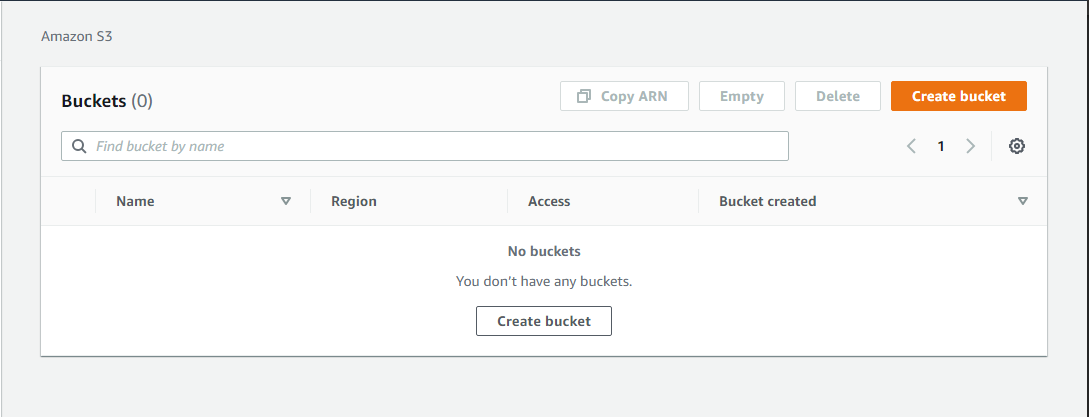
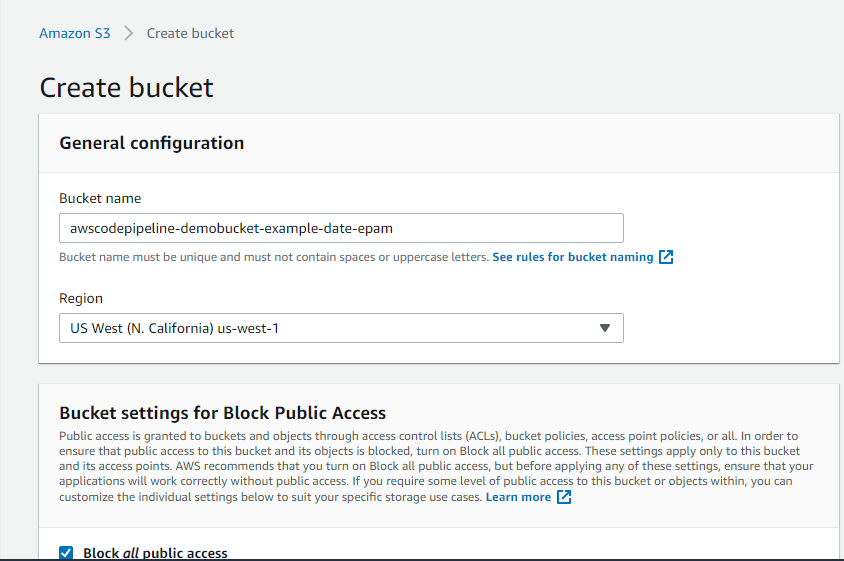
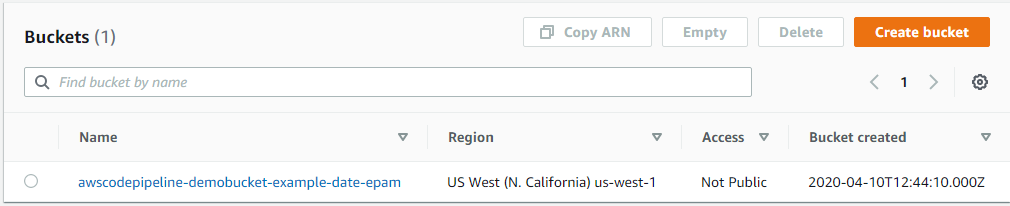
# Create a Simple Pipeline (S3 Bucket)

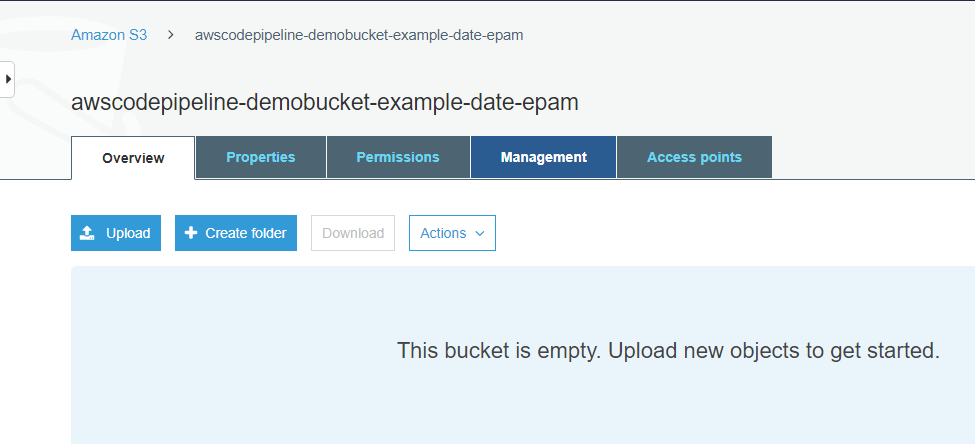
The easiest way to create a pipeline is to use the Create pipeline wizard in the AWS CodePipeline console.

