Tutorial: Create a four-stage pipeline

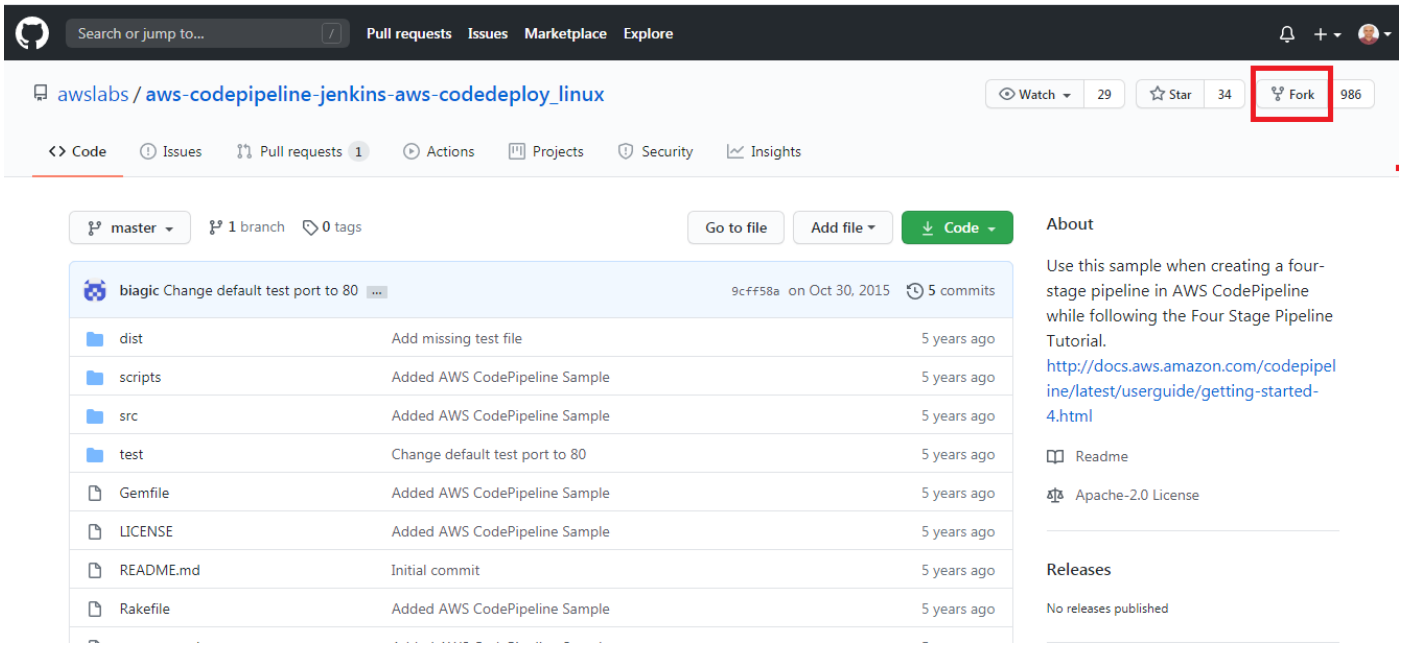
Topics

Step 1: Complete prerequisites

Copy or clone the sample into a GitHub repository

To clone the sample and push to a GitHub repository

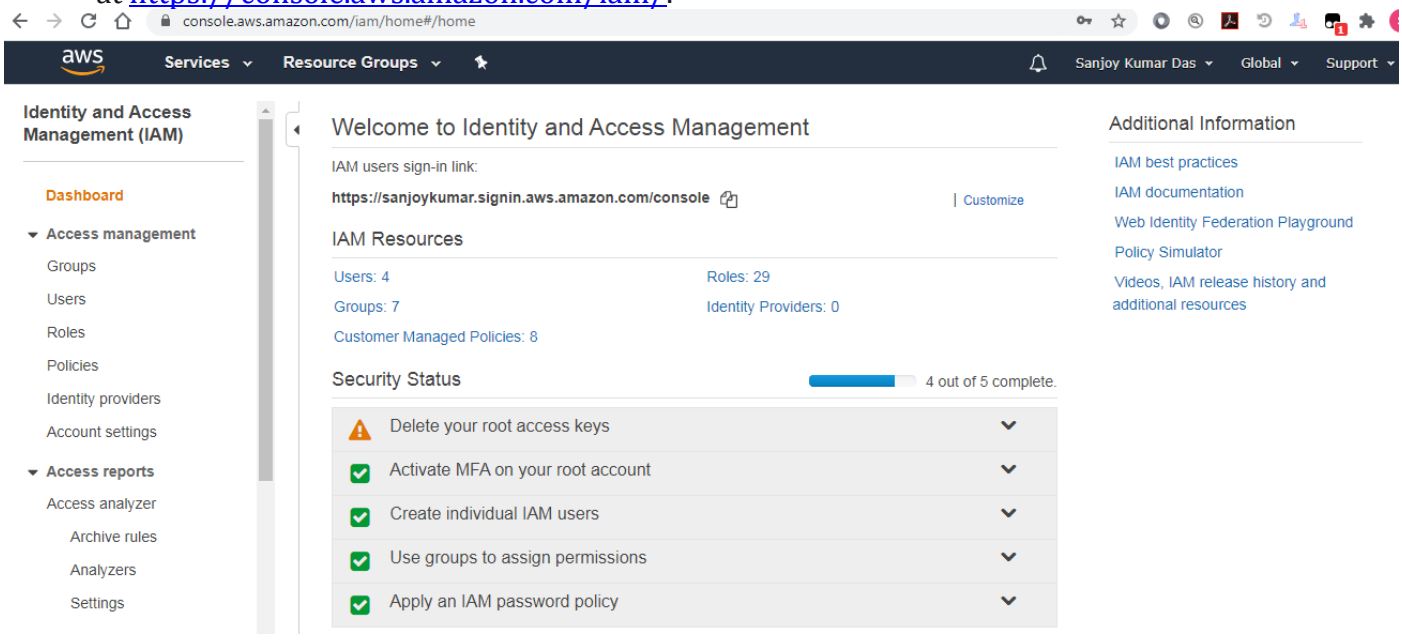
1. Download the sample code from the GitHub repository, or clone the repositories to your local computer. There are two sample packages:
    - If you will be deploying your sample to Amazon Linux, RHEL, or Ubuntu Server instances, choose aws-codepipeline-jenkins-aws-codedeploy_linux.zip.
    - If you will be deploying your sample to Windows Server instances, choose AWSCodePipeline-Jenkins-AWSCodeDeploy_Windows.zip.
2. From the repository, choose **Fork** to clone the sample repo into a repo in my Github account.



