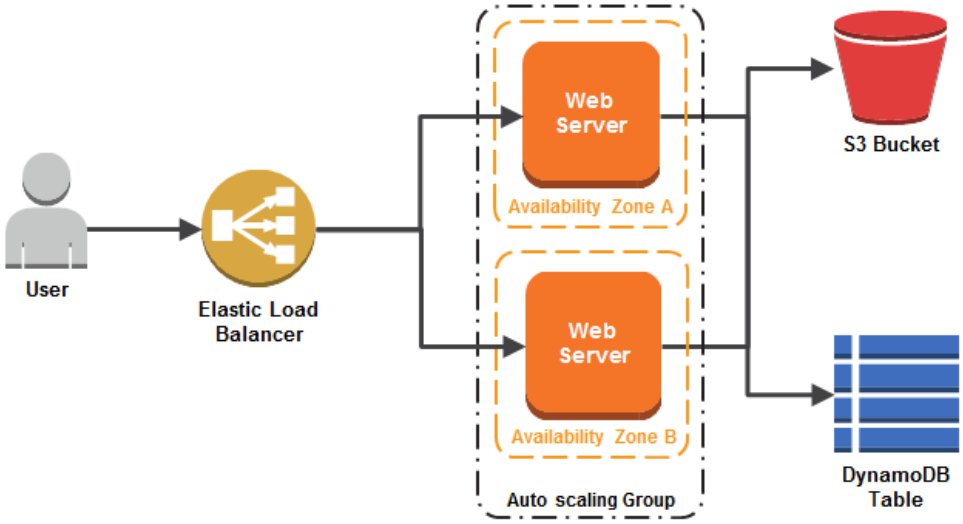
Media Sharing Website
Part 2: Video Transcoding

Table of Contents

Overview

Note: It is not required to complete Part 1 of this lab series in order to take this lab, but the full value of the labs will be realized if done in order.

This diagram represents the architecture of the system built in Lab 1, which resulted in a simple yet fault-tolerant and scalable system to publish images online.



In this second part, you will **extend the system to support videos**. Video files present different challenges compared to images. The architecture you deployed in the first part is able to receive video file uploads but only offers the ability to download them back. Here, we would like to **offer video streaming features** for the users to view the videos online without having to download the complete video file.

However, in order to offer video streaming capability, we must be able to **transform** the various video files we receive from the users into a common “streamable” format. This operation, known as **video transcoding**, is very CPU-intensive and can take a significant amount of time (depending on the video file size) compared to the latency users experience in typical web applications. In order to keep our web application responsive, we are going to implement an **asynchronous video transcoding system**.

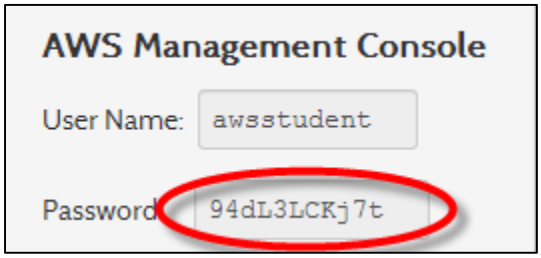
Start your *qwikLAB™*

1. Start your *qwikLAB™*
   Use the 'Start Lab' button to start your lab.
   (Hint: If you are prompted for a token, please use one you've been given or have purchased.)

   You will see the lab creation in progress.

2. Note a few properties of the lab.
   a. **Duration -** The time the lab will run for before shutting itself down.
   b. **Setup Time -** The estimated lab creation time on starting the lab.
   c. **AWS Region** - The AWS Region the lab resources are being created in.

3. Copy the Password provided.
   d. Hint: selecting the value shown and using Ctrl+C works best



4. Click the 'Open Console' button.

5. Make sure that you are not logged into any other instances of the AWS console (in a student account or your own account), as this may cause conflicts when you open the console and log in below for this lab.

6. Login to the AWS Management Console

   Enter the User Name '**awsstudent**' and paste the password you copied from the lab details in *qwikLAB™* into the Password field.

   Click on the 'Sign in using our secure server' button.

In this step you logged into the AWS Management Console using login credentials for a user provisioned via AWS Identity Access Management in an AWS account by *qwikLAB*™.

Media storage

Note: This step will walk you through the creation of an Amazon S3 bucket that was used in Part 1. Detailed explanations have been removed.