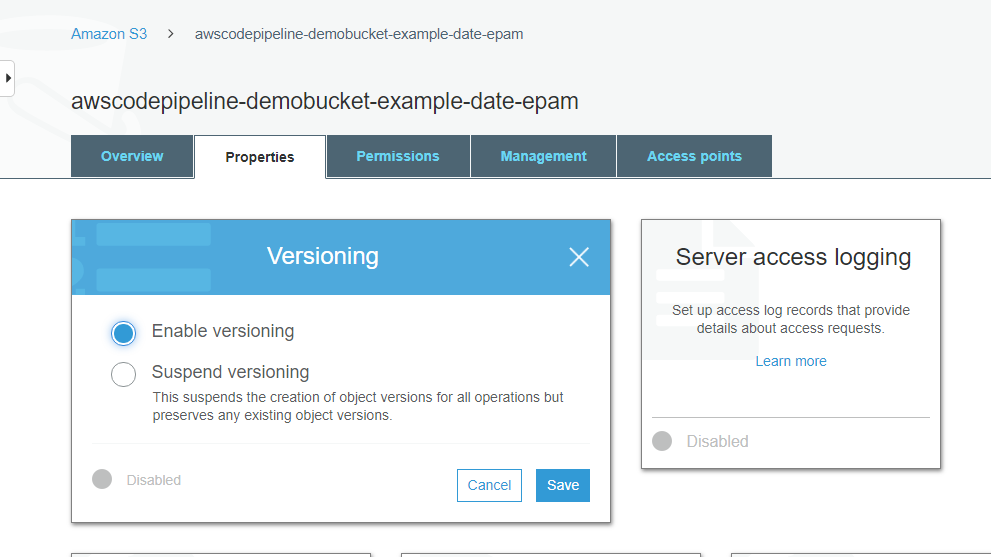
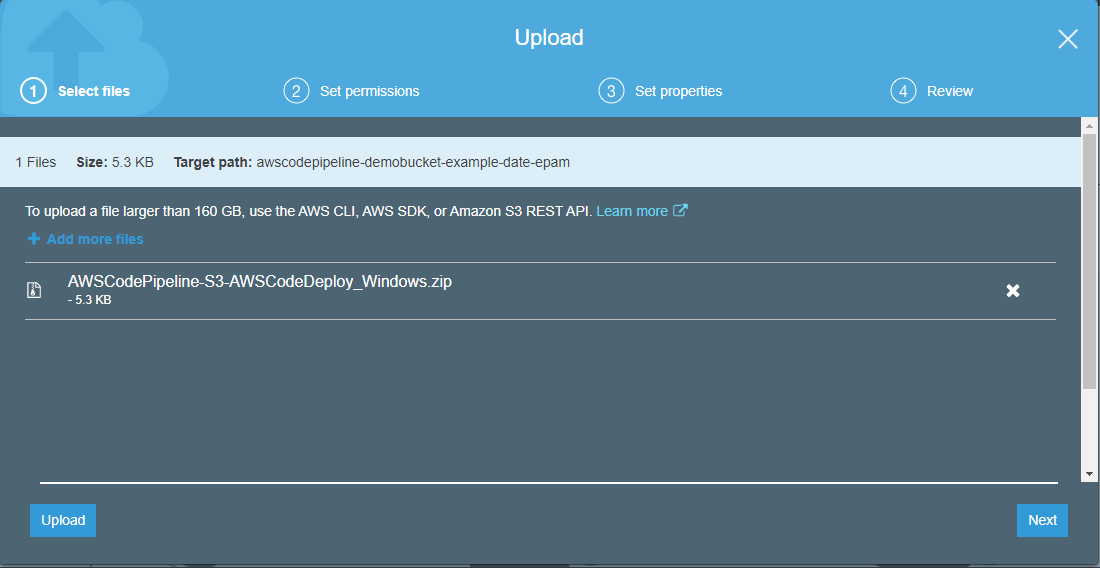
## Step 1: Create an S3 Bucket for Your Application

I can store my source files or applications in any versioned location. In this tutorial, I create an S3 bucket for the sample applications and enable versioning on that bucket. After I have enabled versioning, I copy the sample applications to that bucket.

### To create an S3 bucket

1. Sign into the AWS Management Console and open the Amazon S3 console 
2. Choose Create bucket.
3. In Bucket name, enter a name for your bucket 
4. After the bucket is created, a success banner displays. Choose Go to bucket details. 

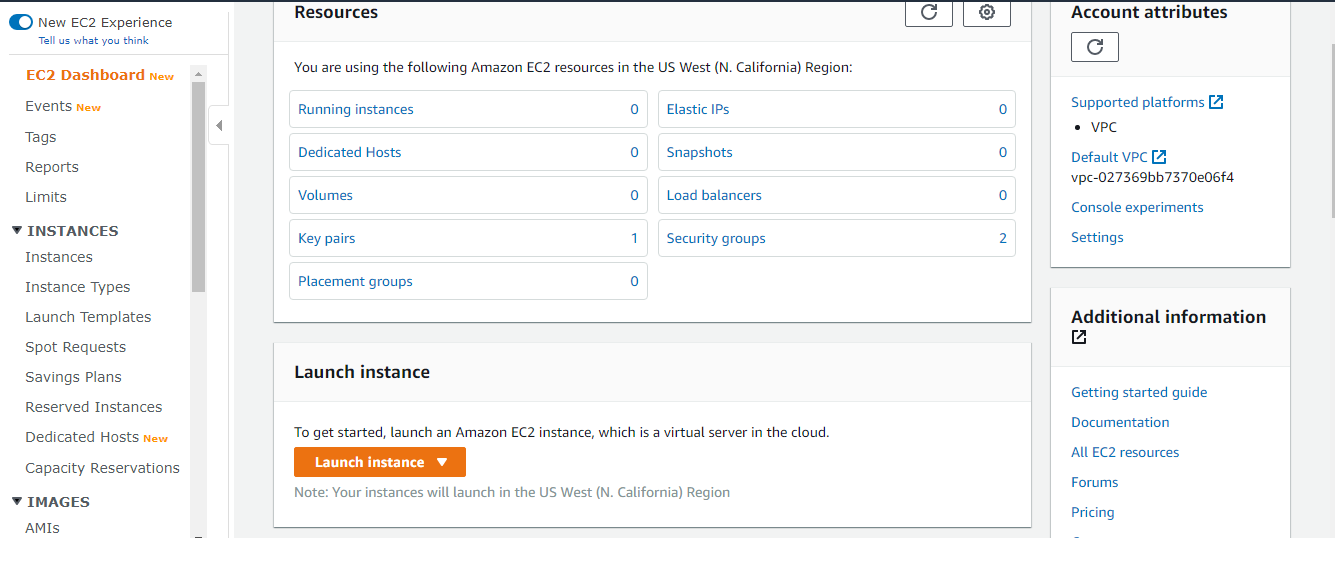
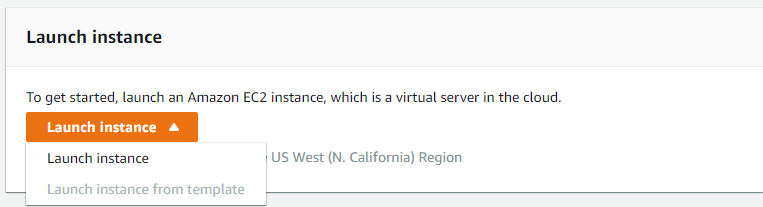
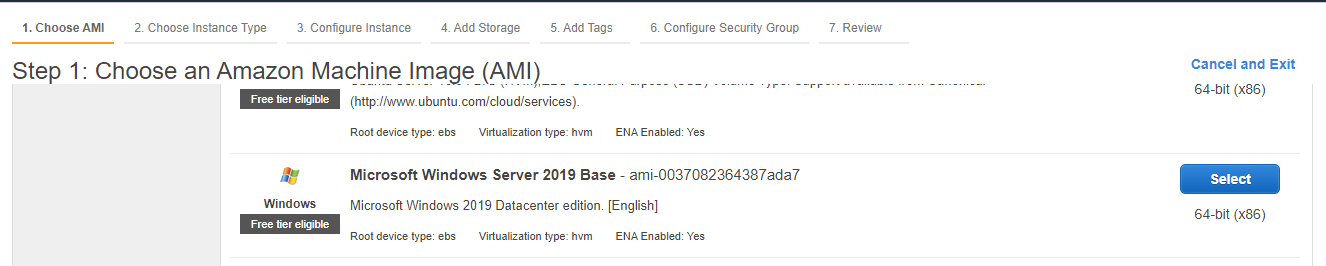
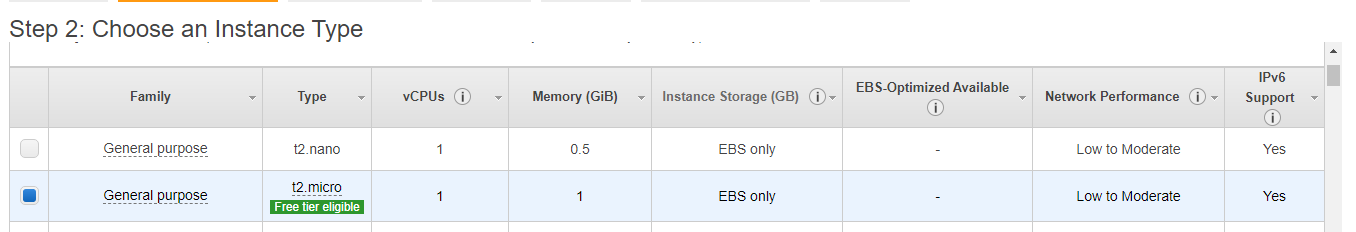


