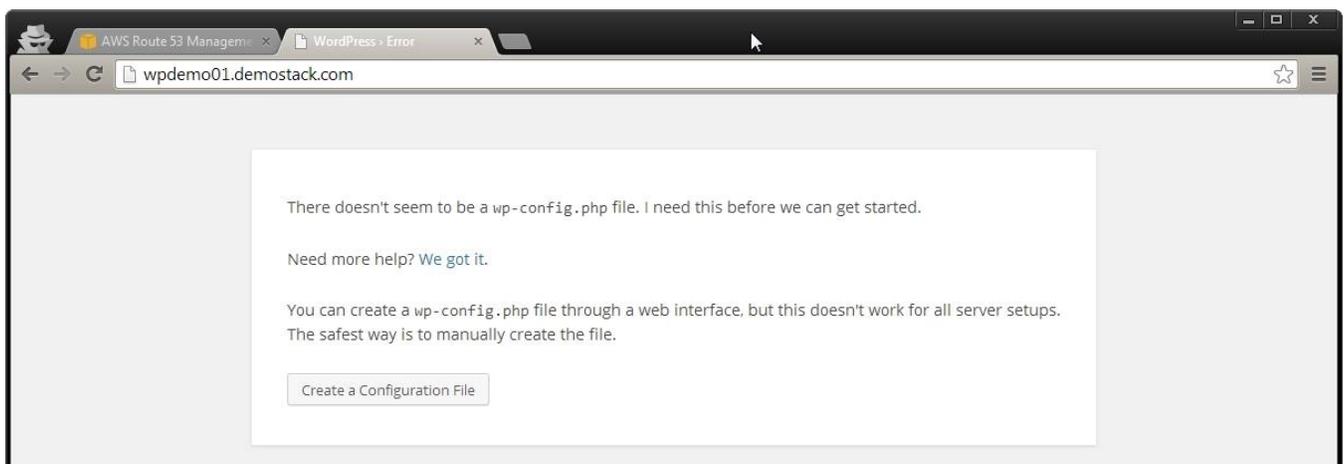


```
Terminal
ubuntu:~$ sudo service apache2 restart
```



```
Terminal
ubuntu:~$ sudo service apache2 restart
* Restarting web server apache2
ubuntu:~$ [ OK ]
```

Apache is now setup. Open your web browser to your domain (ex. <http://www.example.com>) and you should see the **WordPress > Error** page with the **Create a Configuration File** button.



13 Set up MySQL Database using Adminer

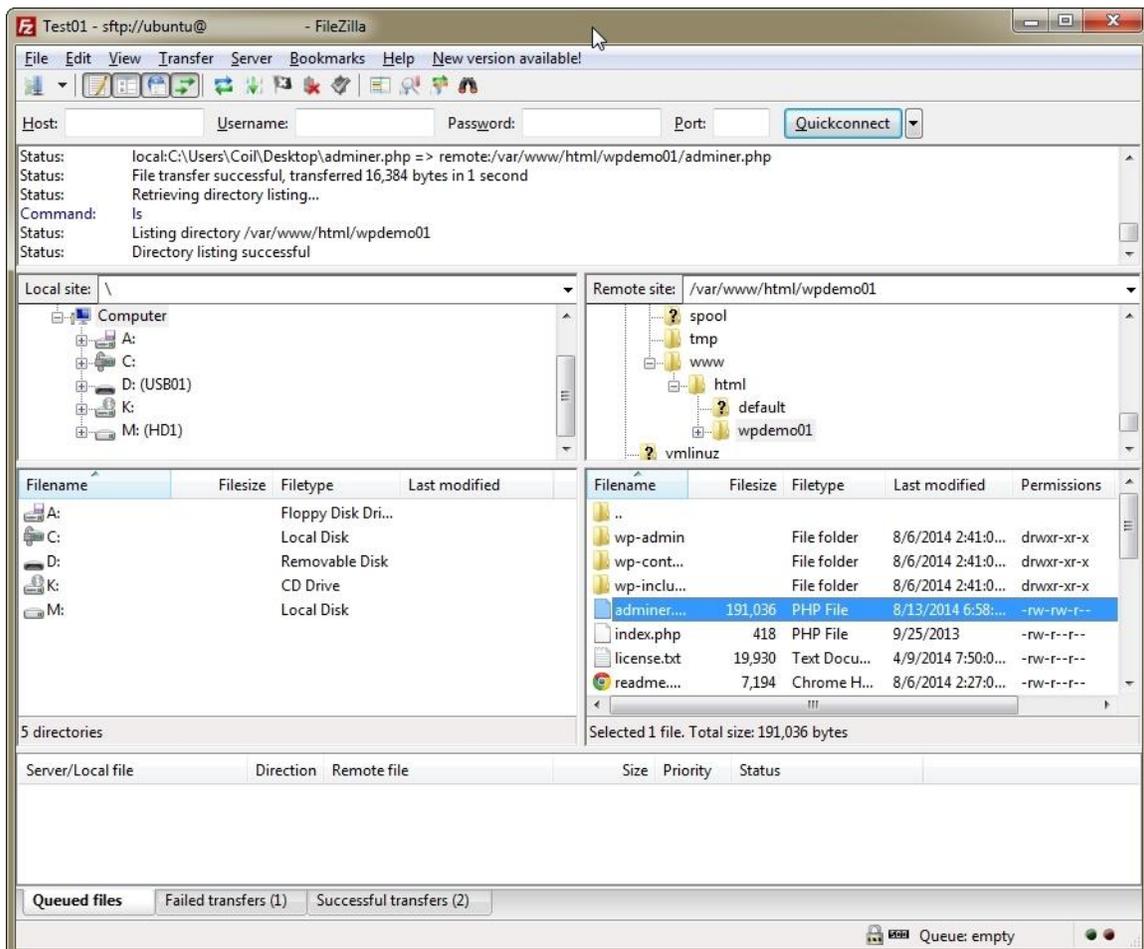
There are multiple ways to set up the MySQL database, but this guide will use Adminer to create a database, create a user, and then assign the user permissions to the database. Delete the **adminer.php** file once the database is setup.

13.1 Download and Set up Adminer

To download Adminer, open your web browser to <http://www.adminer.org/#download>. Click **Adminer 4.1.0 for MySQL** to download the application.

