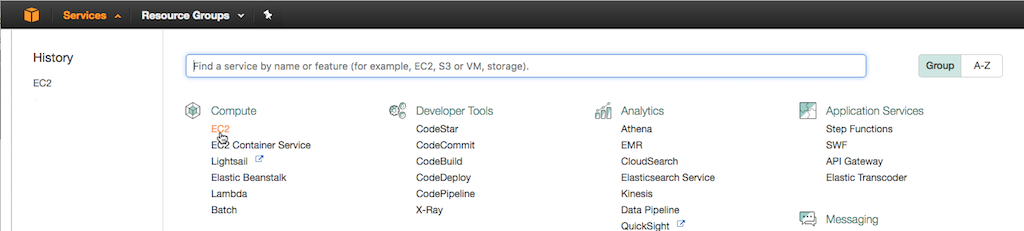
NSetting Up NGINX Web Server

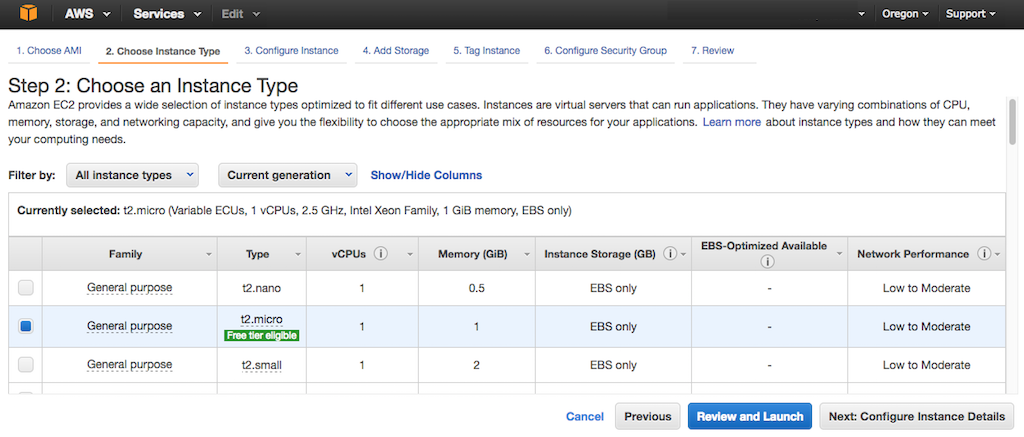
Setting Up AWS

Set up your AWS deployment by creating an EC2 instance and connecting to it.

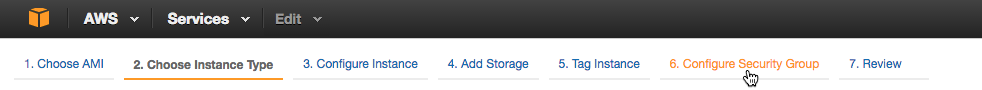
1. On the [AWS home page](https://aws.amazon.com/), sign in to the (AWS Management) Console or create a new account.
2. Click **Services** in the top Console navigation bar, then **EC2** in the **Compute** section.



1. Click the  Launch Instance  button on the page that opens.
2. On the **Step 1** page, click the  Select  button In the **Ubuntu Server 16.04 LTS (HVM), SSD Volume Type** row.
3. On the **Step 2** page, select the **t2.micro** instance type, which as of this writing is marked  Free tier eligible  and selected by default.

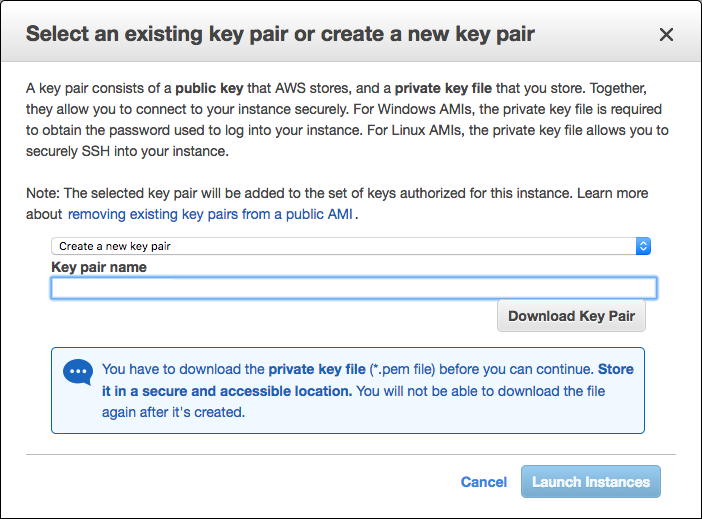


1. By default EC2 instances accept only SSH traffic. To allow incoming HTTP and HTTPS traffic:
   1. At the top of the window, click 6. Configure Security Group.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-configure-security-group.png)