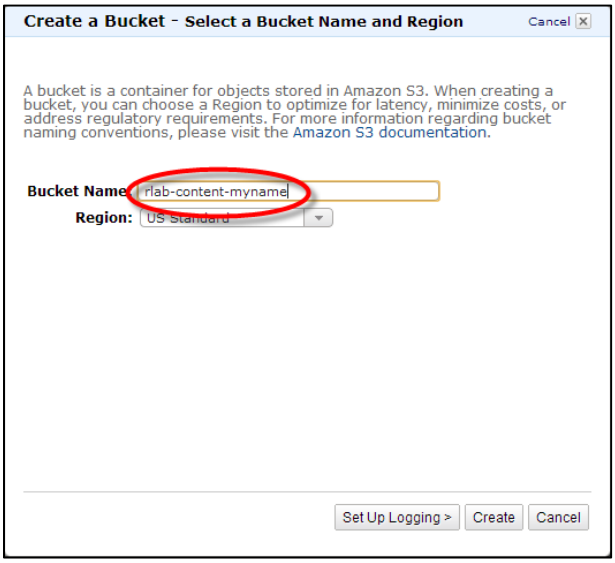
Select the S3 Service

1. Select **S3** from the Console Home:

Creating an Amazon S3 Bucket

2. Click the **Create Bucket** button:

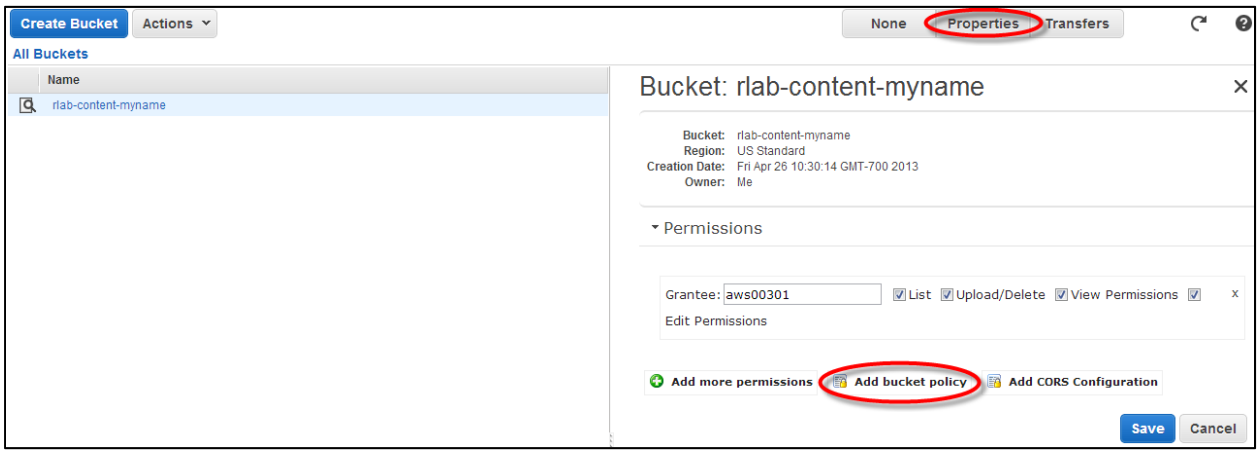
3. Type a name for your bucket. If the bucket name is taken, try another one:



4. **Write down the name of your bucket** as it will be required in future steps.
5. Set the region to **US Standard**.
6. Click the **Create** button to create your bucket.

Assign a Bucket Policy

7. To add a policy to your bucket:
    a. Click on your bucket name
    b. Click **Properties** (in the top-right)
    c. Click on **Permissions** to show more options
    d. Click **Add bucket policy**



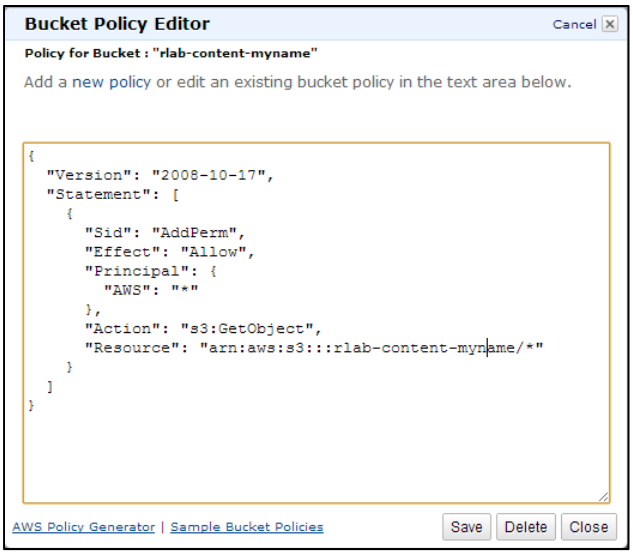
8. In the **Bucket Policy Editor** dialog box, add the following policy to your bucket:

```
{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "AddPerm",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::YOUR_BUCKET_NAME/*"
    }
  ]
}
```

This code is also available from:
https://us-east-1-aws-training.s3.amazonaws.com/self-paced-lab-11/lab11-bucket_policy.json

Replace `YOUR_BUCKET_NAME` with the name of your bucket.

Your policy should look like this:



9. Click **Save** to apply the policy to your bucket.

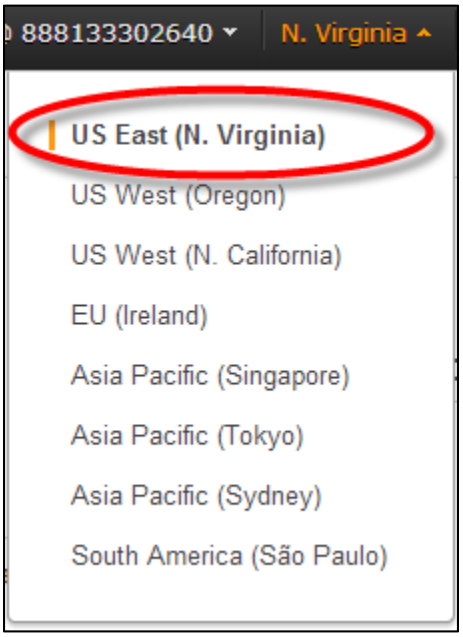
Media Database

Note: This step will walk you through the creation of a DynamoDB table used in Part 1. Details have been removed. Detailed explanations have been removed.