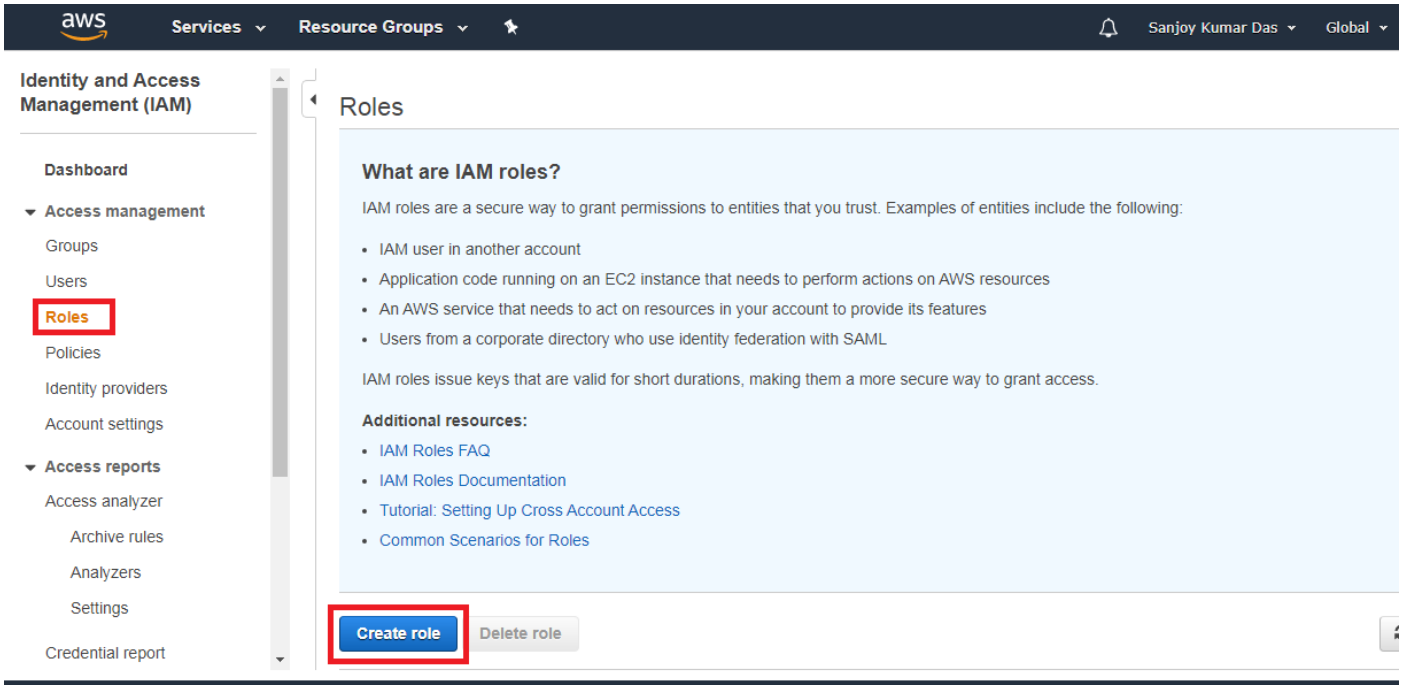
Create an IAM role to use for Jenkins integration

As a best practice, consider launching an EC2 instance to host your Jenkins server and using an IAM role to grant the instance the required permissions for interacting with CodePipeline.

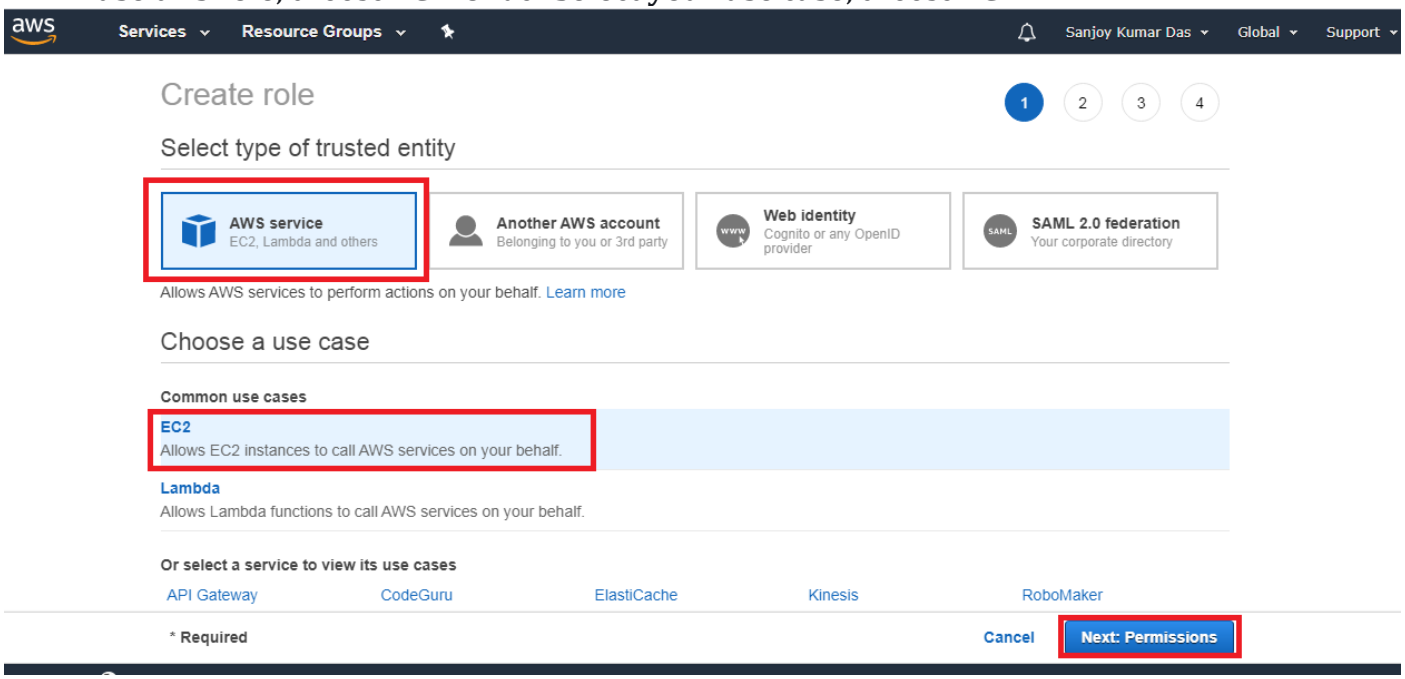
1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.