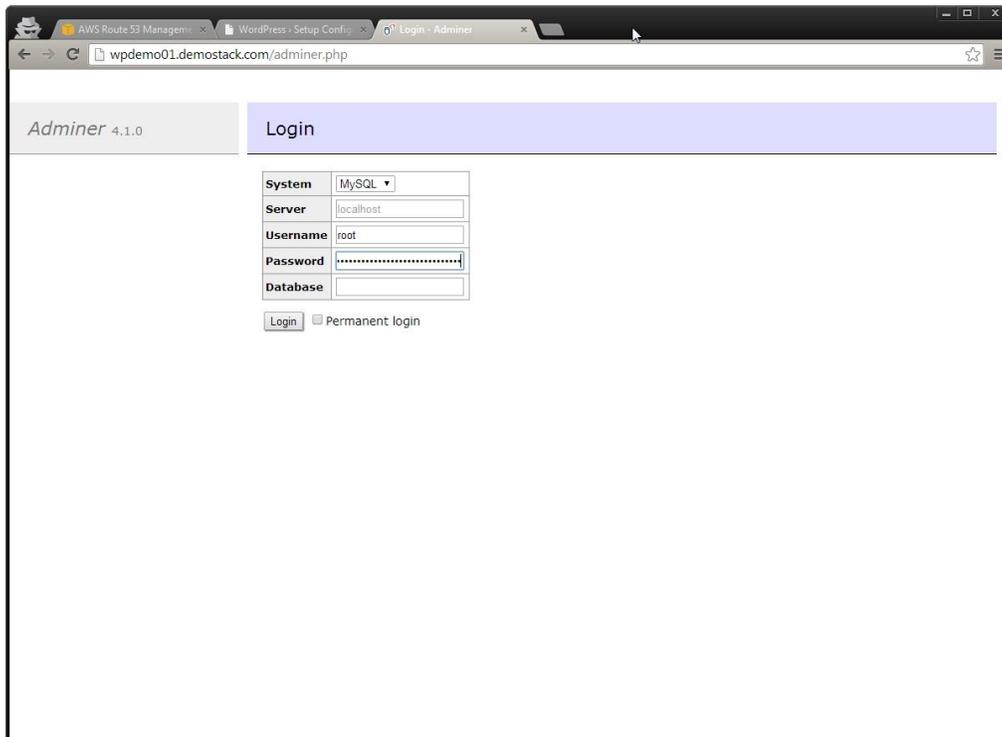


Rename the downloaded file to **adminer.php**. Open up FileZilla and copy adminer.php to `/var/www/html/wpdemo01`.

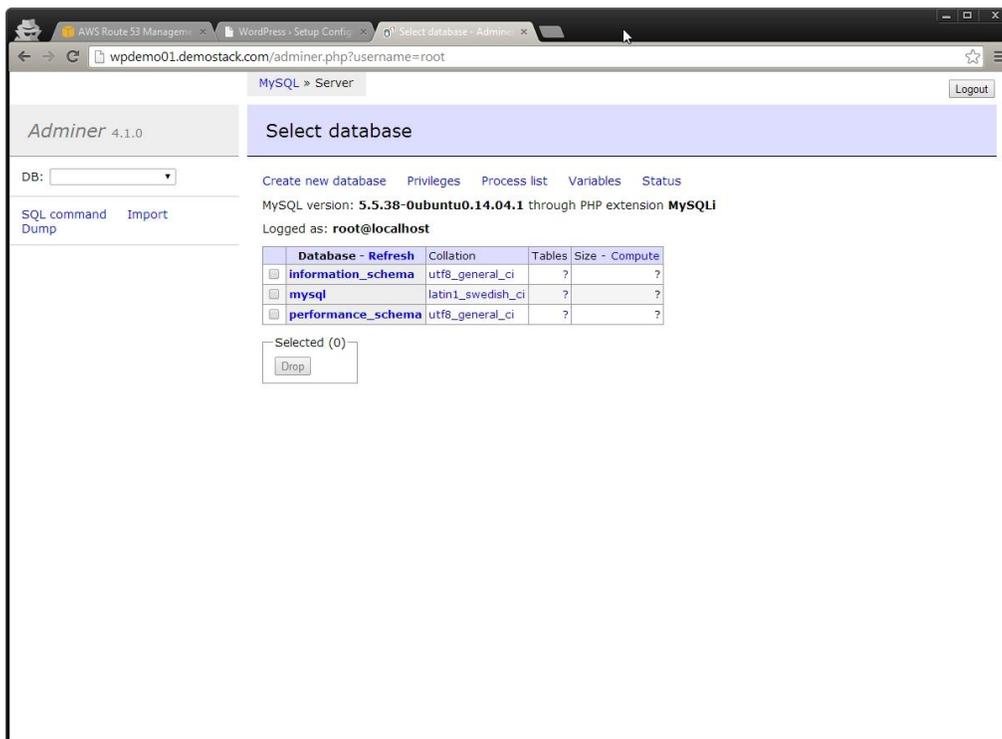


13.2 Create Database and Database User for WordPress

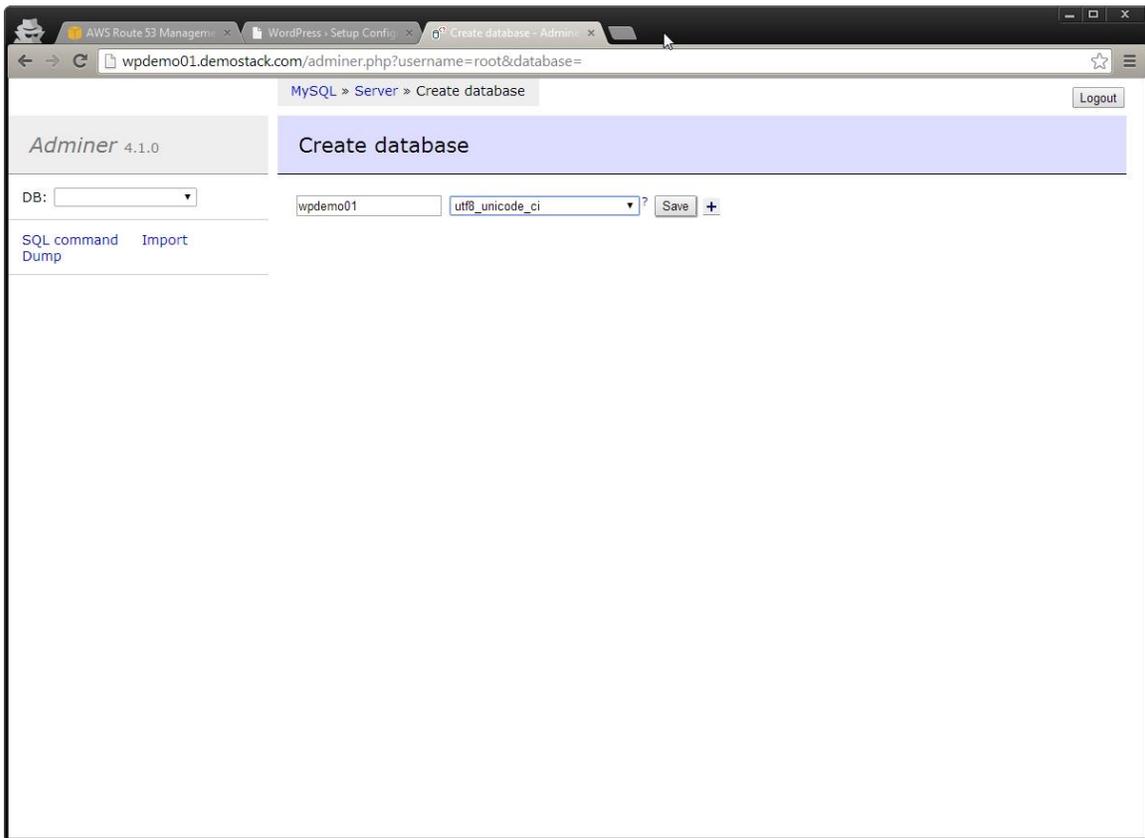
In the web browser, navigate to: <http://example.com/adminer.php>. You should see the Adminer **Login** screen. Type **root** in the **Username** textbox. Type the root password in the **Password** textbox (password set in the **Install MySQL Database** section of this guide). Click **Login**.