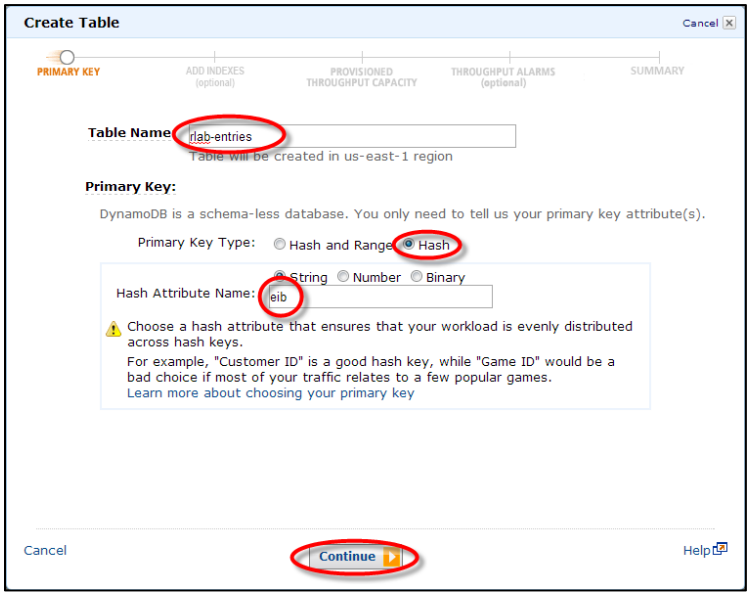
Creating an Amazon DynamoDB table

1. Click **Services** at the top-left of the window and select **DynamoDB**:

2. In the top-right of the screen (next to **Help**), ensure that the AWS Region is set to **US East (N. Virginia)**:

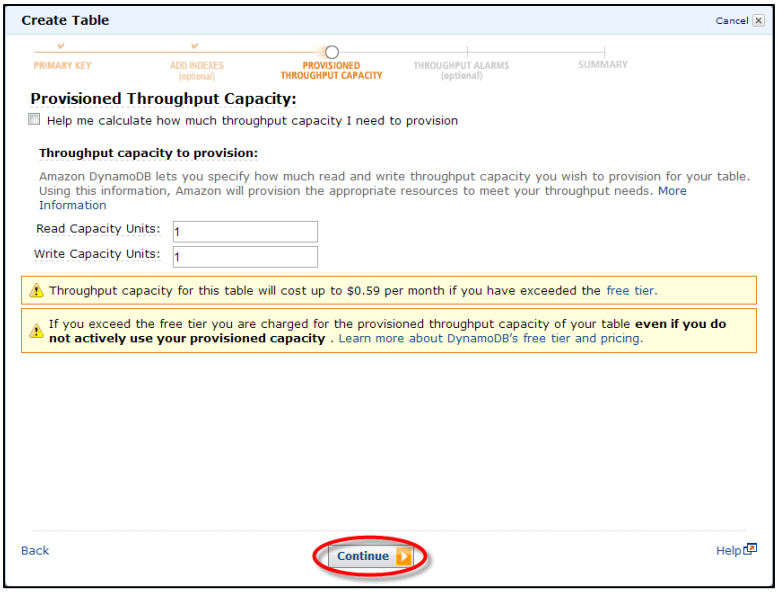


3. Click the **Create Table** button.
4. **Choose your own name** for the table. It must be a unique name, so you might have to try a few times to find a valid name. No spaces are permitted.
5. **Write down the name of your table** as it will be required to complete future steps.
6. Set the **Primary Key Type** to **Hash, String.**
7. Set the **Hash Attribute Name** to: `eib`

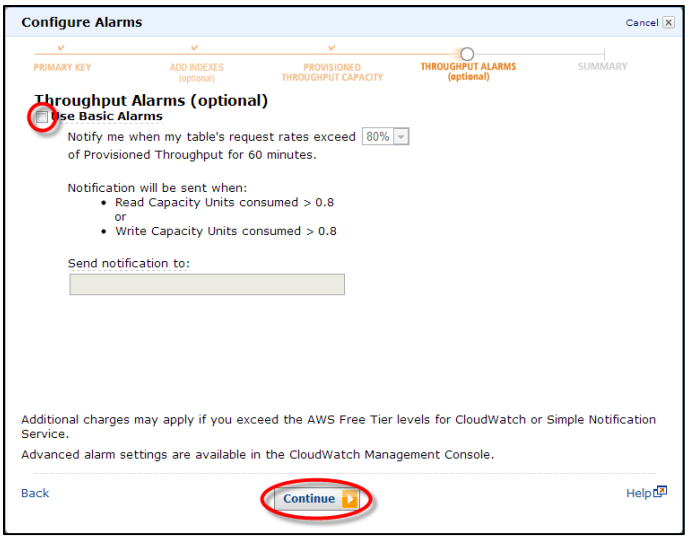


Information about the DynamoDB data model is available at:
http://docs.amazonwebservices.com/amazondynamodb/latest/developerguide/DataModel.html

8. Click **Continue** to create the table.
9. On **Add Indexes (optional)**, click **Continue**.
10. On **Provisioned Throughput Capacity**, click **Continue**.



11. On **Throughput Alarms**, **uncheck "Use Basic Alarms"** and click **Continue**:



12. On the **Review** screen, click **Create**.

    The table status will show as **CREATING**. The table creation process may take a few minutes:

Asynchronous Communication

Asynchronous communication between two subsystems is often implemented using **message queues**. Amazon Simple Queue Service (**Amazon SQS**) offers a reliable, highly scalable, hosted queue for storing messages. Amazon SQS is scalable, it was designed to enable an unlimited number of computers to read and write an unlimited number of messages at any time. To prevent messages from being lost or becoming unavailable, all messages are stored redundantly across multiple servers and availability zones. As such, Amazon SQS is a scalable and fault-tolerant system and doesn't introduce a weakness or a bottleneck in our current architecture.