1. On the Properties tab, choose Versioning. Choose Enable versioning, and then choose Save. 
2. On the Permissions tab, leave the defaults.
3. Next, download a sample from a GitHub repository and save it into a folder or directory on your local computer.
   1. Open the GitHub repository that hosts the sample. To deploy to Windows Server instances using CodeDeploy, use the sample in <https://github.com/awslabs/AWSCodePipeline-S3-AWSCodeDeploy_Windows>.
   2. Choose the dist folder.
   3. Choose the file name. To deploy to Windows Server instances, use AWSCodePipeline-S3-AWSCodeDeploy\_Windows.zip.
   4. Choose View Raw, and then save the sample file to your local computer. Download the compressed (zipped) file. Do not unzip the file.
4. In the Amazon S3 console, for your bucket, upload the file: 

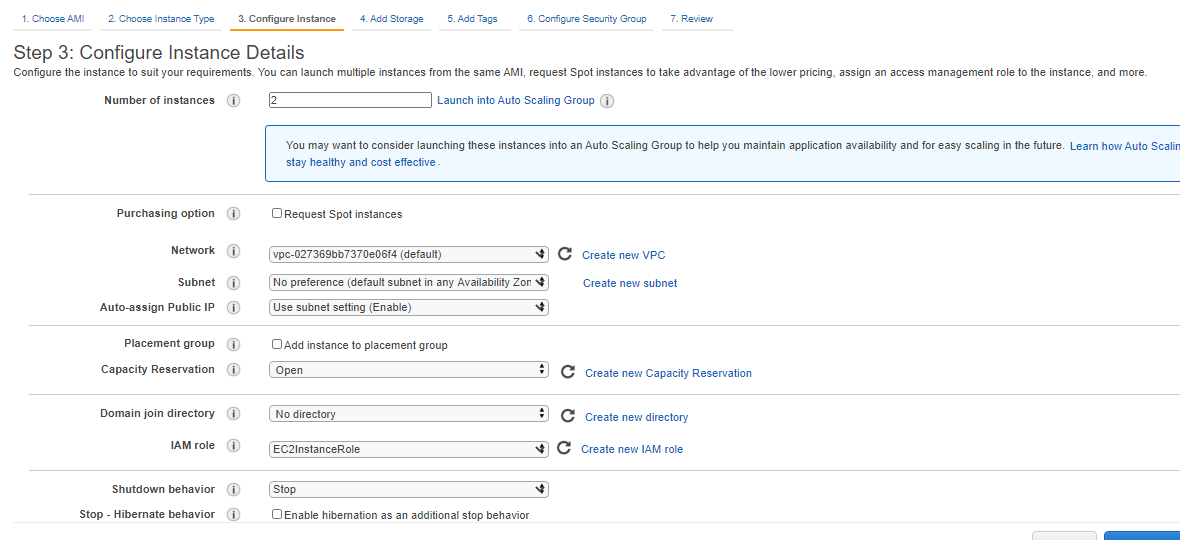
## Step 2: Create Amazon EC2 Windows Instances and Install the CodeDeploy Agent