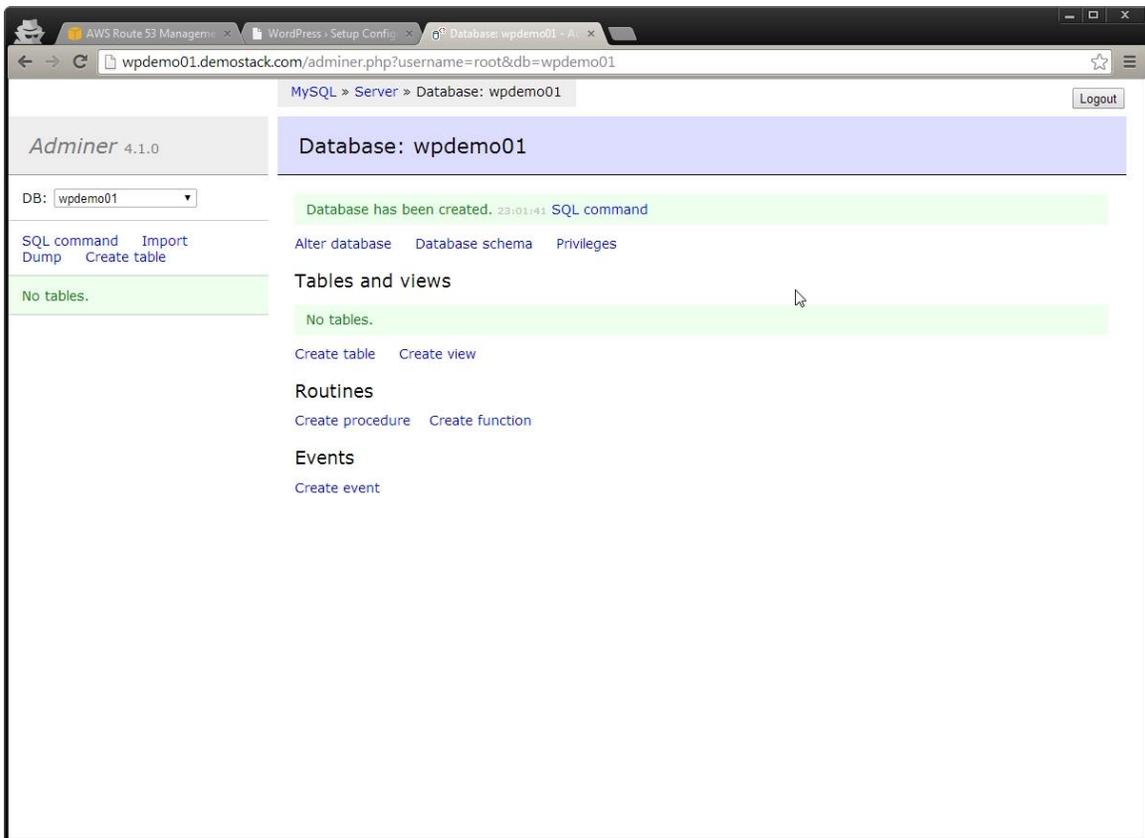
On the **Select database** screen, click **Create new database**.



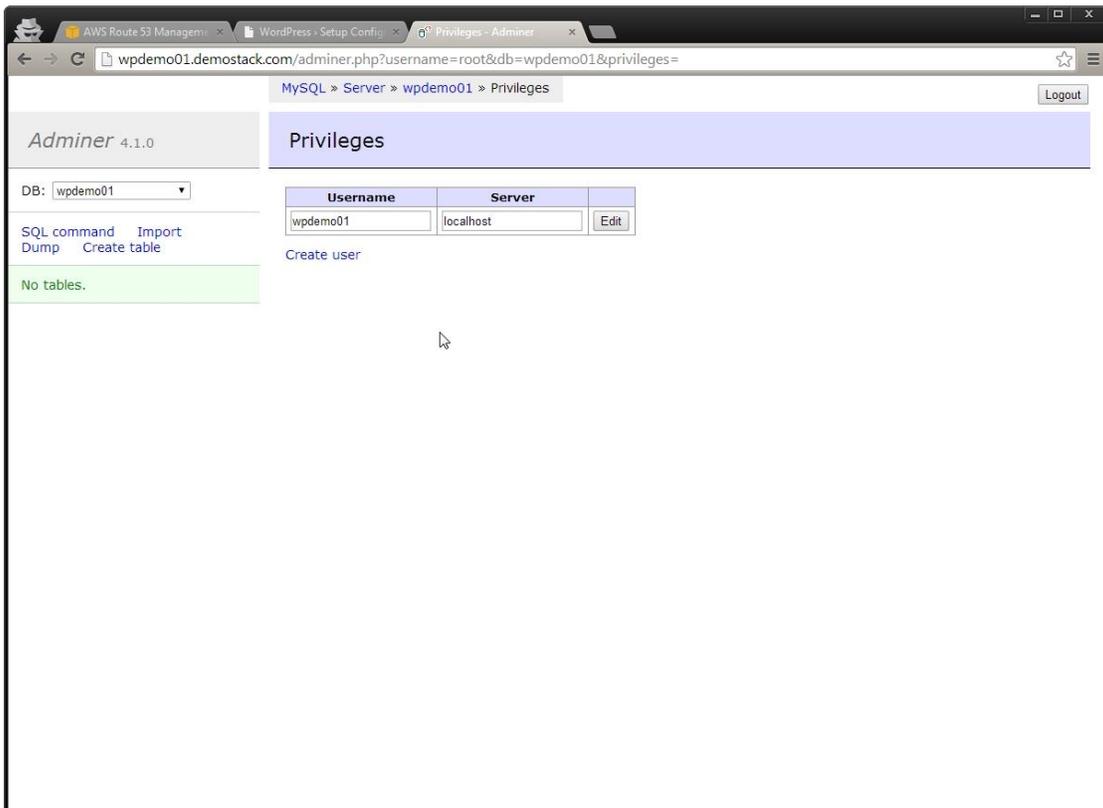
On the **Create database** screen, type in **wpdemo01** or a database name of your choice. Select **utf8_unicode_ci** from the dropdown. Click **Save**.