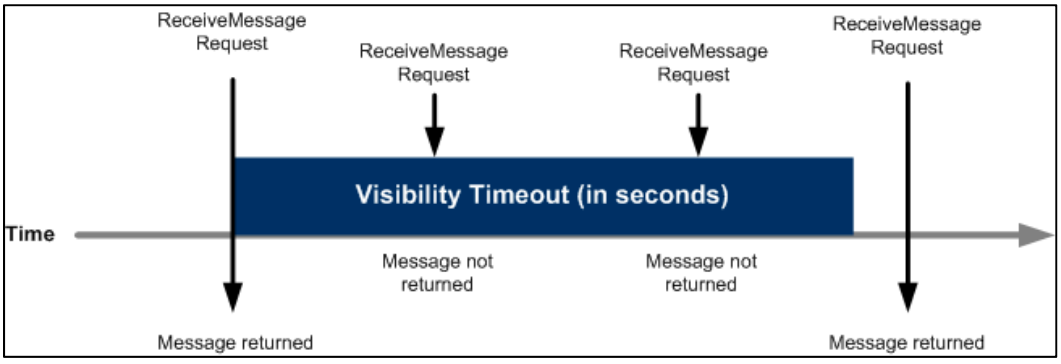
Creating a SQS Queue

1. Select **SQS** from the Console Home:

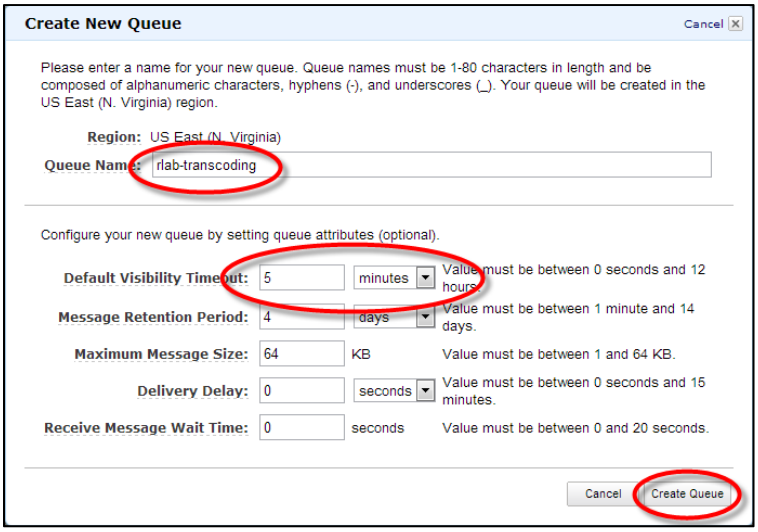
2. Click **Create New Queue**:

When a consuming component in your system receives and processes a message from the queue, the message remains in the queue. Why doesn't Amazon SQS automatically delete it? Because your system is distributed, there is no guarantee that the component will receive and fully process the message (it is possible the connection could break or the component could fail before receiving the message). Therefore, Amazon SQS does not delete the message, so **the transcoder must delete the message** from the queue after receiving and processing it.

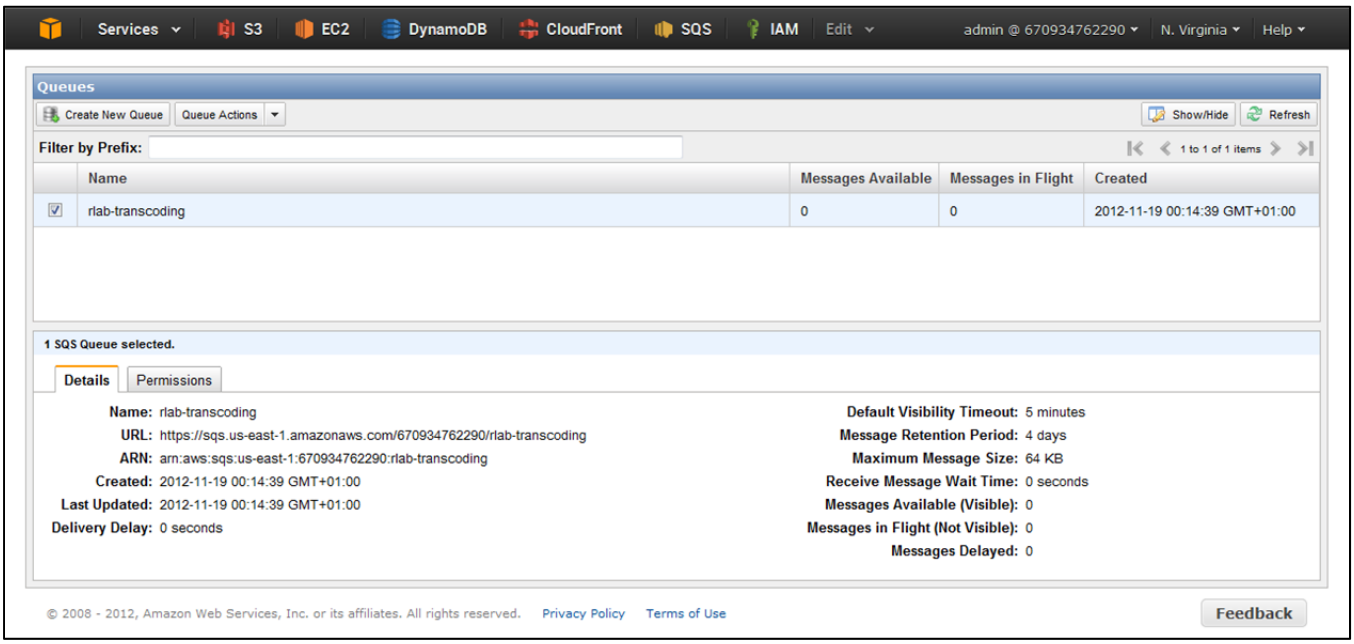
Immediately after the component receives the message, the message is still in the queue. However, you don't want other components in the system receiving and processing the message again. Therefore, Amazon SQS blocks them with a **visibility timeout**, which is a period of time during which Amazon SQS prevents other consuming components from receiving and processing that message. The following figure and discussion illustrates the concept.