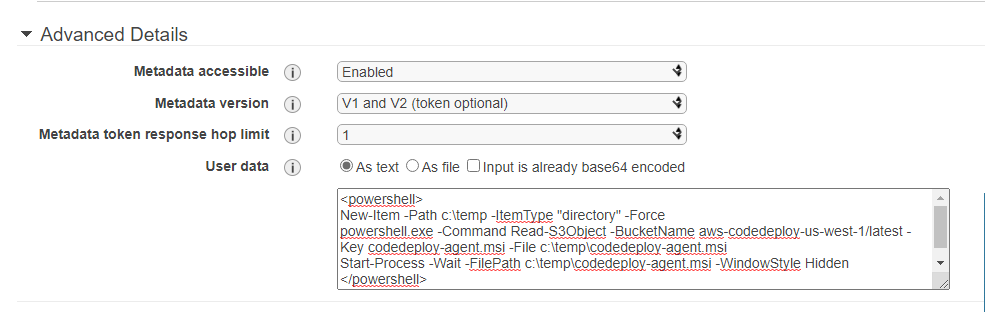
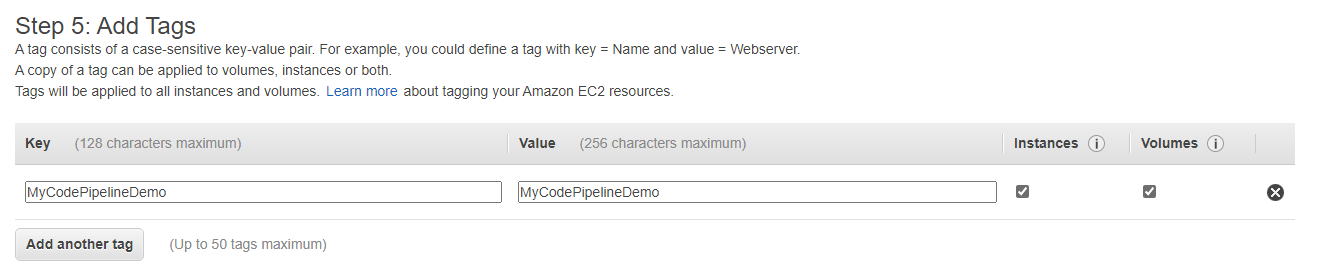
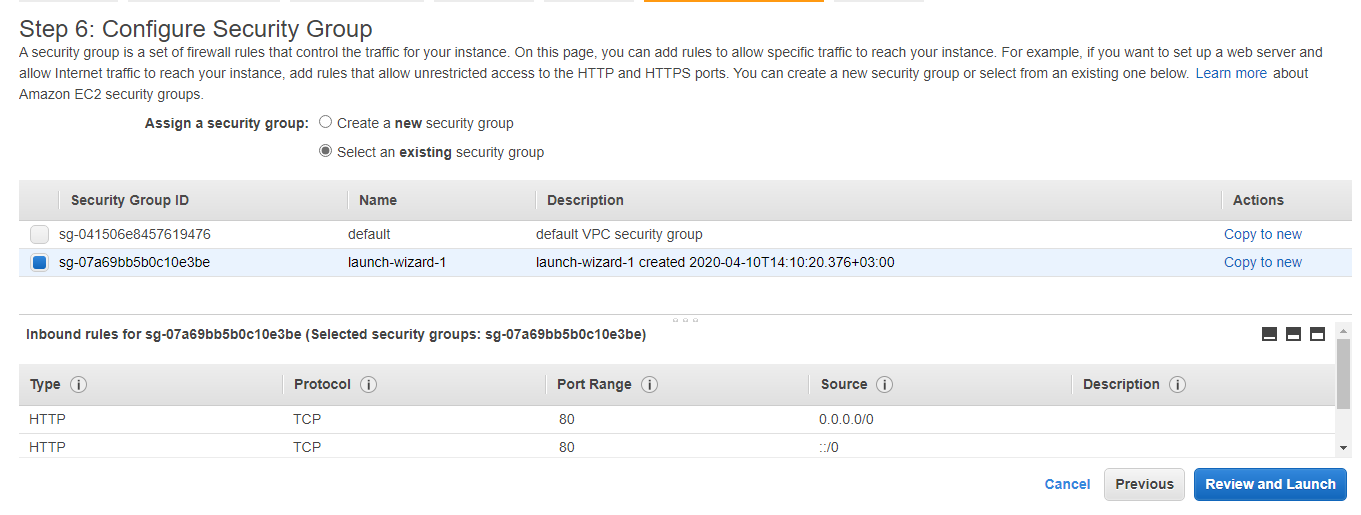
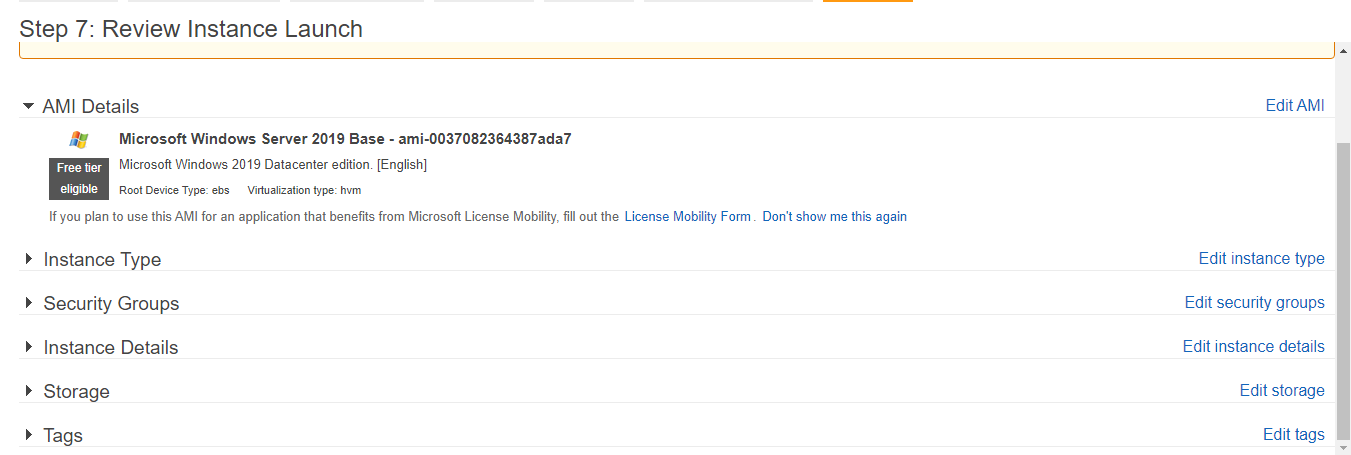
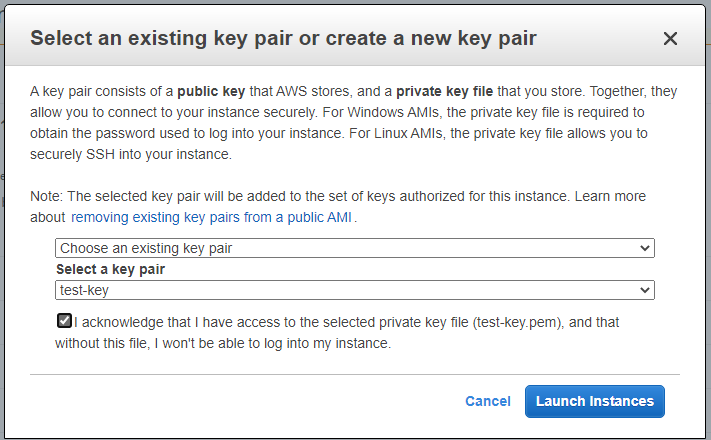
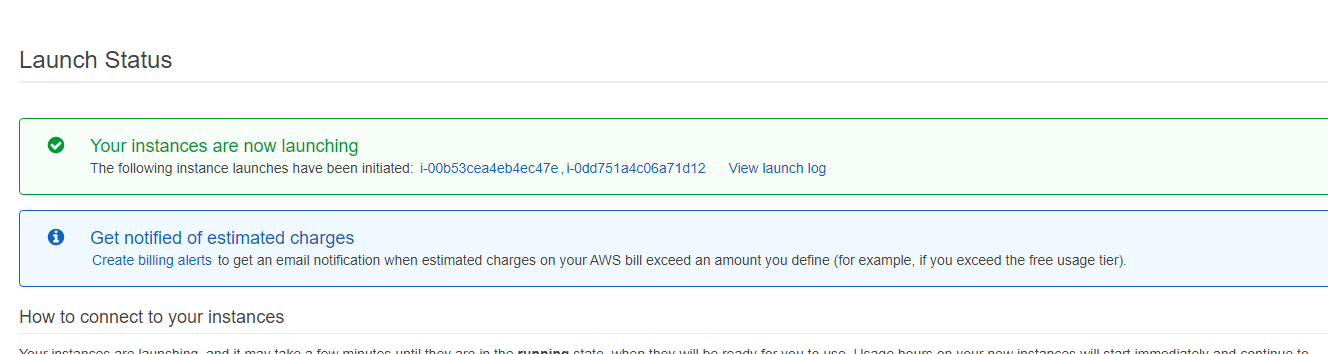
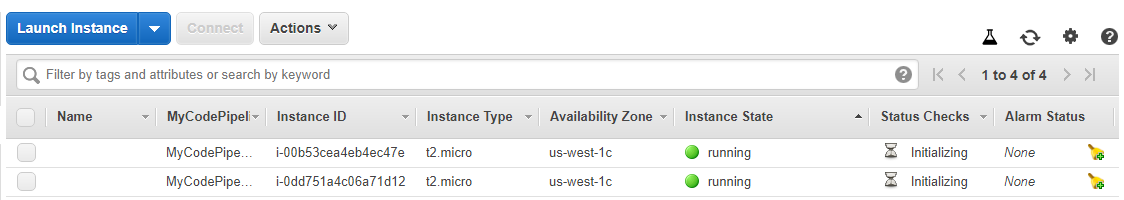
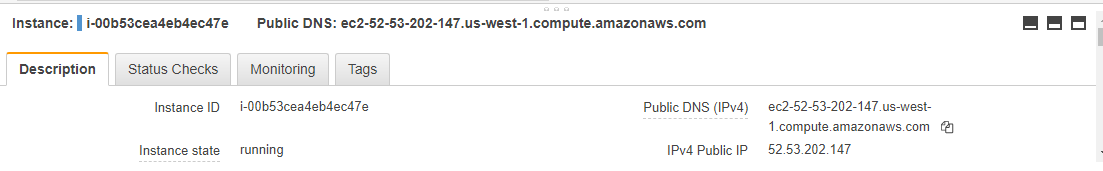
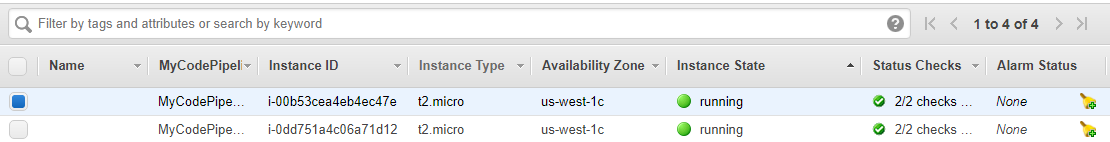
In this step, I create the Windows Server Amazon EC2 instances to which you will deploy a sample application. As part of this process, I install the CodeDeploy agent on the instances. The CodeDeploy agent is a software package that enables an instance to be used in CodeDeploy deployments.