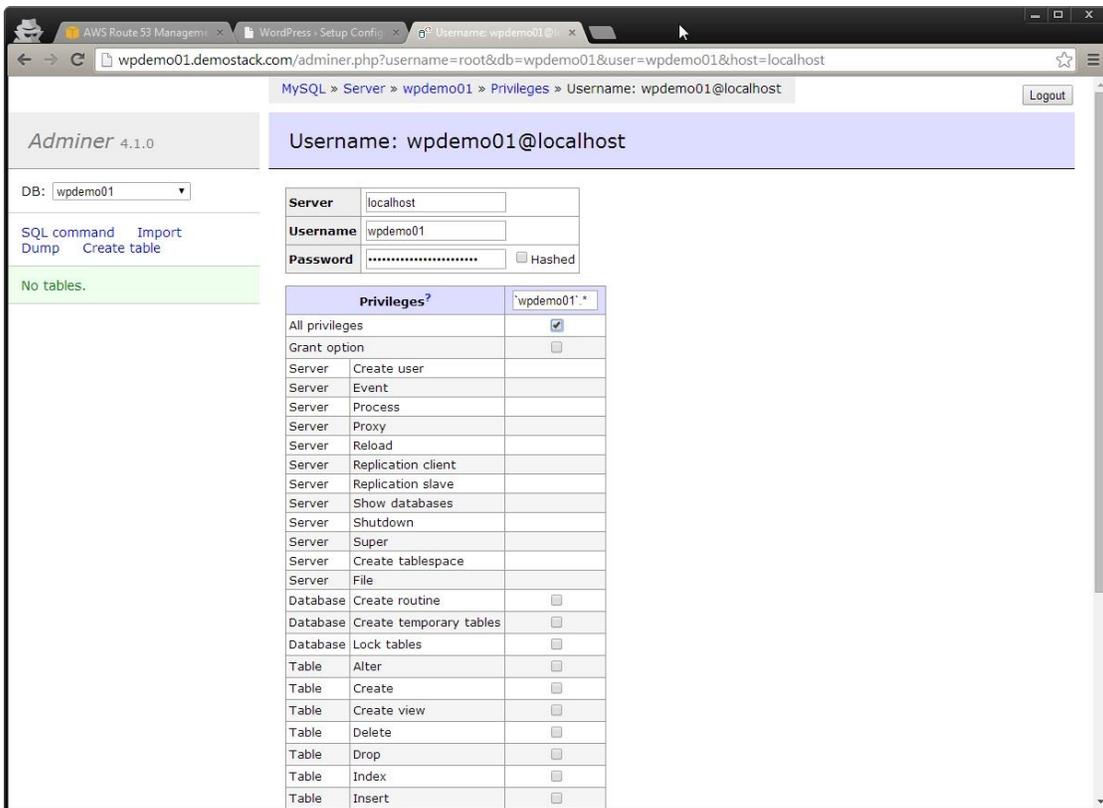
You should see a message at the top of the screen in green that says, **Database has been created.** Click **Privileges**.



On the **Privileges** screen, type **wpdemo01** into the **Username** textbox and **localhost** into the **Server** textbox. Click **Create user**.

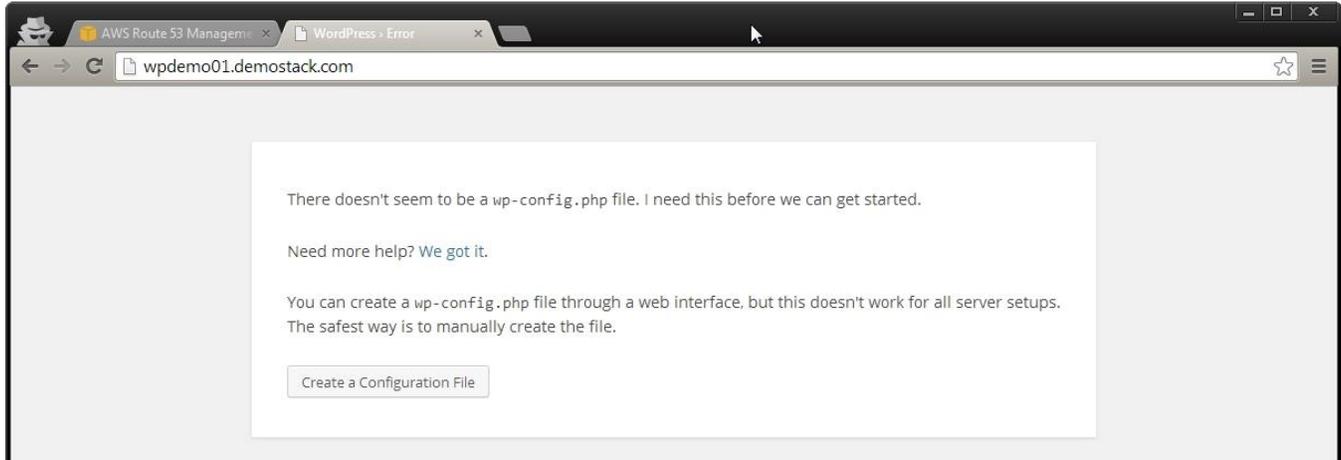


In the **Username** screen, enter in a new password into the **Password** textbox. Make sure the checkbox next to **All privileges** is checked. Click **Save** at the bottom of the page.