3. Provide a **Name** for your queue.
4. **Write down the name of your queue** as it will be required to complete future steps.
5. Set the **Default Visibility Timeout** to **5 minutes**. This means that transcoders will have maximum 5 minutes to complete their task.
6. Click **Create Queue**:



After a few seconds the queue will be ready to receive messages. Note that the AWS Console shows the number of messages in the queue ("Messages Available") and the number of messages currently being processed ("Messages in Flight"):



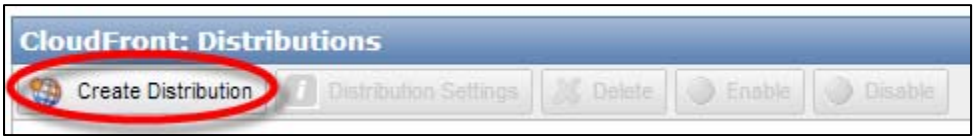
Content Delivery

To deliver video streaming using a Flash player embedded in the web pages of the application, we need a *streaming web server*. Fortunately, this feature is supported by **Amazon CloudFront**, a web service that speeds up distribution of your static and dynamic web content to end users. Amazon CloudFront provides streaming distributions that deliver digital media using Adobe Flash Media Server and the Real-Time Messaging Protocol.

Creating an Amazon CloudFront Streaming Distribution

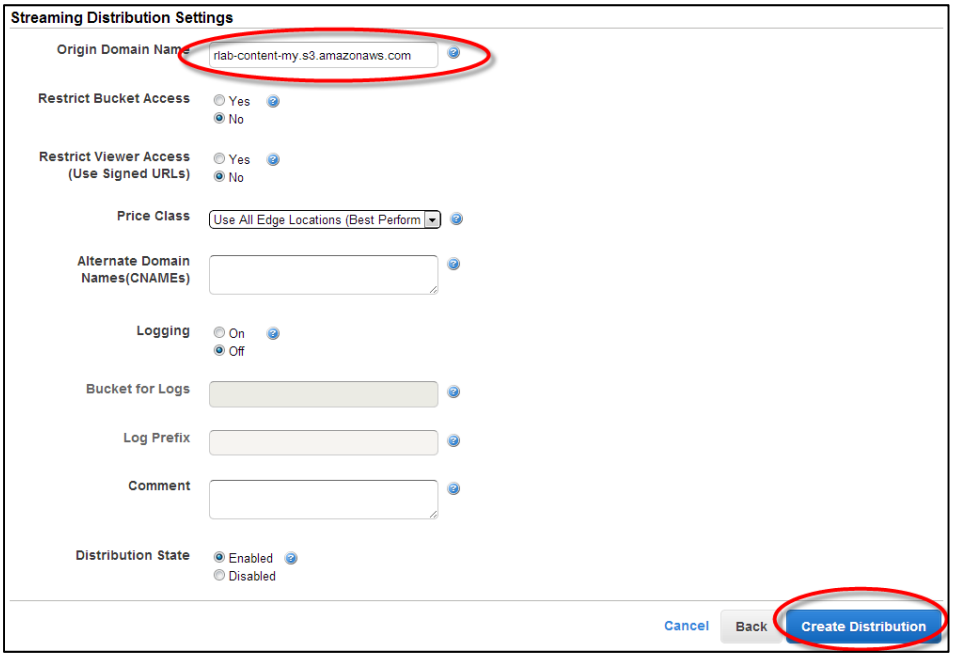
1. Select **CloudFront** from the Console Home:

2. Click the **Create Distribution** button:



3. Choose the **Streaming** delivery method and click **Continue**:

4. In **Origin Domain Name** type the name of your S3 bucket. The full name will appear for auto-completion.
5. Click **Create Distribution**:



Your CloudFront distribution will be created in a couple of minutes.

6. **Note the Domain Name of your distribution** as it will be needed in the next steps. (Click the **Distribution Settings** to see the details in a bigger window):