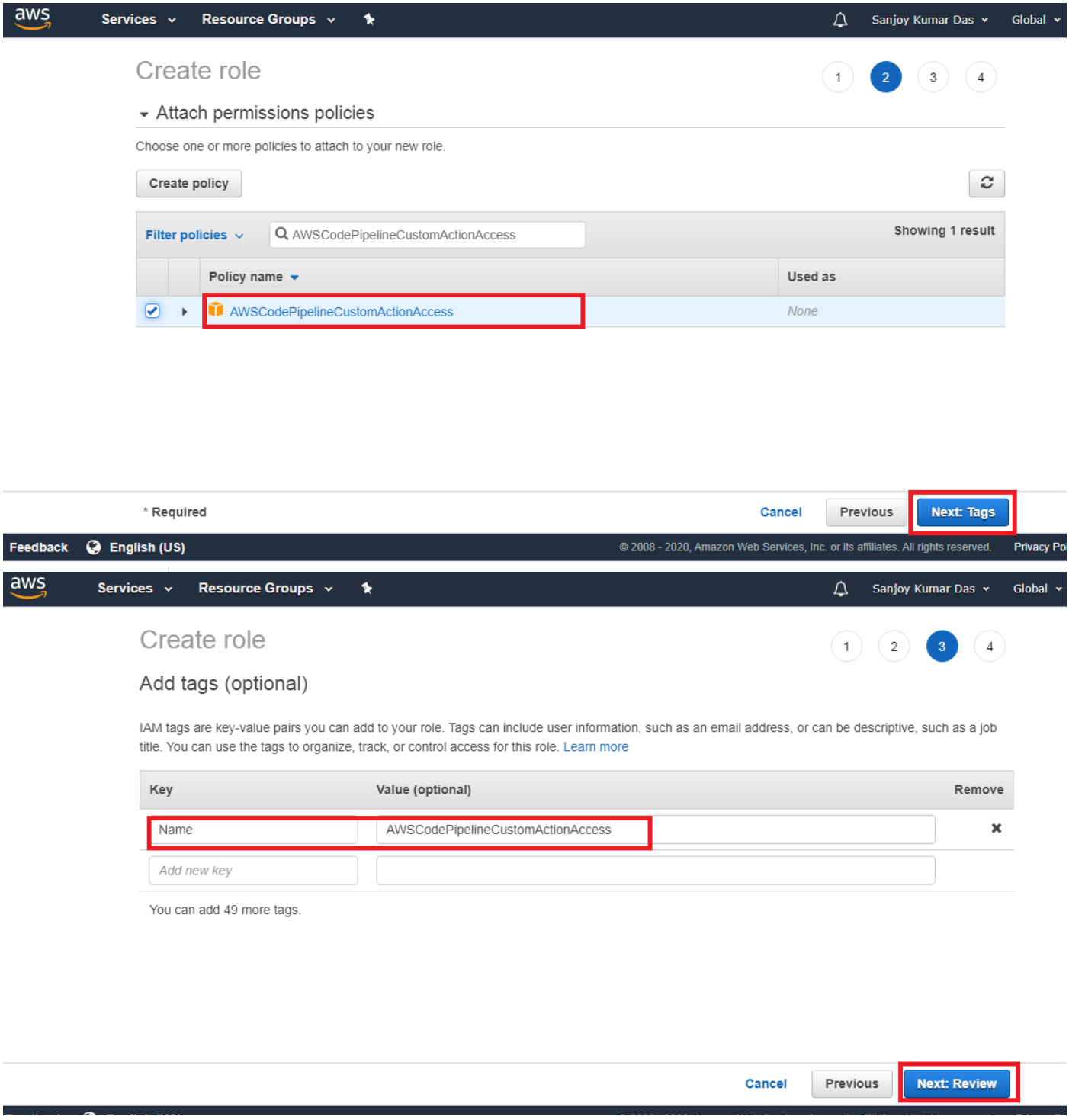
2. In the IAM console, in the navigation pane, choose **Roles**, and then choose **Create role**.



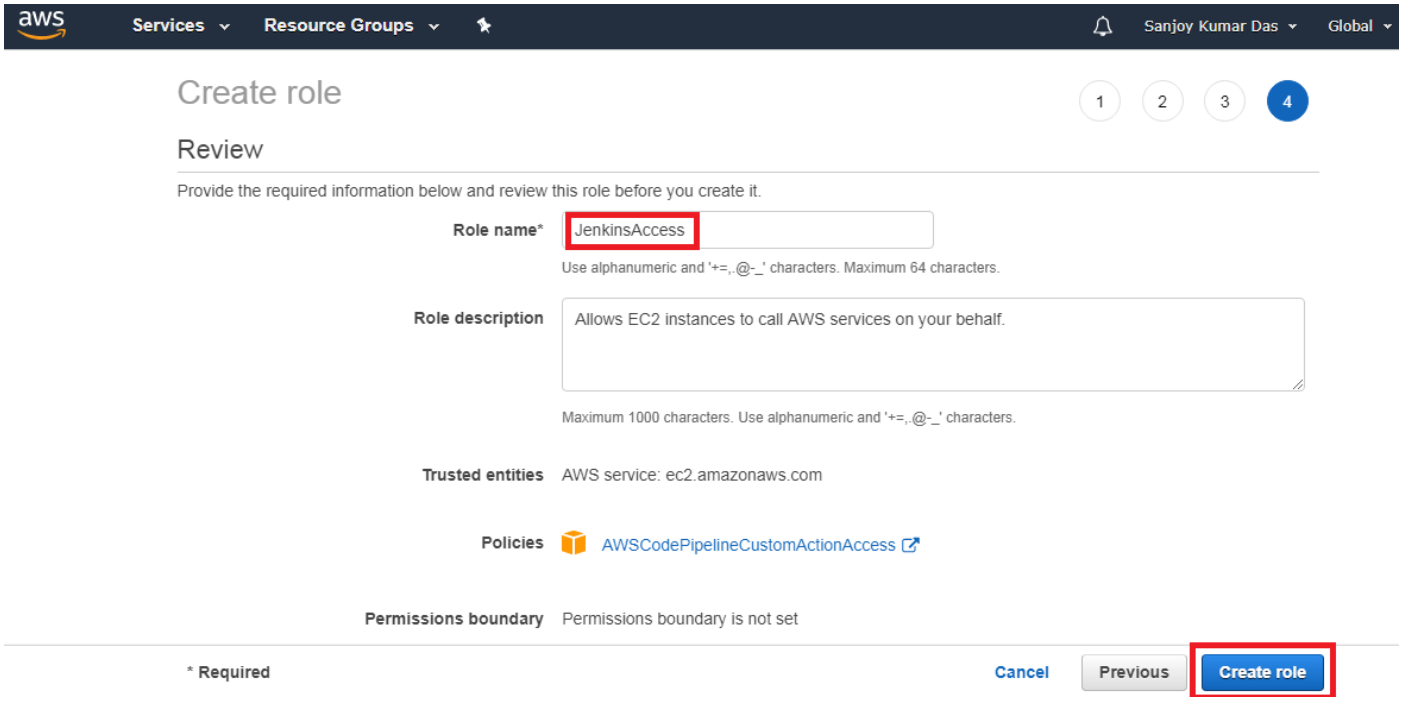
3. Under **Select type of trusted entity**, choose **AWS service**. Under **Choose the service that will use this role**, choose **EC2**. Under **Select your use case**, choose **EC2**.