### To launch instances

1. Open the Amazon EC2 console 
2. From the console dashboard, choose Launch instance, and select Launch instance from the options that pop up. 
3. On the Step 1: Choose an Amazon Machine Image (AMI) page, locate the Microsoft Windows Server 2019 Base option, and then choose Select. 
4. On the Step 2: Choose an Instance Type page, choose the free tier eligible t2.micro type as the hardware configuration for your instance, and then choose Next: Configure Instance Details. 
5. On the Step 3: Configure Instance Details page, do the following:

* In Number of instances, enter 2.
* In Auto-assign Public IP, choose Enable.
* In IAM role, choose an IAM role that has been configured for use as an IAM instance profile for use with CodeDeploy.

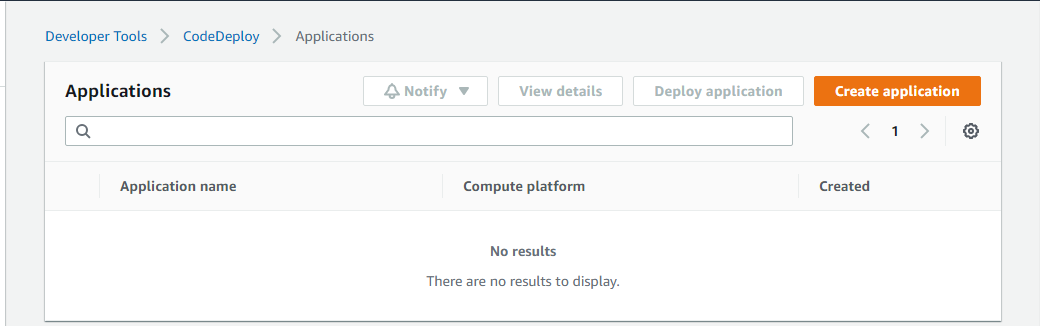
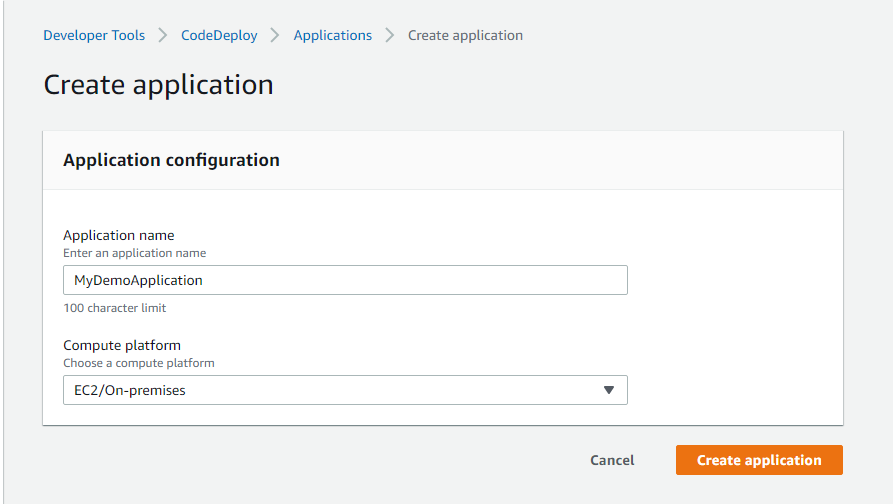


1. Expand Advanced Details, and in User data, with As text selected, enter the following: 
2. Leave the rest of the items on the Step 3: Configure Instance Details page unchanged. Choose Next: Add Storage, leave the Step 4: Add Storage page unchanged, and then choose Next: Add Tags.
3. On the Add Tags page, choose Add Tag. Enter Namein the Key field, enter MyCodePipelineDemo in the Value field, and then choose Next: Configure Security Group. 
4. On the Configure Security Group page, allow port 80 communication so you can access the public instance endpoint. 
5. Choose Review and Launch. 
6. On the Review Instance Launch page, choose Launch, and then do one of the following when prompted for a key pair: 
7. Choose View Instances to close the confirmation page and return to the console.  
8. You can view the status of the launch on the Instances page. When you launch an instance, its initial state is pending. After the instance starts, its state changes to running, and it receives a public DNS name. (If the Public DNS column is not displayed, choose the Show/Hide icon, and then select Public DNS.) 
9. It can take a few minutes for the instance to be ready for you to connect to it. Check that your instance has passed its status checks. You can view this information in the Status Checks column. 