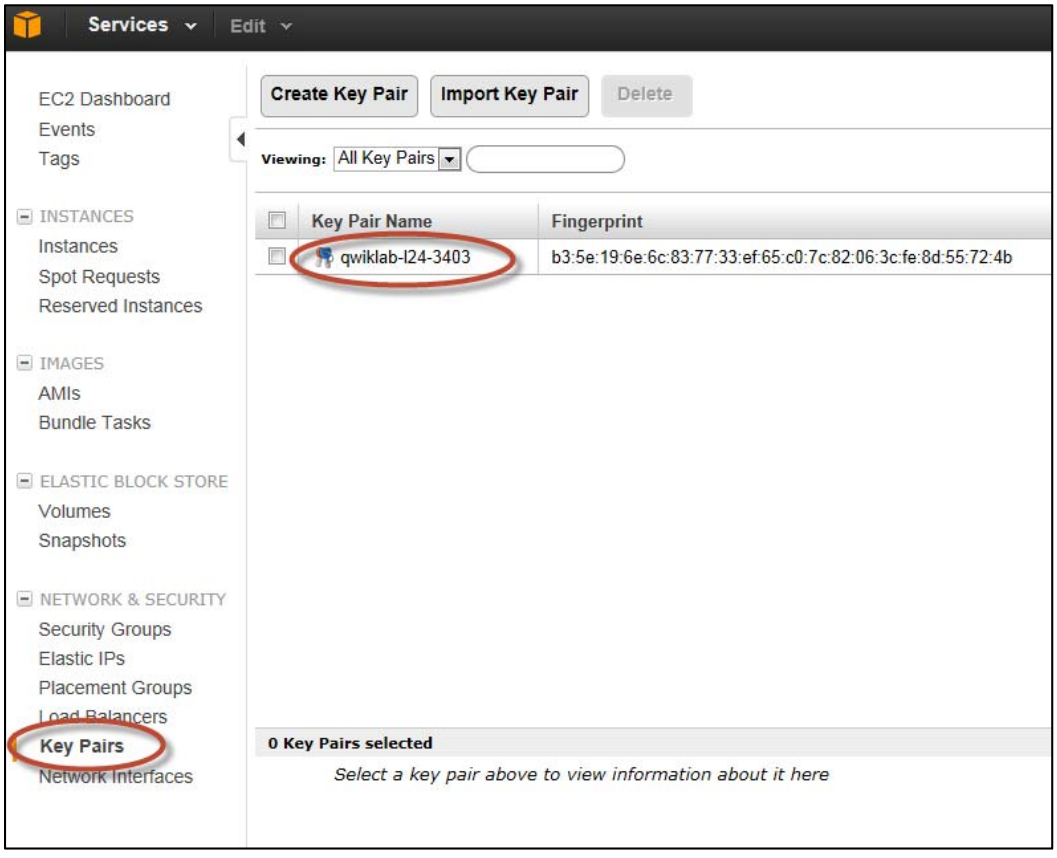
Web Servers and Transcoders

As you did in Part 1, you will use a **CloudFormation script** to build the front end, including an Elastic Load Balancer, a Security Group, an Auto Scaling Group and EC2 instances for web servers. This script will also build an Auto Scaling group and transcoder instances polling the SQS queue.

Get Your KeyPair Name

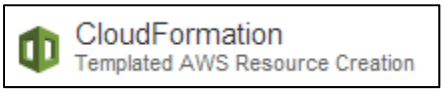
1. Select **EC2** from the Console Home:

2. Select “Key Pairs” from the left column and find your Key Pair Name.
   Make a note of this name. You will need it later.



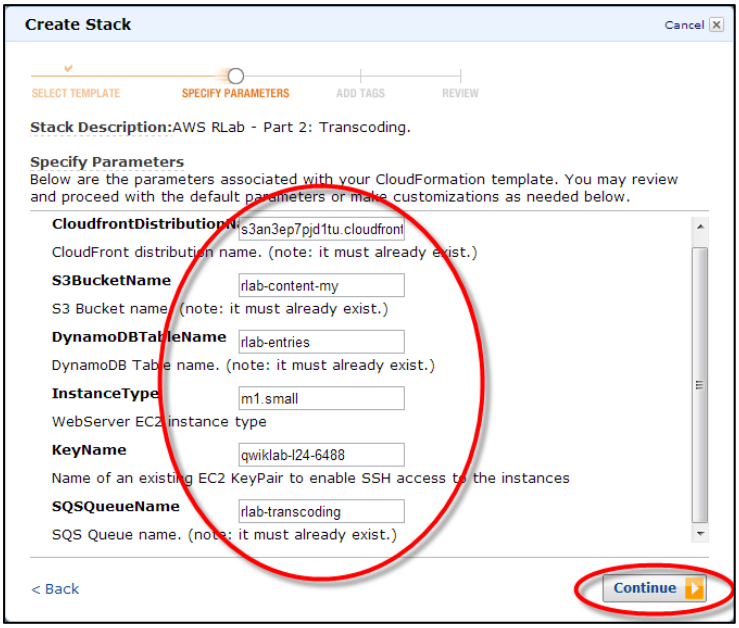
Deployment using CloudFormation

3. Select **CloudFormation** from the Console Home:

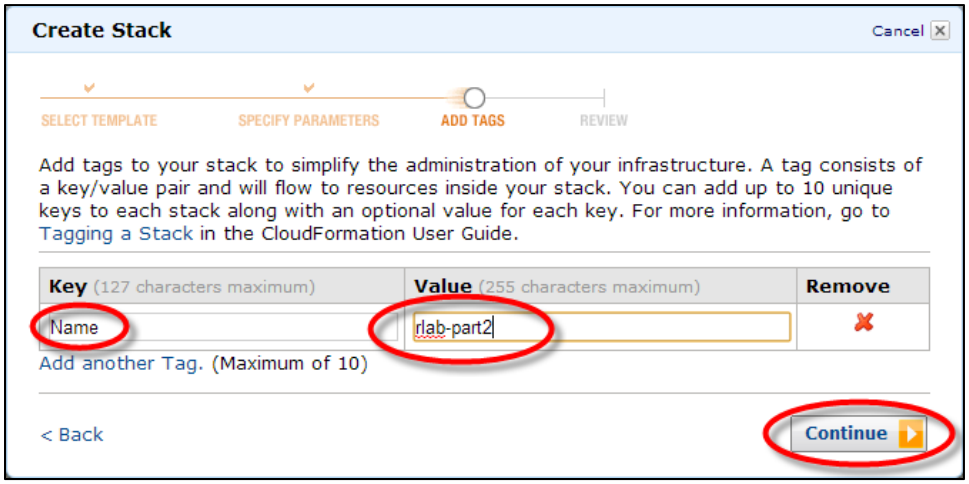


4. Ensure that the Region (top-right of window) is still set to **US East**.
5. Click **Create Stack**.
6. Provide a **Stack Name** of your own choosing.
7. Click **Provide a Template URL** and enter this as the URL for the template:

https://us-east-1-aws-training.s3.amazonaws.com/self-paced-lab-12/rlab-part2-cfn.template