4. Choose **Next: Permissions**. On the **Attach permissions policies** page, select the `AWSCodePipelineCustomActionAccess` managed policy, and then choose **Next: Tags**. Choose **Next: Review**.



5. On the **Review** page, in **Role name**, enter the name of the role to create specifically for Jenkins integration (for example, `JenkinsAccess`), and then choose **Create role**.

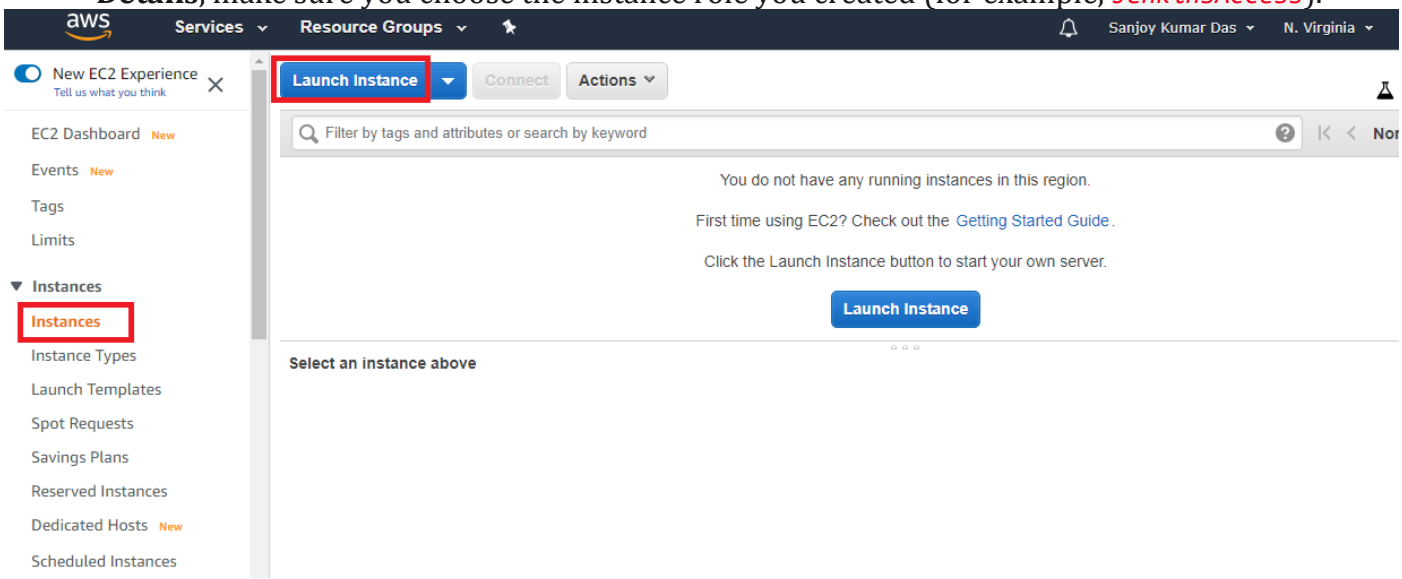


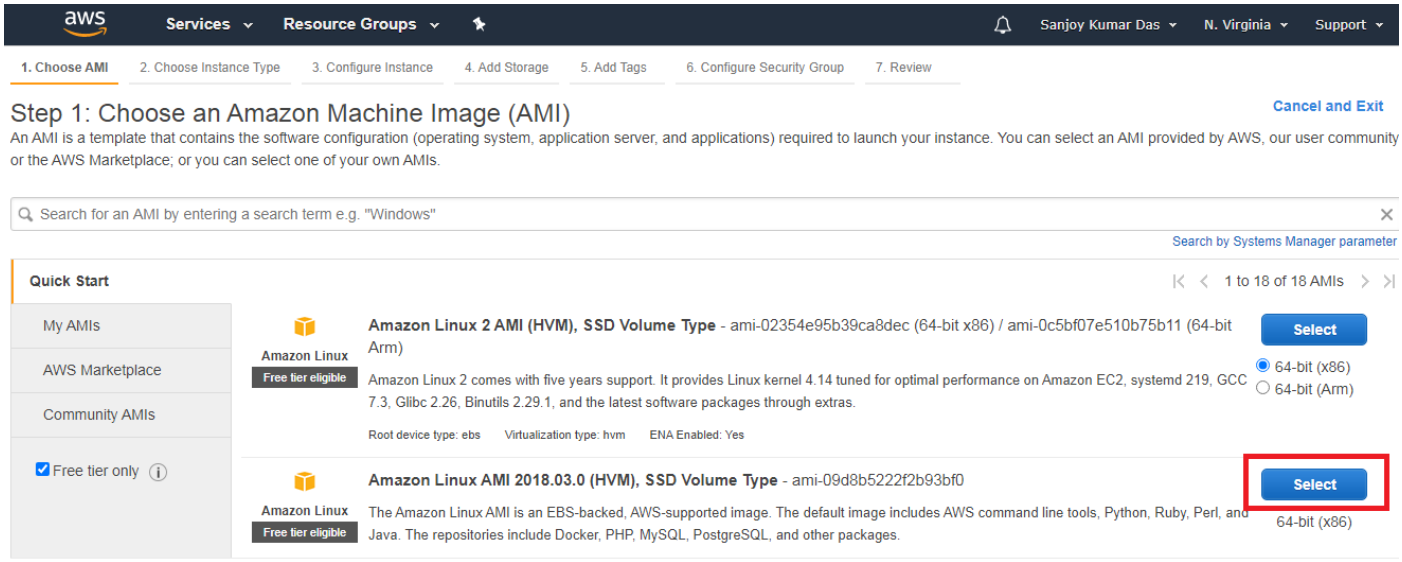
When you create the EC2 instance where you will install Jenkins, in **Step 3: Configure Instance Details**, make sure you choose the instance role (for example, `JenkinsAccess`).