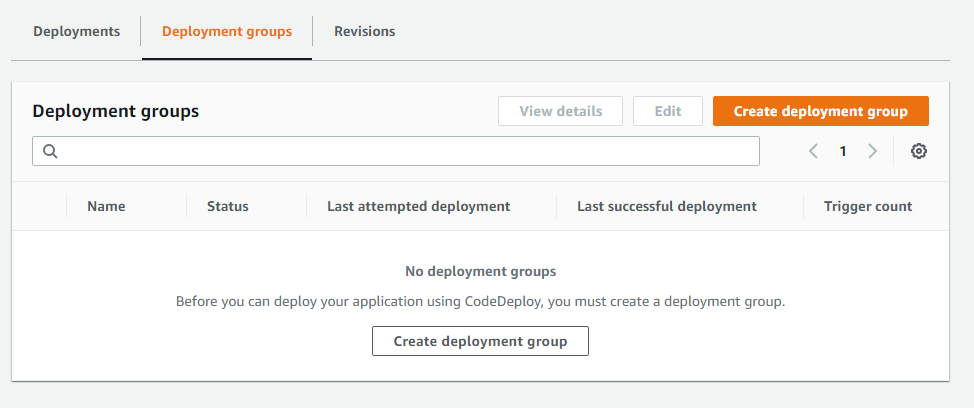
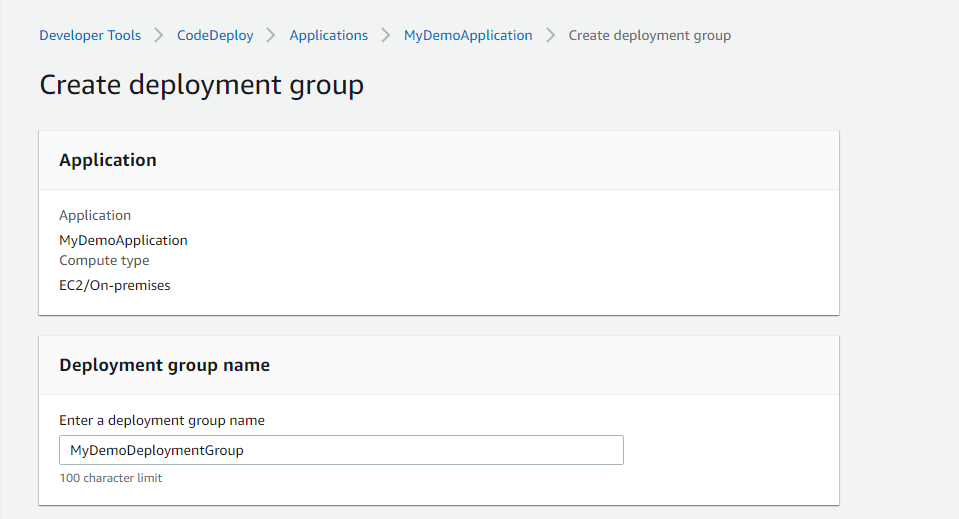
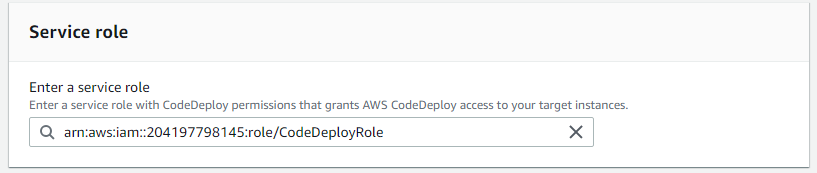
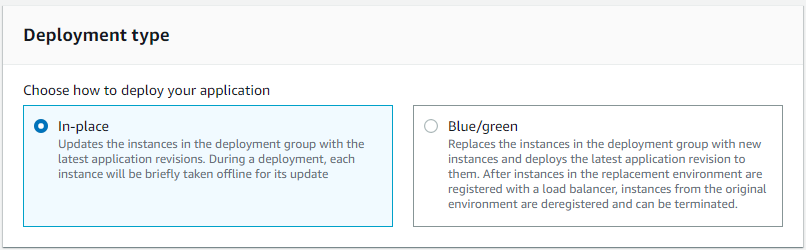
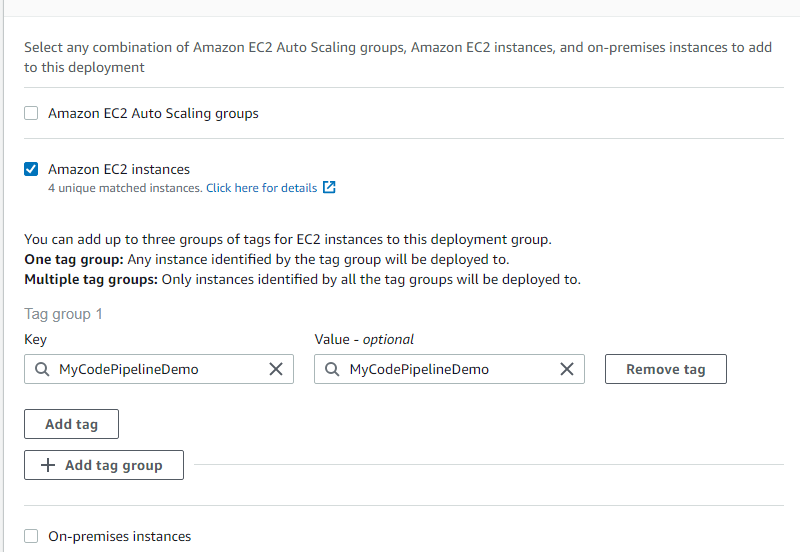
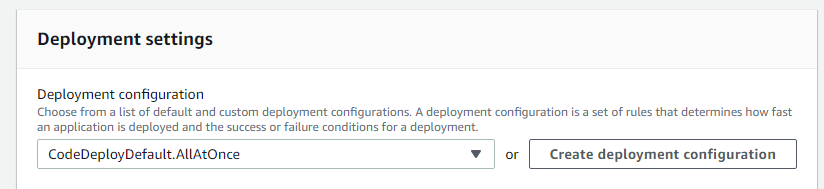
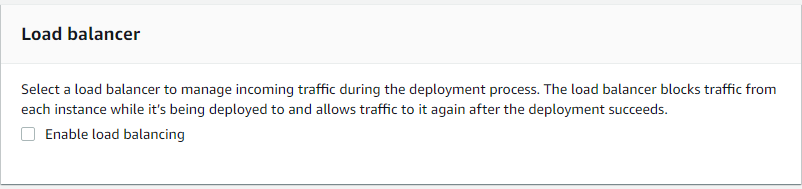
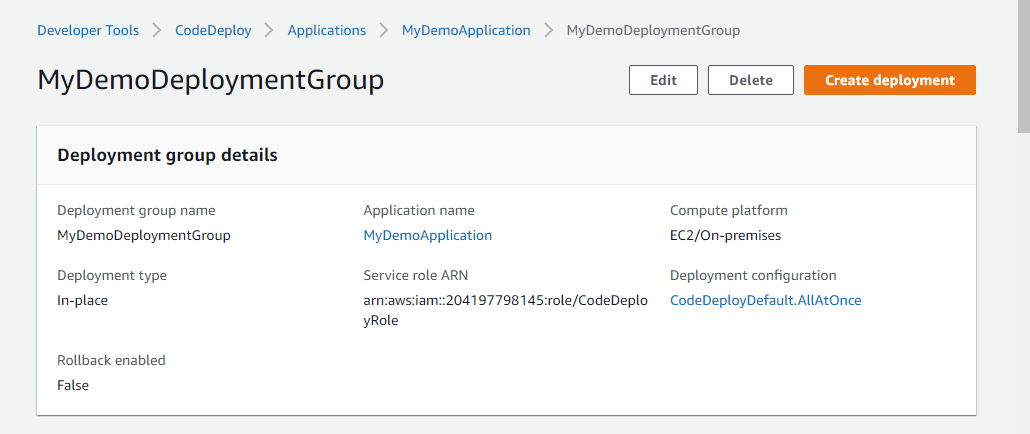
## Step 3: Create an Application in CodeDeploy

In CodeDeploy, an application is an identifier, in the form of a name, for the code I want to deploy. CodeDeploy uses this name to ensure the correct combination of revision, deployment configuration, and deployment group are referenced during a deployment. I select the name of the CodeDeploy application I create in this step when I create your pipeline later.