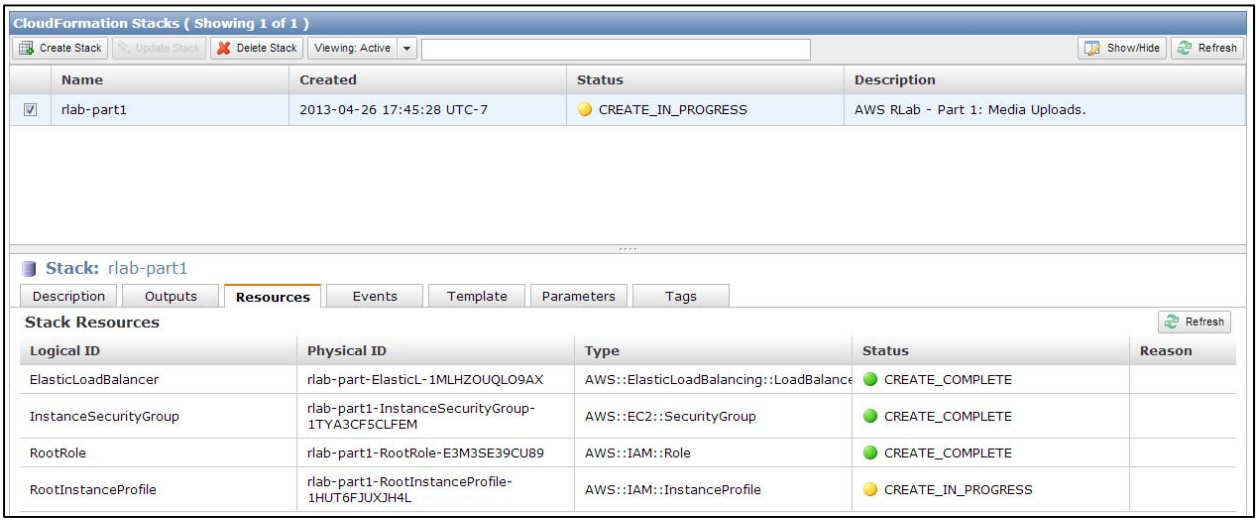


8. Click **Continue**.
9. The stack template contains *parameter placeholders*. Please enter:
    a. Your **CloudFront Distribution Name** (from Step 4)
    b. Your **S3 Bucket Name** (from Step 1)
    c. Your **DynamoDB Table Name** (from Step 2)
    d. The **Key Pair name** for your QuickLabs session that you wrote down from the previous section
    e. The **SQS Queue Name** (from Step 3)
10. Click **Continue**.
11. You can provide some tags which will be assigned to the resources created by this stack:

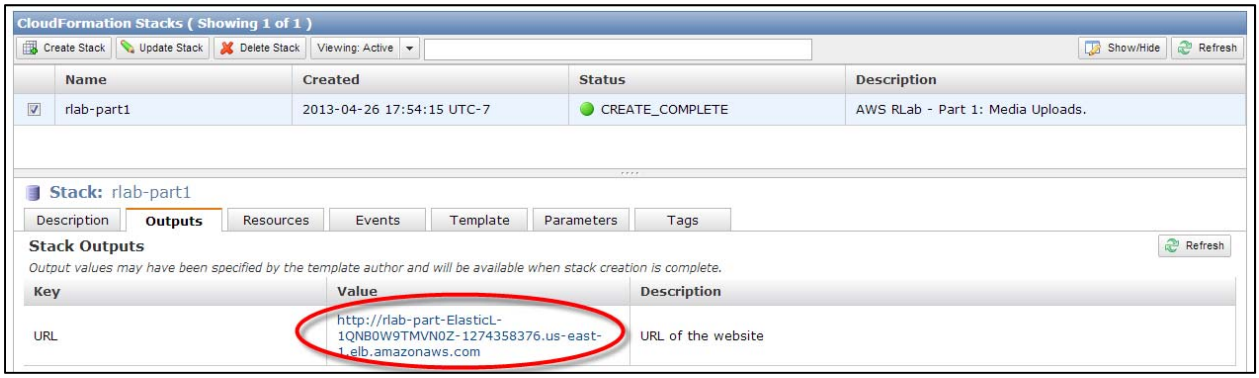


12. Finally, review your stack parameters, then click **Continue** and **Close**.

The stack creation process will take a couple of minutes. You can follow the stack creation process by checking the **Resources tab**:

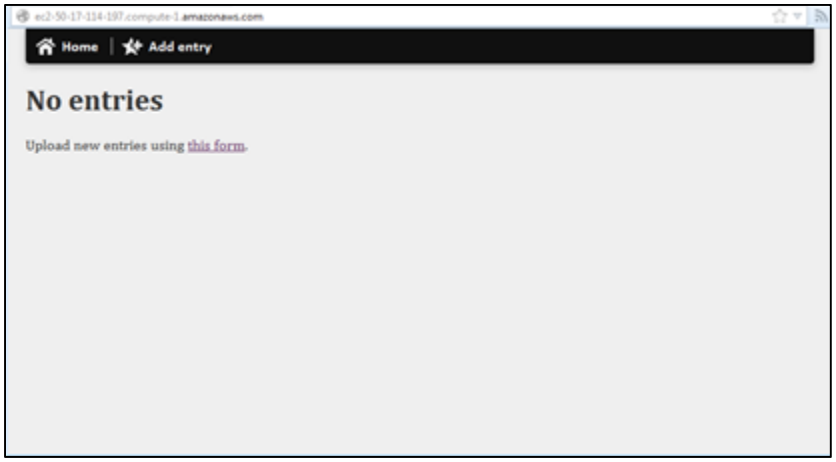


13. Once the stack creation is completed, the **Outputs tab** will show the URL endpoint of the Elastic Load Balancer:



Testing the deployment

1. **Copy and paste this URL into a new tab** of your web browser. You should see the web application home page, which is now being access via the Elastic Load Balancer:



Add a video. The link below will open up a webpage with three sample videos that you can download to your local machine and then upload into the application.