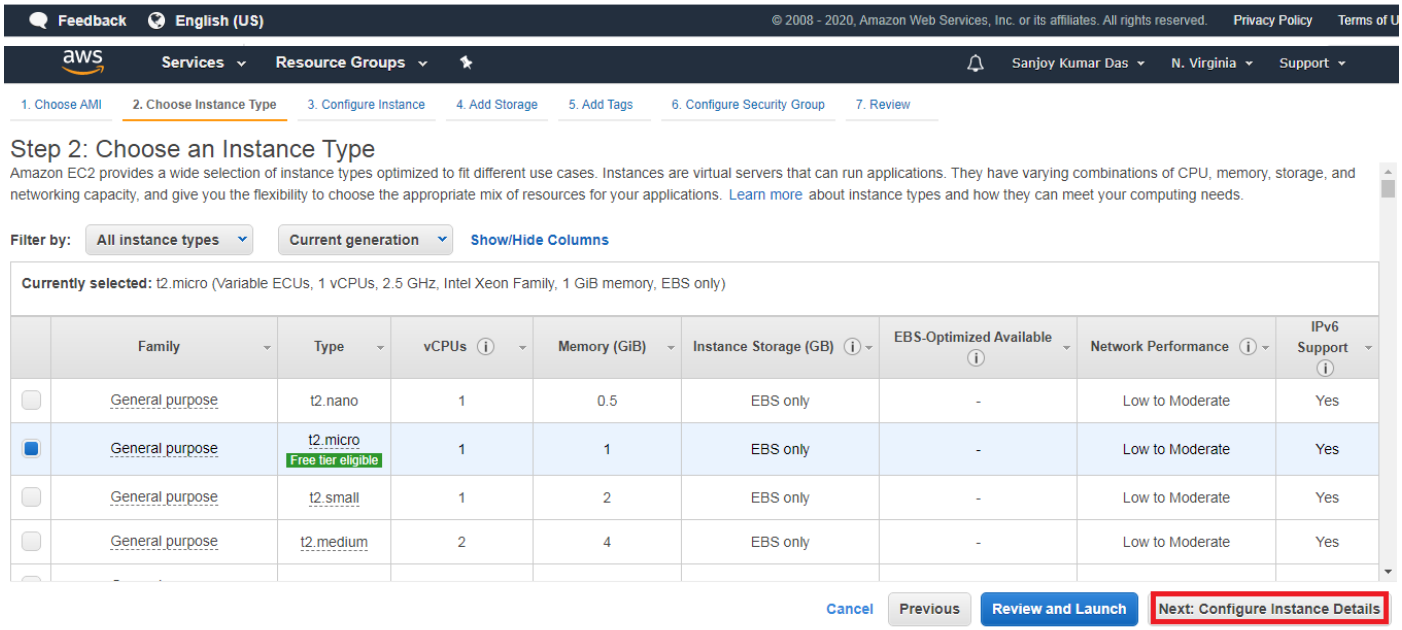
Install and configure Jenkins and the CodePipeline Plugin for Jenkins

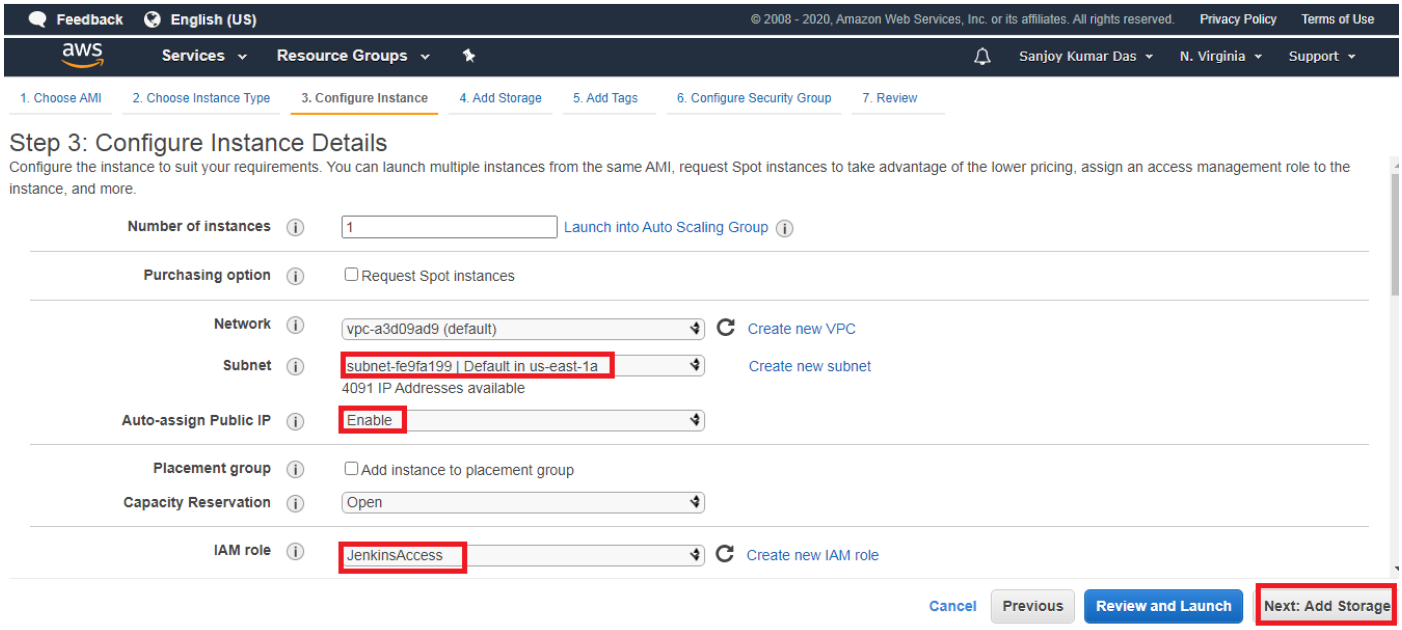
To install Jenkins and the CodePipeline Plugin for Jenkins