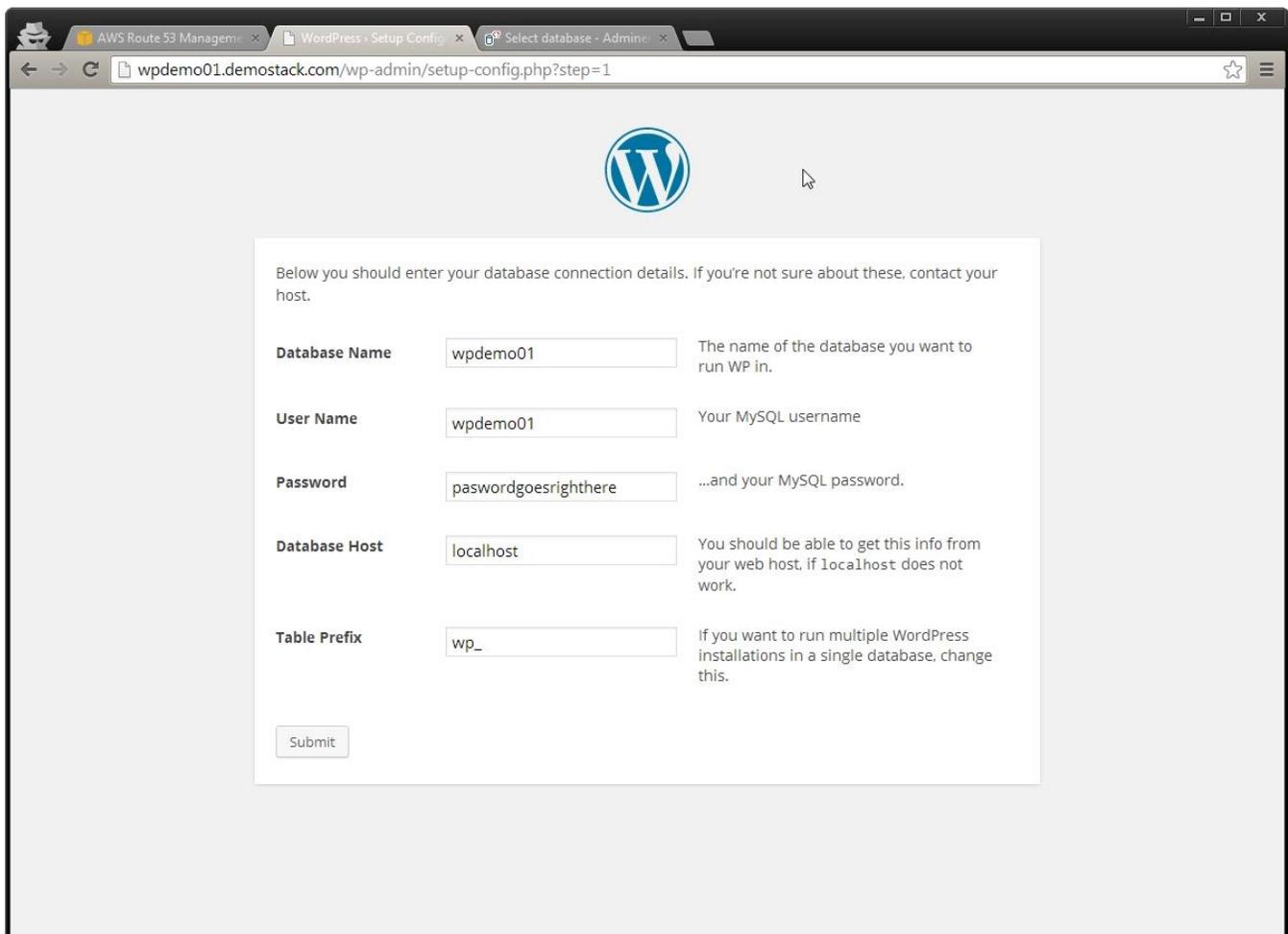


14 Configure WordPress

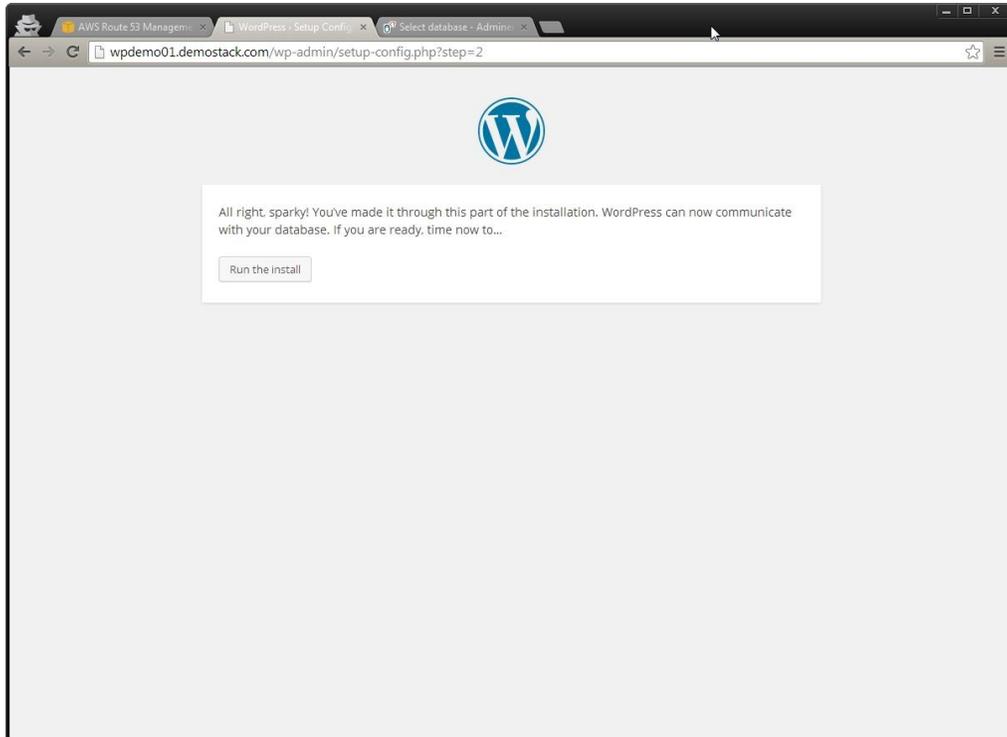
Open your web browser to your domain (ex. <http://www.example.com>) and you should see the **WordPress > Error** page with the **Create a Configuration File** button. Click **Create a Configuration File**.



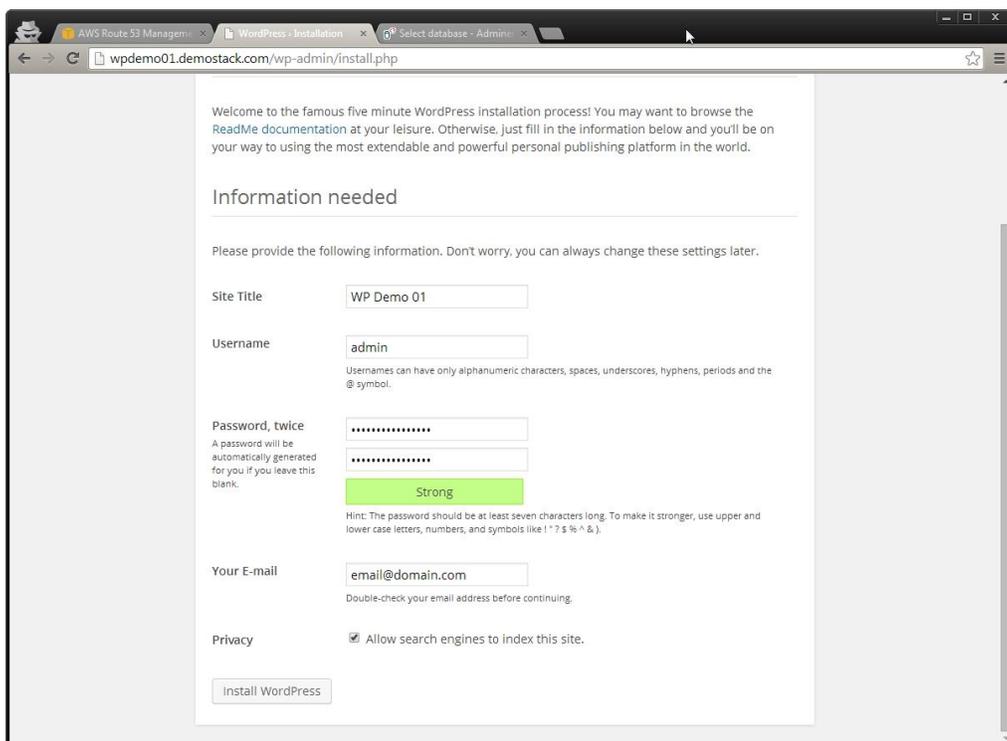
On next screen, you will be prompted for the database information. If you followed the previous section of this guide, enter **wpdemo01** in the **Database Name** textbox, enter **wpdemo01** in the **User Name** textbox, enter in the password you created in the **Password** textbox, enter **localhost** in the **Database Host** textbox, and enter **wp_** in the **Table Prefix** textbox. Click **Submit**.