### To create an application in CodeDeploy

1. Open the CodeDeploy console. 
2. Choose Create application. 

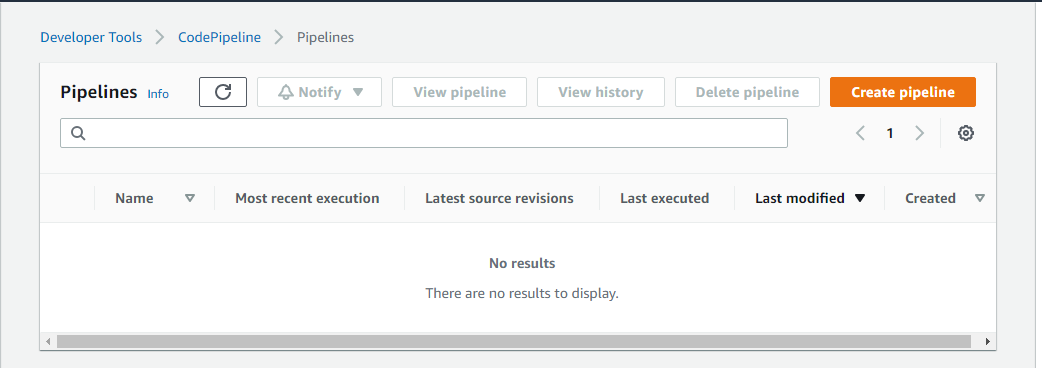
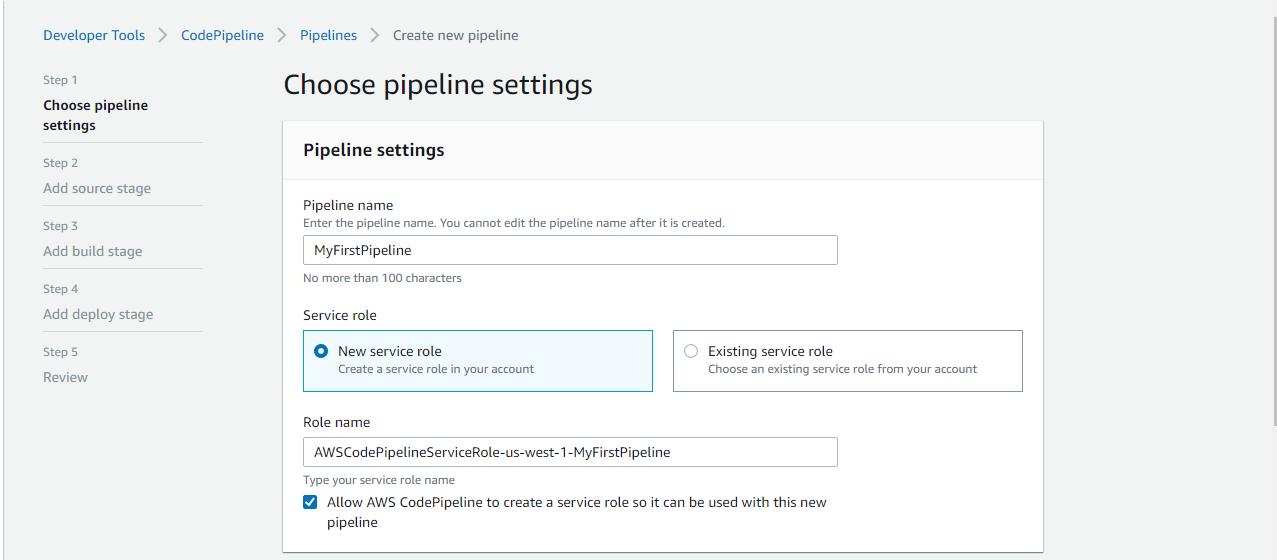
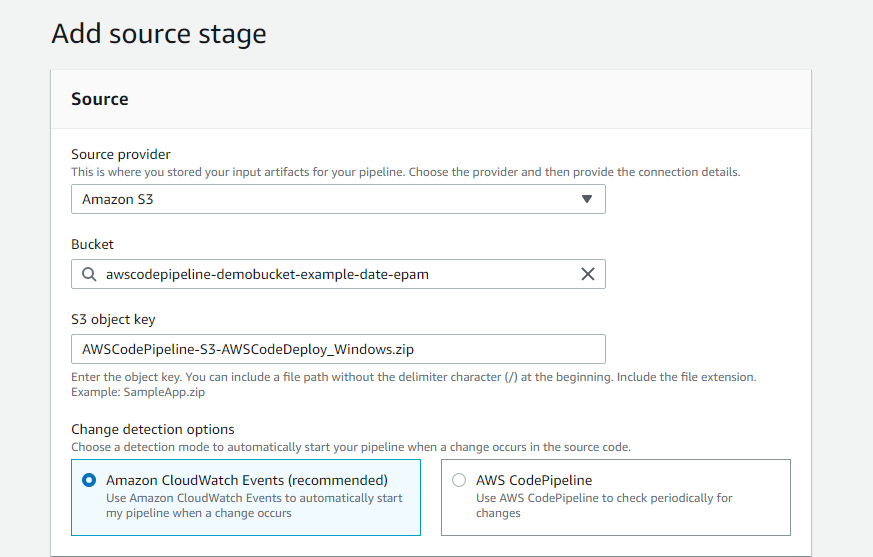
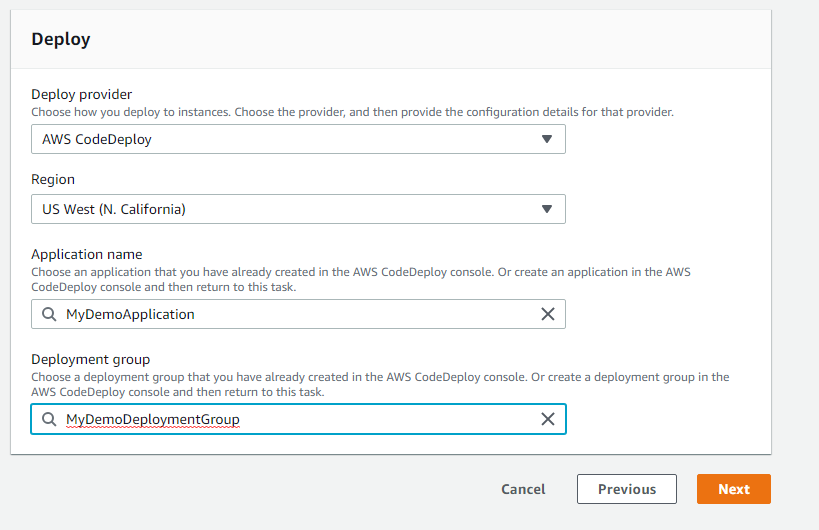
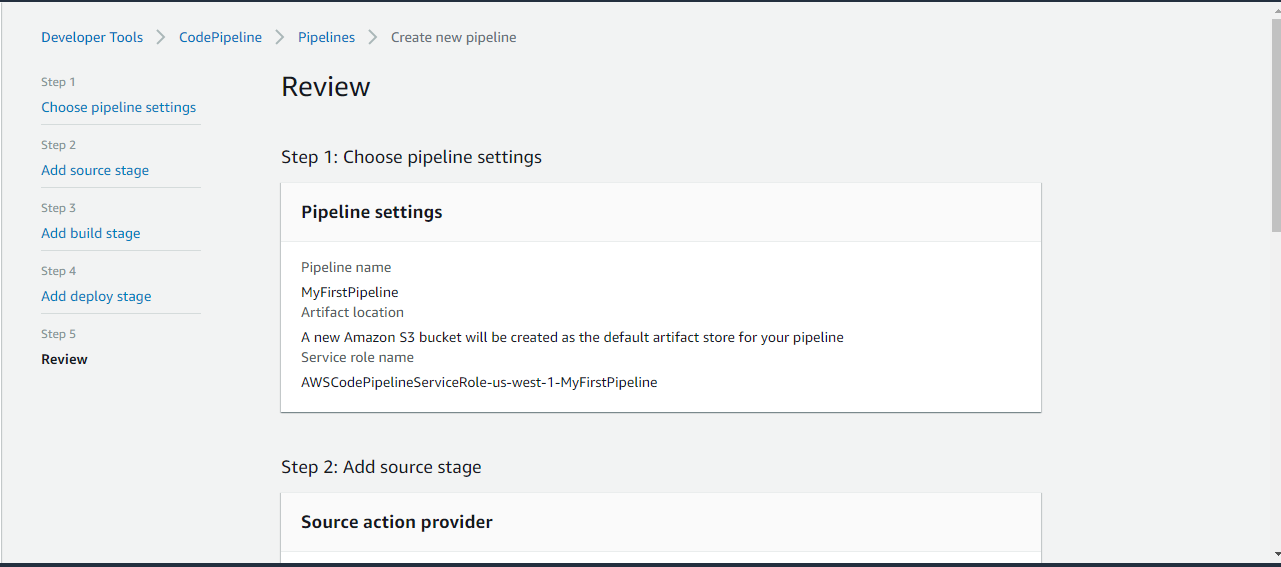
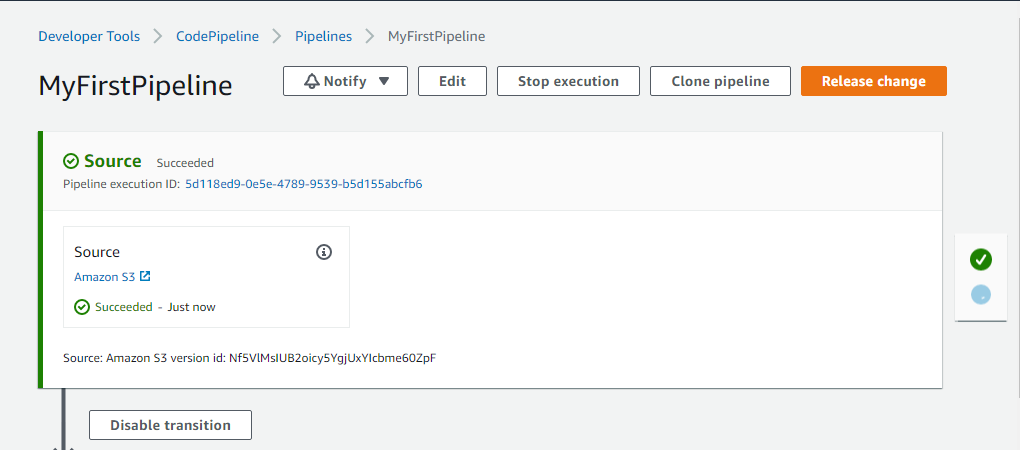
### To create a deployment group in CodeDeploy