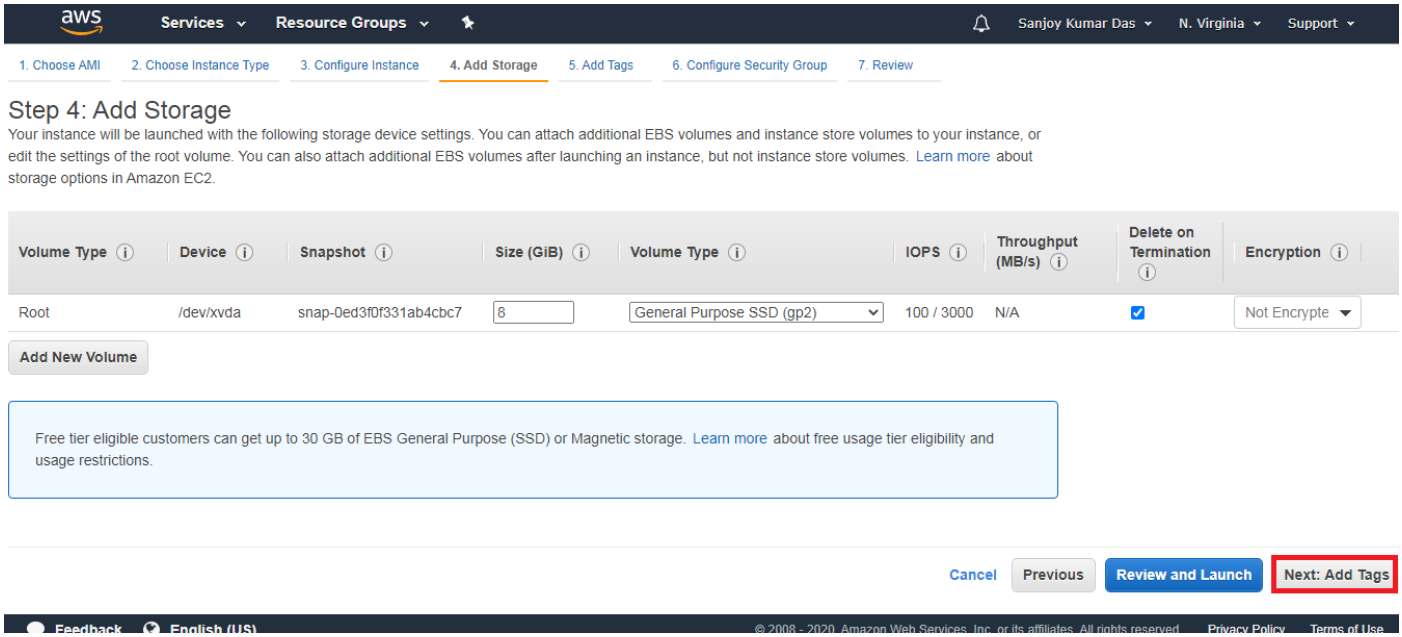
1. Create an EC2 instance where you will install Jenkins, and in **Step 3: Configure Instance Details**, make sure you choose the instance role you created (for example, `JenkinsAccess`).