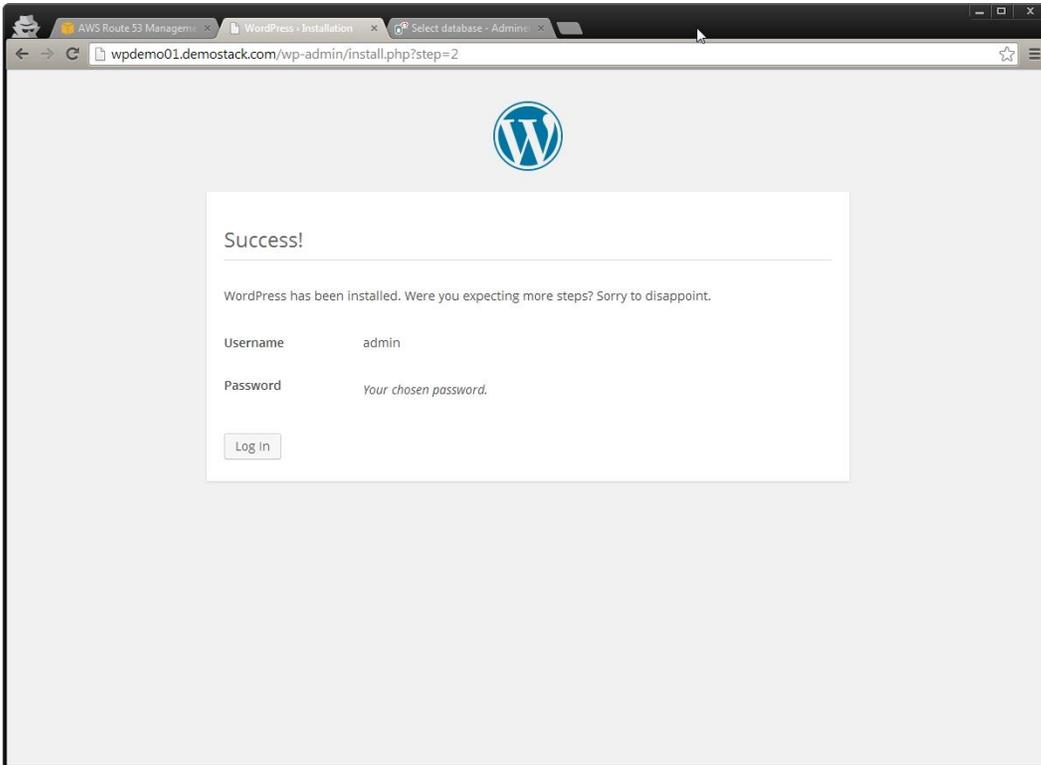
On the last configuration screen, you should see the message, **All right, sparky! You've made it through this part of the installation.** This means WordPress can communicate properly with the database. Click **Run the install.**



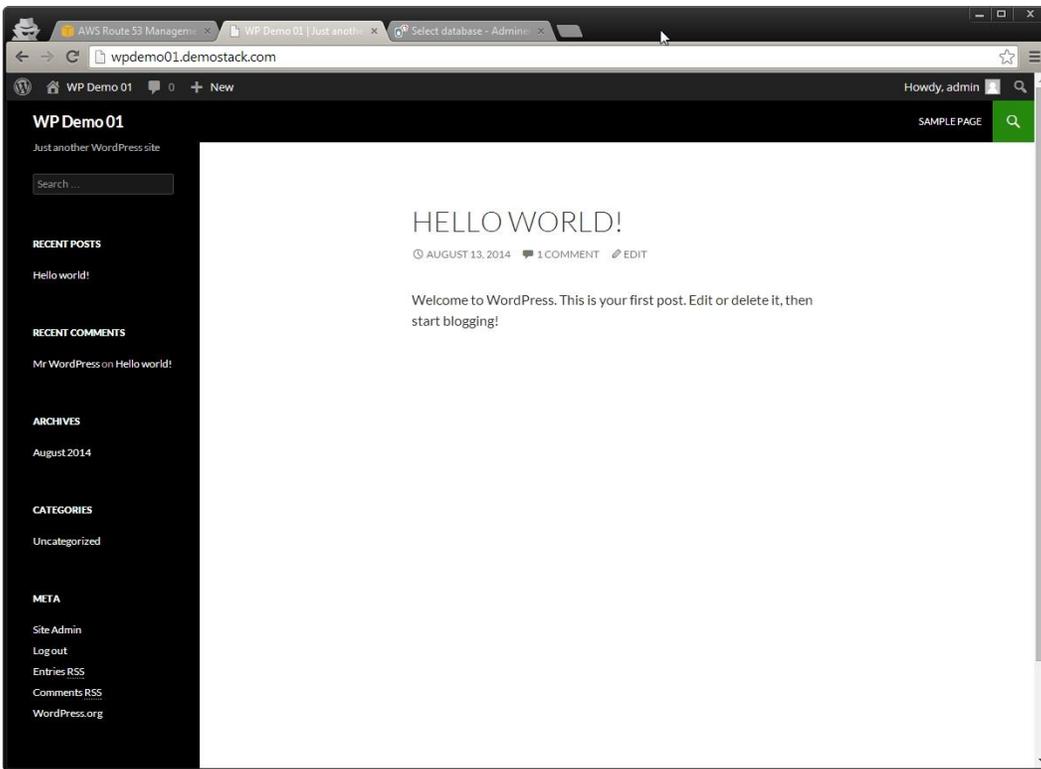
On the **Welcome** screen, enter in the name of your website in the **Site Title** textbox. Enter in a new username into the **Username** textbox. Enter in a new password in both of the **Password** textboxes. This username and password will be used to log in to the WordPress Admin Dashboard. Enter your email in the **Your E-mail** textbox. Check the box next to **Allow search engines to index this site** if you want your website to show up in Google and Bing search results. Click **Install WordPress.**



If you see the **Success** screen, you've successfully configured WordPress.

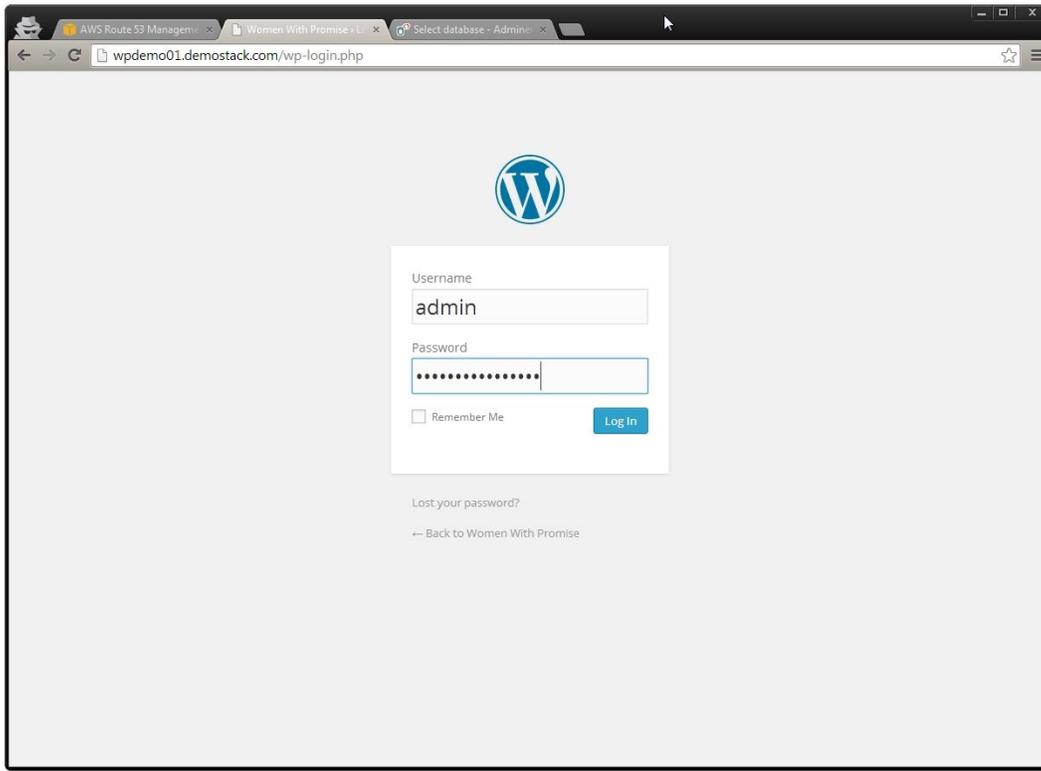


Navigate your web browser to <http://wpdemo01.demostack.com>. You should see the homepage of your new WordPress website! On the page, you'll see the default **Hello World** article.