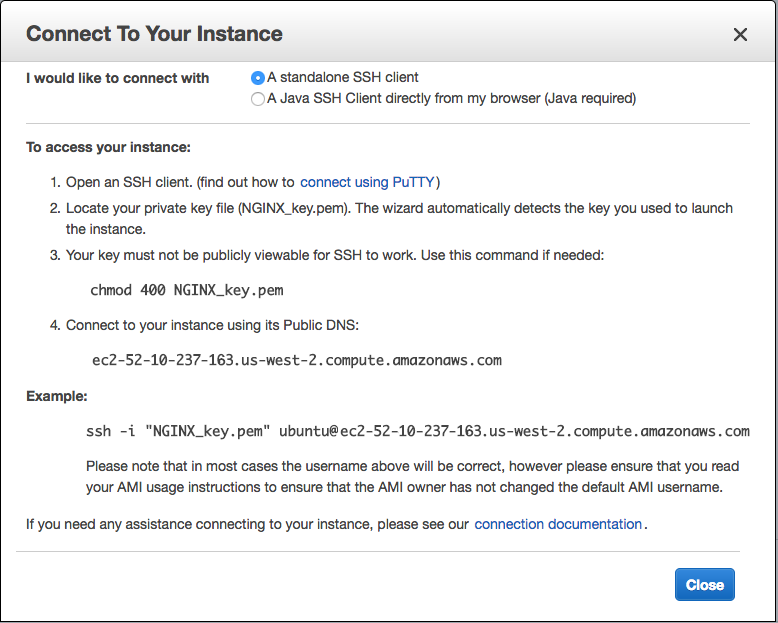
* 1. On the **Step 6** page, click the **Add Rule** button below the table, then select **HTTP** in the **Type** column. We are allowing access to our instances from any IP address (the value in the **Source** column is Anywhere).
  2. Repeat Step b for **HTTPS**.

1. Click the  Review and Launch  button at the bottom of the page.
2. On the **Step 7** page, click the  Launch  button at the bottom.
3. In the box that pops up, create a new key pair:
   1. Select **Create a new key pair** from the upper drop‑down menu.
   2. Type a name in the **Key pair name** field, such as **NGINX\_key**.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-key-pair-name.png)

* 1. Click the **Download Key Pair** button.
  2. In your file manager utility, move the downloaded **.pem** file to a secure location. For the tutorial we’re placing it in the **/Desktop/NGINX** directory.
  3. Click the  Launch Instances  button.

1. On the **Launch Status** page that opens, click the  View Instances  button at the bottom.
2. On the page that opens, the new instance appears in the table. Give it a name:
   1. Click the pencil icon in the **Name** column.
   2. Type the name in the field and click the check‑mark icon.
3. Click the **Connect** button at the top of the screen. A window like the following pops up.

[](https://www.nginx.com/wp-content/uploads/2016/10/aws-ec2-connect-instance-popup.png)

1. Follow the directions on the pop‑up to finish connecting to your instance (including the ones accessible by clicking **connect using PuTTY**, if you’re using Windows). This includes pasting the sample ssh command into your terminal once you’ve navigated to your key.

SSH INTO THE TERMIANL

Installing NGINX Software

1. Update the NGINX software:

**sudo apt-get update -y**

**sudo apt-get upgrade -y**

**sudo apt-get disk-upgrade -y**

**sudo apt-get install awscli**

**sudo apt-get install nginx**

**sudo apt install nodejs npm**

To open the AWS UI, go to Route 53 and add \* for ‘name’ after clicking create record then save.

In the server, go to /etc/nginx/site-available and create yourdomainname.tld add the code below. Change path and subdomain as need.

Setup the link:

LINK TO SITE ENABLED

sudo ln -s /etc/nginx/sites-available/demo.epicmath.com /etc/nginx/sites-enabled/demo.epicmath.com

// server\_name ~^(?<sub>[a-zA-Z0-9-]+)\.domain\.com$; # will cause the sub-domain to

// yourdomainname.tld

server {

listen 80 default\_server;

listen [::]:80 default\_server ipv6only=on;

root /var/www/html;

index index.html index.htm;

server\_name bwatest.com;

location / {

try\_files $uri $uri/ =404;

}

}

server {

listen 80;

listen [::]:80;

root /var/www/html/video;

index index.html index.htm index.php;

server\_name video.bwatest.com;

location / {

try\_files $uri $uri/ /index.html;

}

}

server {

listen 80;

listen [::]:80;

root /var/www/html/display;

index index.html index.htm index.php;

server\_name display.bwatest.com;

location / {

try\_files $uri $uri/ /index.html;

}

============FOR NODE APPS=======================

server {  
listen 80;  
server\_name q-apps.io www.q-apps.io;

location / {  
 proxy\_pass [http://localhost:8080;](http://localhost:8080;/)  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection ‘upgrade’;  
 proxy\_set\_header Host $host;  
 proxy\_cache\_bypass $http\_upgrade;  
 }

}

#for subdomain 1

server {  
listen 80;  
server\_name chat.q-apps.io;

location / {  
 proxy\_pass [http://localhost:9000;](http://localhost:9000;/)  
 proxy\_http\_version 1.1;  
 proxy\_set\_header Upgrade $http\_upgrade;  
 proxy\_set\_header Connection ‘upgrade’;  
 proxy\_set\_header Host $host;  
 proxy\_cache\_bypass $http\_upgrade;  
 }  
}

FOR SSL CERTIFICATE WITH LETSENCRYPT

<https://certbot.eff.org/>

Follow the steps;

Repeat for all domains and subdomains.