1. On the page that displays your application, choose Create deployment group. 
2. In Deployment group name, enter MyDemoDeploymentGroup. 
3. In Service Role, choose a service role that trusts AWS CodeDeploy with, at minimum, the trust and permissions. 
4. Under Deployment type, choose In-place. 
5. Under Environment configuration, choose Amazon EC2 Instances. Choose Name in the Key field, and in the Value field, enter MyCodePipelineDemo. 
6. Under Deployment configuration, choose CodeDeployDefault.OneAtaTime. 
7. Under Load Balancer, clear Enable load balancing. 
8. Expand the Advanced section. Under Alarms, choose Ignore alarm configuration.
9. Choose Create deployment group. 

## Step 4: Create Your First Pipeline in CodePipeline

In this part, I create the pipeline. The sample runs automatically through the pipeline.

### To create a CodePipeline automated release process

1. Sign into the AWS Management Console and open the CodePipeline console 
2. In Step 1: Choose pipeline settings, in Pipeline name, enter MyFirstPipeline.
3. In Service role, do one of the following. Choose New service role to allow CodePipeline to create a new service role in IAM. 
4. Leave the settings under Advanced settings at their defaults, and then choose Next.
5. In Step 2: Add source stage, in Source provider, choose Amazon S3. In Bucket, enter the name of the S3 bucket you created in Step 1: Create an S3 Bucket for Your Application. In S3 object key, enter the object key with or without a file path, and remember to include the file extension. 
6. In Step 3: Add build stage, choose Skip build stage, and then accept the warning message by choosing Skip again. Choose Next.
7. In Step 4: Add deploy stage, in Deploy provider, choose AWS CodeDeploy. The Region field defaults to the same AWS Region as your pipeline. In Application name, enter MyDemoApplication, or choose the Refresh button, and then choose the application name from the list. In Deployment group, enter CodePipelineDemoFleet, or choose it from the list, and then choose Next. 
8. In Step 5: Review, review the information, and then choose Create pipeline. 
9. The pipeline starts to run. I can view progress and success and failure messages as the CodePipeline sample deploys a webpage to each of the Amazon EC2 instances in the CodeDeploy deployment.  

### To verify your pipeline ran successfully

1. View the initial progress of the pipeline. The status of each stage changes from No executions yet to In Progress, and then to either Succeeded or Failed. The pipeline should complete the first run within a few minutes.
2. After Succeeded is displayed for the action status, in the status area for the Staging stage, choose Details. This opens the AWS CodeDeploy console.
3. In the Deployment group tab, under Deployment lifecycle events, choose the instance ID. This opens the EC2 console.
4. On the Description tab, in Public DNS, copy the address, and then paste it into the address bar of your web browser. View the index page for the sample application you uploaded to your S3 bucket. 