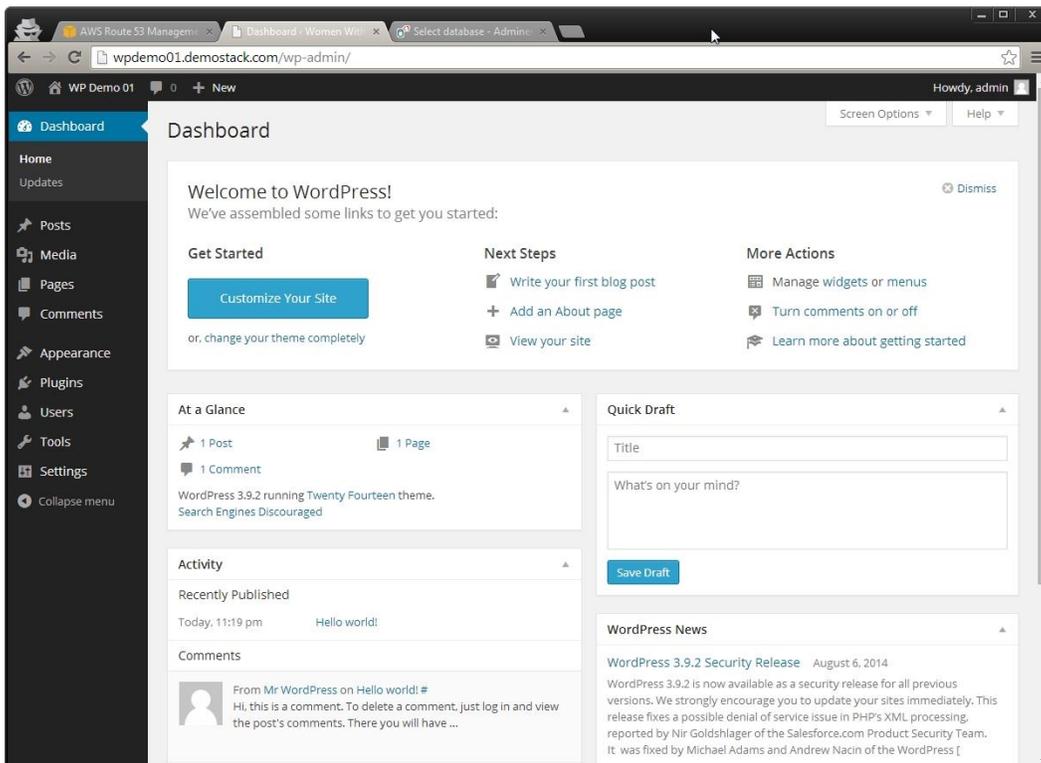


14.1 Log in and Configure WordPress using the Admin Dashboard

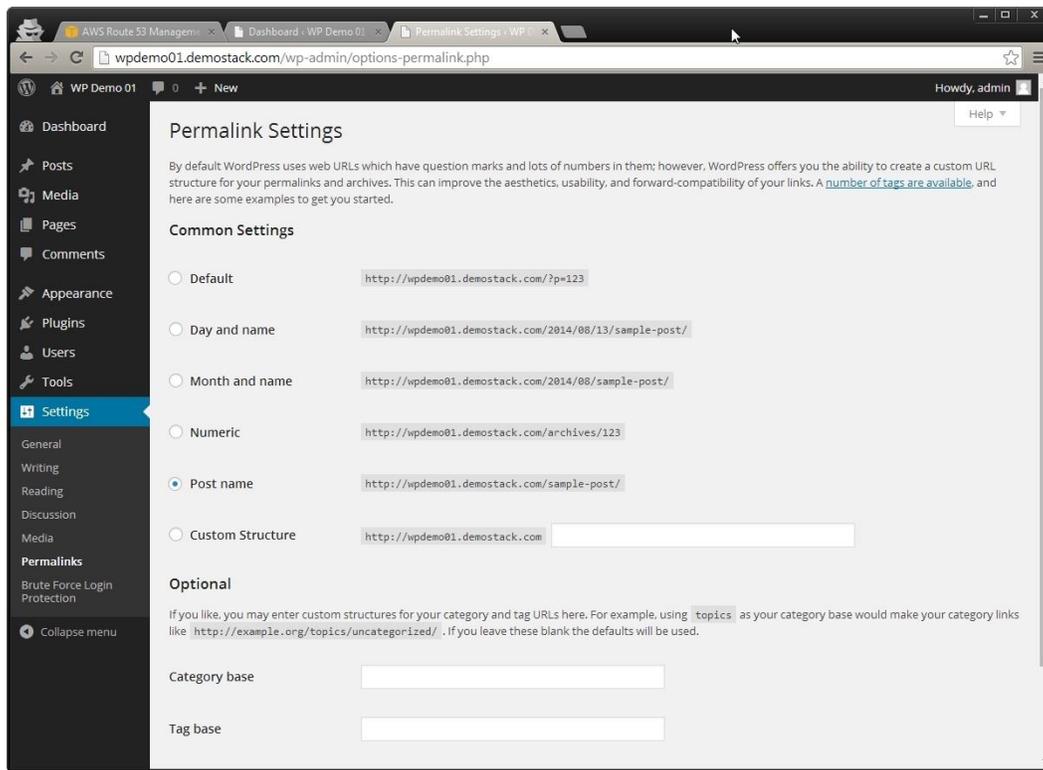
Navigate your web browser to <http://wpdemo01.demostack.com/wp-login.php>. You should see the **WordPress Dashboard Login** screen with a **Username** and **Password** textbox. Enter in the username and password and click **Log In**.