FOR PAID SSL CERTIFICATE

server {

listen 80 default\_server;

listen [::]:80 default\_server;

root /var/www/html/assets/cdn/mirror;

index index.html index.htm;

server\_name cdn.pilotx.tv;

location / {

try\_files $uri $uri/ =404;

}

listen [::]:443 ssl;

listen 443 ssl;

ssl\_certificate /etc/ssl/pilotx\_chain\_cert.crt;

ssl\_certificate\_key /etc/ssl/pilotx.key;

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

}

server {

listen 80;

listen [::]:80;

root /var/www/html/assets/cdn/video;

index index.html index.htm index.php;

server\_name video.pilotx.tv;

location / {

try\_files $uri $uri/ /index.html;

}

listen [::]:443 ssl;

listen 443 ssl;

ssl\_certificate /etc/ssl/pilotx\_chain\_cert.crt;

ssl\_certificate\_key /etc/ssl/pilotx.key;

ssl\_protocols TLSv1 TLSv1.1 TLSv1.2;

}

=======================================================================

**OTHERS**

scp -i cdnbox.pem assets2.tar.gz [ubuntu@ec2-54-202-121-12.us-west-2.compute.amazonaws.com:/home/ubuntu/](mailto:ubuntu@ec2-54-202-121-12.us-west-2.compute.amazonaws.com:/home/ubuntu/)

ZIPP a folder

for compressing a directory:

tar -czvf $DESIRED\_ARCHIVE\_NAME $TARGET\_DIRECTORY

Eg. tar -czvf name.tar.gz name

for decompressing:

tar -xzvf $TARBALL\_NAME

Eg. tar -xzvf stuff.tar.gz

CHANGING FOLDER MODE

sudo chown -R $USER ~/.blabla

sudo chown -R ubuntu assets

sudo chgrp -R ubuntu assets

sudo chmod -R 777 $Name

===============Git – How to Change Master Branch=================

1 ) Renamed master branch to oldmaster.

git branch -m master oldmaster

Now there is no master branch on my local machine.

2) Renamed my masterTemp branch to master

git branch -m masterTemp master

// The branch which was named masterTemp on my local machine is now master

3) Delete the branch from remote

git branch -rD master

4) Push the new master branch to remote

git push --force origin master

=========================CHECKOUT TO REMOTE BRANCH==========================

After cloning the repo, run $ cd myproject

$ git checkout -b dev origin/dev

Verify whether you are in the desired branch by the following command;

$ git branch

===================Deployment Key Setup========================================

https://docs.gitlab.com/ee/ssh/

tunnel into an ec2 instance

ssh-keygen -o -t rsa -b 4096 -C "youremail@bluewaterads.com"

tell the prompt you want the files sent to your current working directory, not the ~/.ssh/ directory (example in /home/ubuntu set directory to /home/ubuntu/id\_rsa).

upload id\_rsa.pub as gitlab\_id\_rsa.pub to the S3 bucket bwa-ops-assets/access/ops-automation/$YOUR\_USERNAME/

upload id\_rsa as gitlab\_id\_rsa to the S3 bucket bwa-ops-assets/access/ops-automation/$YOUR\_USERNAME/

copy "config" from the bwa-ops-assets/access/ops-automation/ bucket directory to your username directory.

head into gitlab, and for each project you want ssh key access to, go to Settings->Repository.

a) If you haven't uploaded the key to gitlab yet, copy paste the contents of id\_rsa.pub into the text box under "Deploy Keys" and ensure write access is allowed

b) Otherwise, just select your deploy key from "Privately accessible deploy keys" in the "deploy Keys" secion and activate it.

GIT SSH Deployment Keys

Here are the steps for getting the deployment keys onto a server:

(from home directory on the EC2 box)

DEV

aws s3 cp s3://bwa-ops-assets/access/ops-automation/$YOUR\_USERNAME/gitlab\_id\_rsa ~/.ssh/gitlab\_id\_rsa

aws s3 cp s3://bwa-ops-assets/access/ops-automation/$YOUR\_USERNAME/ssh\_config ~/.ssh/config

sudo chmod 600 ~/.ssh/gitlab\_id\_rsa

sudo chmod 644 ~/.ssh/config

server {

listen 80 default\_server;

listen [::]:80 default\_server ipv6only=on;

root /var/www/html/playapp;

index index.html index.htm;

server\_name playapp.bwaserver.com;

location / {

try\_files $uri $uri/ =404;

}

}