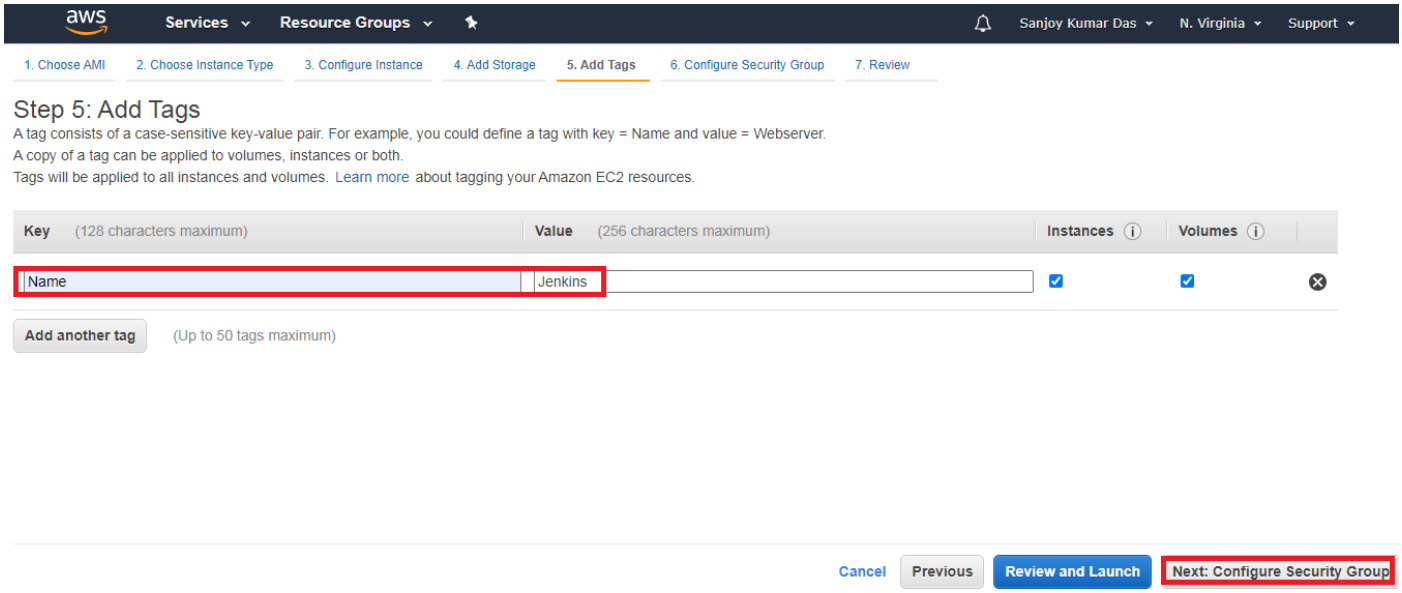


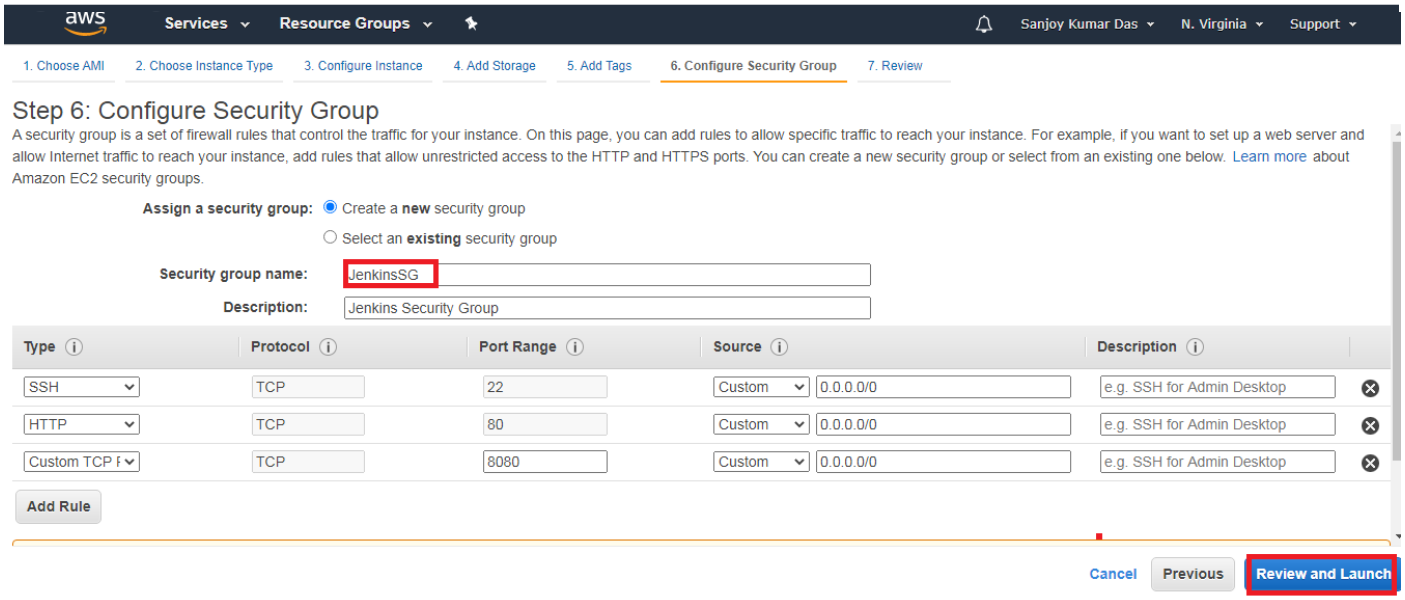

Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 1: Choose an Amazon Machine Image (AMI)
Cancel and Exit
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community or the AWS Marketplace; or you can select one of your own AMIs.
Search for an AMI by entering a search term e.g. "Windows"
Search by Systems Manager parameter
Quick Start
1 to 18 of 18 AMIs
My AMIs
AWS Marketplace
Community AMIs
Free tier only
Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-02354e95b39ca8dec (64-bit x86) / ami-0c5bf07e510b75b11 (64-bit Arm)
Select
64-bit (x86)
64-bit (Arm)
Amazon Linux
Free tier eligible
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
Root device type: ebs
Virtualization type: hvm
ENA Enabled: Yes
Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-09d8b5222f2b93bf0
Select
64-bit (x86)
Amazon Linux
Free tier eligible
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

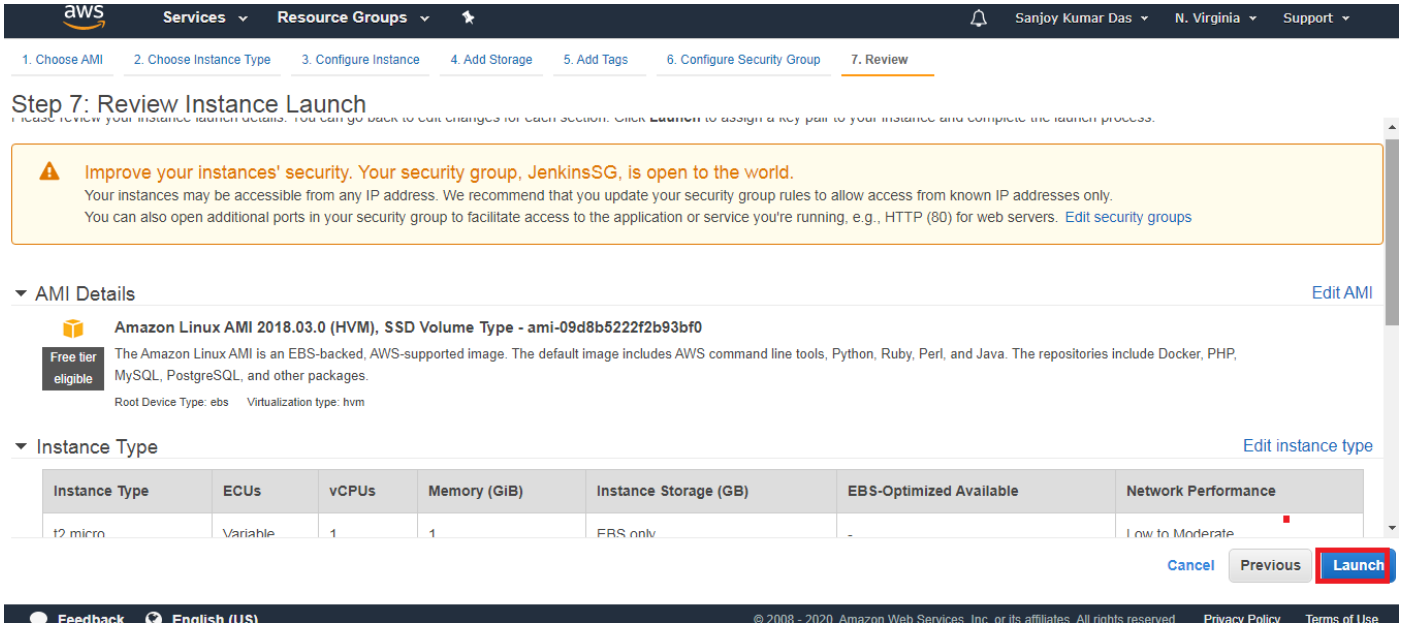

Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of U
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 2: Choose an Instance Type
Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.
Filter by:
All instance types
Current generation
Show/Hide Columns
Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)
Family
Type
vCPUs
Memory (GiB)
Instance Storage (GB)
EBS-Optimized Available
Network Performance
IPv6 Support
General purpose
t2.nano
1
0.5
EBS only
-
Low to Moderate
Yes
General purpose
t2.micro
Free tier eligible
1
1
EBS only
-
Low to Moderate
Yes
General purpose
t2.small
1
2
EBS only
-
Low to Moderate
Yes
General purpose
t2.medium
2
4
EBS only
-
Low to Moderate
Yes
Cancel
Previous
Review and Launch
Next: Configure Instance Details