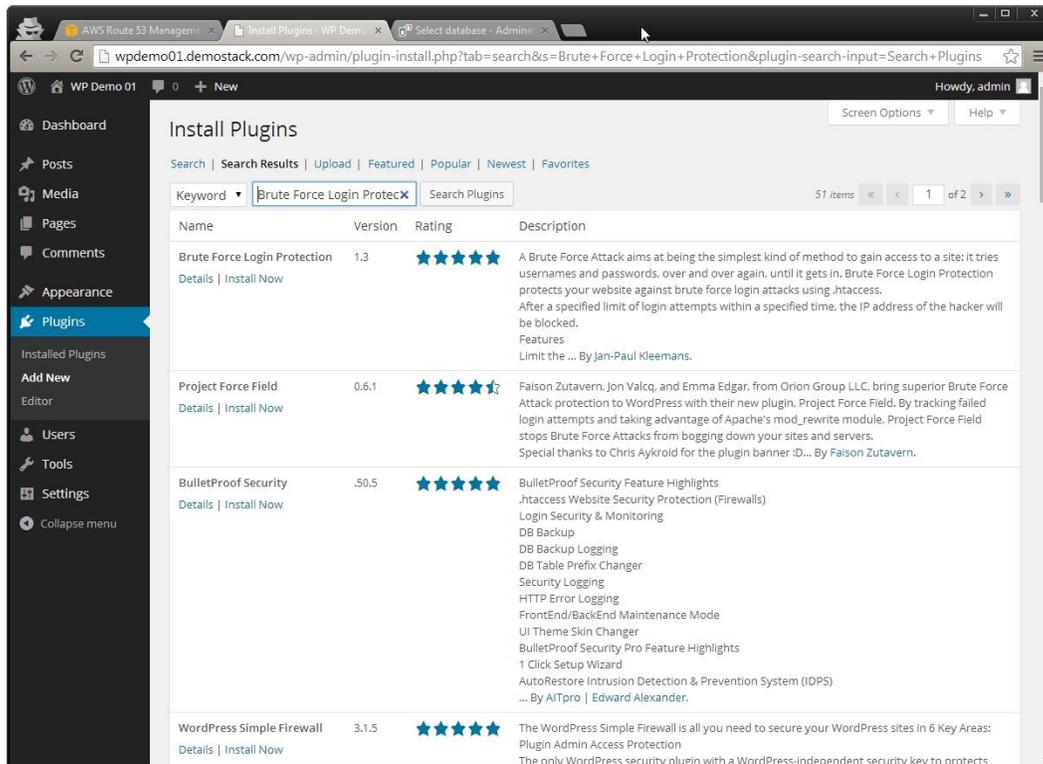
You should see the **Dashboard** screen.



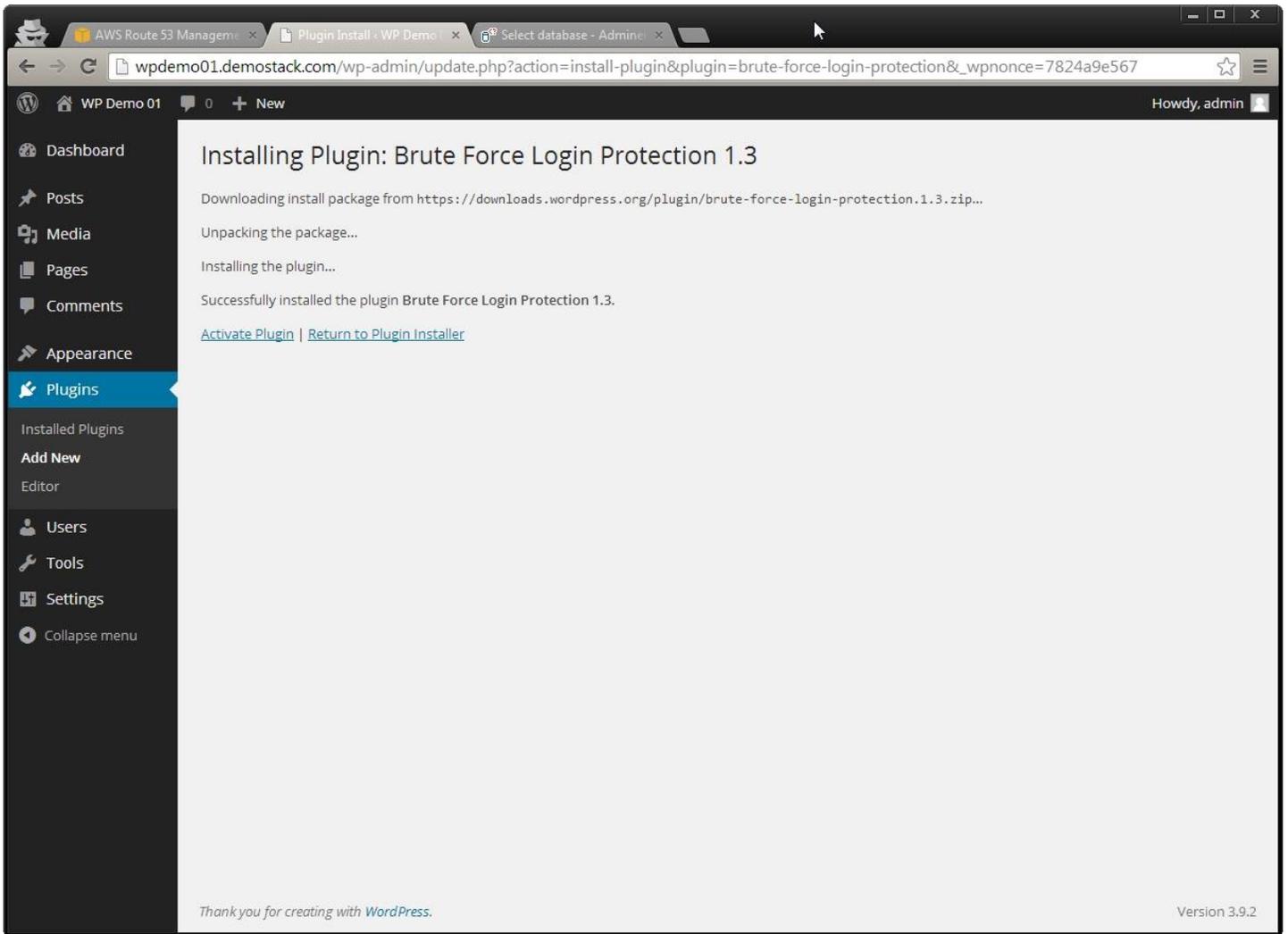
Let's change the way WordPress handles URLs so it is clean and search engine friendly. On the left menu, hover over **Settings** and then click **Permalinks**. Select the radio button next to **Post name**. Click **Save Changes** at the bottom of the page.



Let's also add a plugin that protects the login page from too many invalid attempts. On the left menu, hover over **Plugins** and click **Add New**. Type in **Brute Force Login Protection** and click **Search Plugins**. Under the first result, click **Install Now**.



On the **Installing Plugin** page, click **Activate Plugin**.



Congratulations! You've successfully set up a WordPress website on Amazon Web Services.

15 Final Notes

WordPress is a fantastic tool for maintaining your website. It's easy to use and has a very large library of plugins and themes to aid in customizing your website. To keep the website secure, continue to update WordPress and the plugins to ensure you have the latest versions.

To further increase security on the WordPress website:

- delete the **adminer.php** file so no one can access the database
- copy the values from <https://api.wordpress.org/secret-key/1.1/salt/> to your **wp-config.php** file to make your website harder to hack

15.1 Final Permissions

Once WordPress is setup, run the commands below in PuTTY to apply the proper permissions to the WordPress files:

Set Apache as the owner of the WordPress files:

```
sudo chown www-data:www-data /var/www/html/wpdemo01 -R  
[Enter]
```

Set the recommended permissions for the WordPress folders:

```
sudo find /var/www/html/wpdemo01 -type d -exec chmod 755 {} \;  
[Enter]
```

Set the recommended permissions for the WordPress files:

```
sudo find /var/www/html/wpdemo01 -type f -exec chmod 644 {} \;  
[Enter]
```