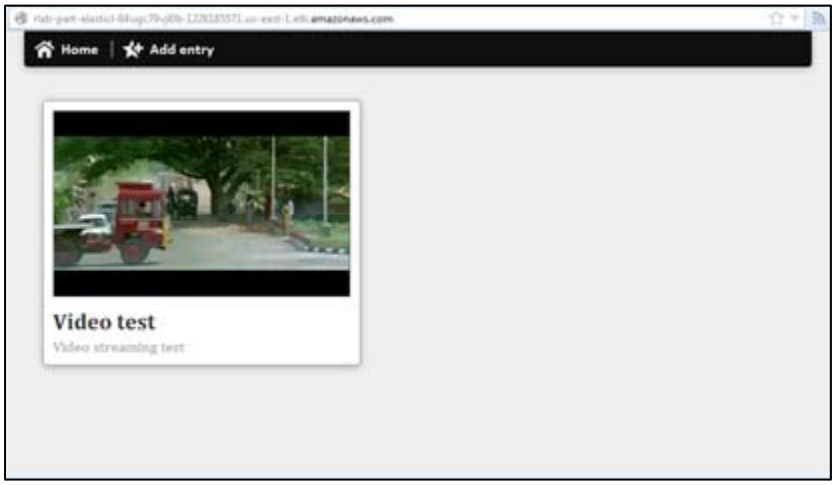
2. Click on the link below to open the webpage, then right-click on one or more of the links and save the .mp4 video file to a location on your local machine (i.e. Desktop or Downloads).

   https://us-east-1-aws-training.s3.amazonaws.com/self-paced-lab-11/videos.htm

Once the file upload is completed, the application shows “transcoding in progress”:

You can check in the AWS Console that the SQS queue shows a message being processed:

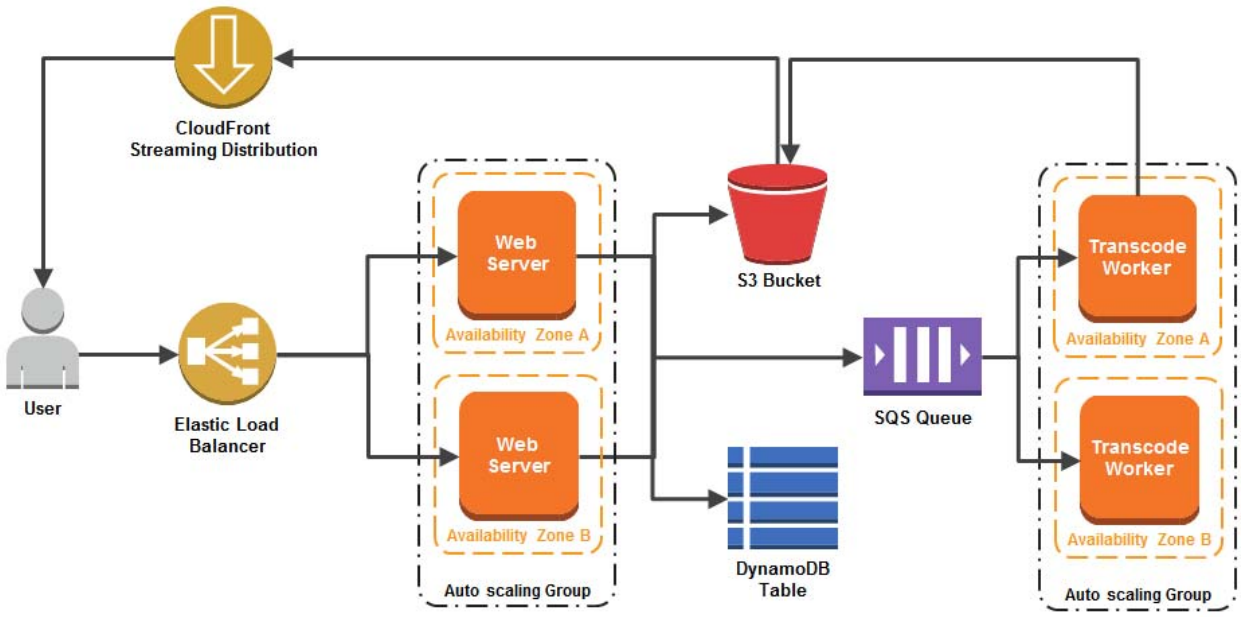
3. After a couple of seconds, **refresh the web application home page**. You should see a thumbnail of your video:



4. **Click on your video thumbnail** and you should be able to watch your video with a Video Streaming Flash viewer:

Final Architecture Overview

The following diagram represents the final architecture you built in this lab:



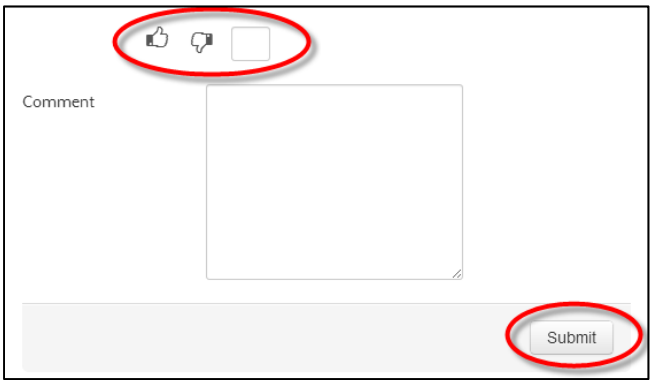
The front end web servers and the back end transcoders are both fault-tolerant and scalable. In case of increased inbound traffic, the Auto Scaling group will automatically provision new instances for the front end, and independently provision transcoder instances if there are many videos to manage. Data storage systems and the queue are fault-tolerant and scalable by design.

End Lab

This concludes the lab. The topic continues in *Lab 12 Part 2: Video Transcoding*.

1. **Sign out** of the AWS Management Console.
2. Click the **End Lab** button in *qwikLAB*™:

We appreciate your feedback. Please give the lab a thumbs-up/down, enter a comment and click **Submit**:



Any errors in this lab may be reported to aws-course-feedback@amazon.com.