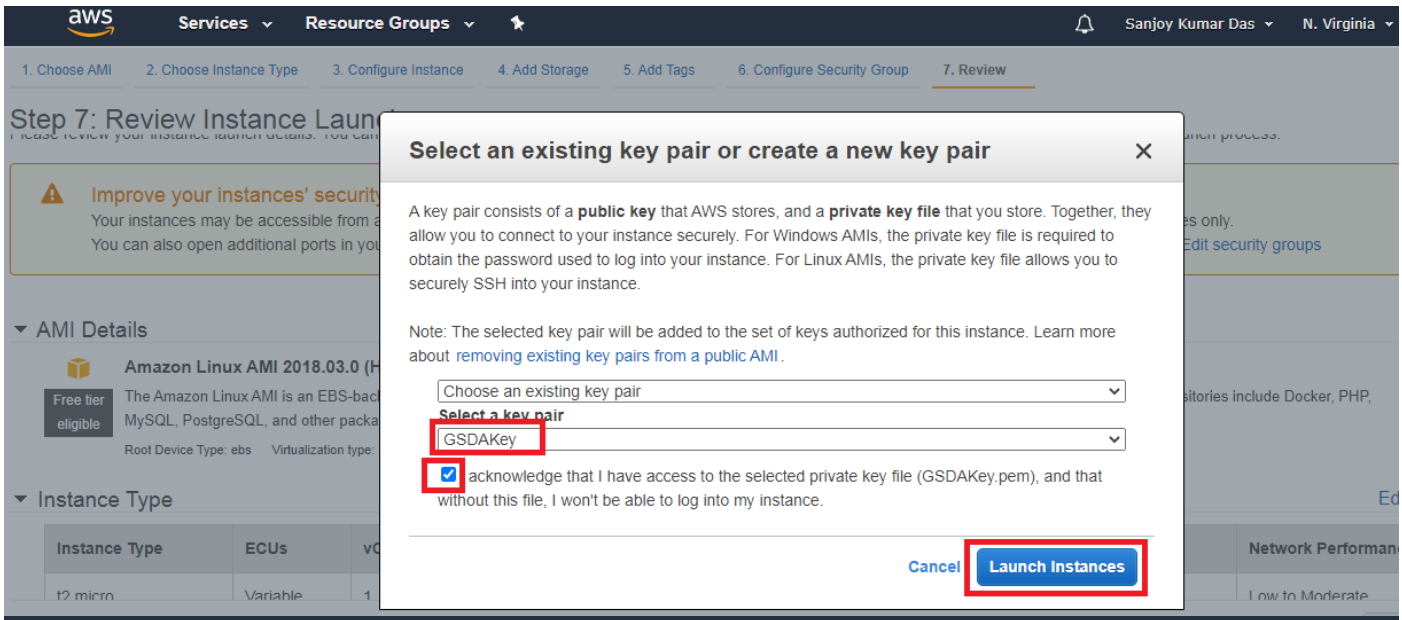

Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 3: Configure Instance Details
Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.
Number of instances
1
Launch into Auto Scaling Group
Purchasing option
Request Spot instances
Network
vpc-a3d09ad9 (default)
Create new VPC
Subnet
subnet-fe9fa199 | Default in us-east-1a
Create new subnet
4091 IP Addresses available
Auto-assign Public IP
Enable
Placement group
Add instance to placement group
Capacity Reservation
Open
IAM role
JenkinsAccess
Create new IAM role
Cancel
Previous
Review and Launch
Next: Add Storage

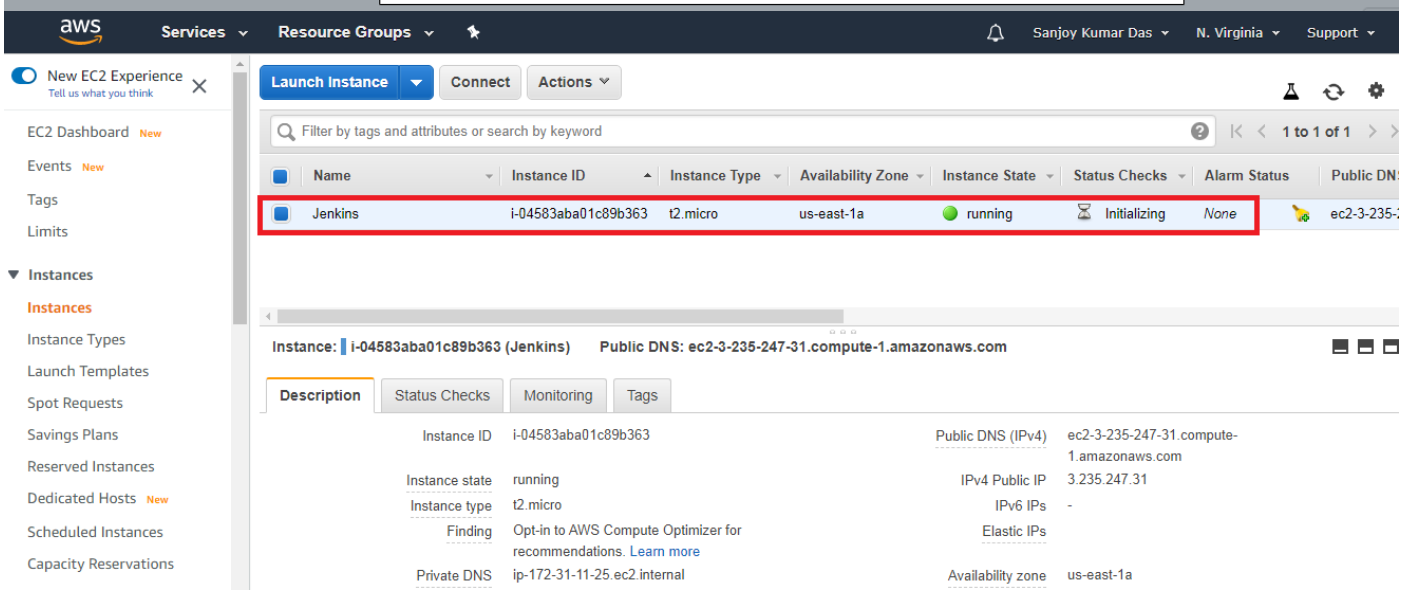

aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 4: Add Storage
Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.
Volume Type
Device
Snapshot
Size (GiB)
Volume Type
IOPS
Throughput (MB/s)
Delete on Termination
Encryption
Root
/dev/xvda
snap-0ed3f0f331ab4cbc7
8
General Purpose SSD (gp2)
100 / 3000
N/A
Not Encrypte
Add New Volume
Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.
Cancel
Previous
Review and Launch
Next: Add Tags
Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved
Privacy Policy
Terms of Use


aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 5: Add Tags
A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.
Key (128 characters maximum)
Value (256 characters maximum)
Instances
Volumes
Name
Jenkins
Add another tag
(Up to 50 tags maximum)
Cancel
Previous
Review and Launch
Next: Configure Security Group


aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 6: Configure Security Group
A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.
Assign a security group:
Create a new security group
Select an existing security group
Security group name:
JenkinsSG
Description:
Jenkins Security Group
Type
Protocol
Port Range
Source
Description
SSH
TCP
22
Custom
0.0.0.0/0
e.g. SSH for Admin Desktop
HTTP
TCP
80
Custom
0.0.0.0/0
e.g. SSH for Admin Desktop
Custom TCP
TCP
8080
Custom
0.0.0.0/0
e.g. SSH for Admin Desktop
Add Rule
Cancel
Previous
Review and Launch







2. Install Jenkins on the EC2 instance.

wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

#yum upgrade

yum install jenkins java-1.8.0-openjdk-devel

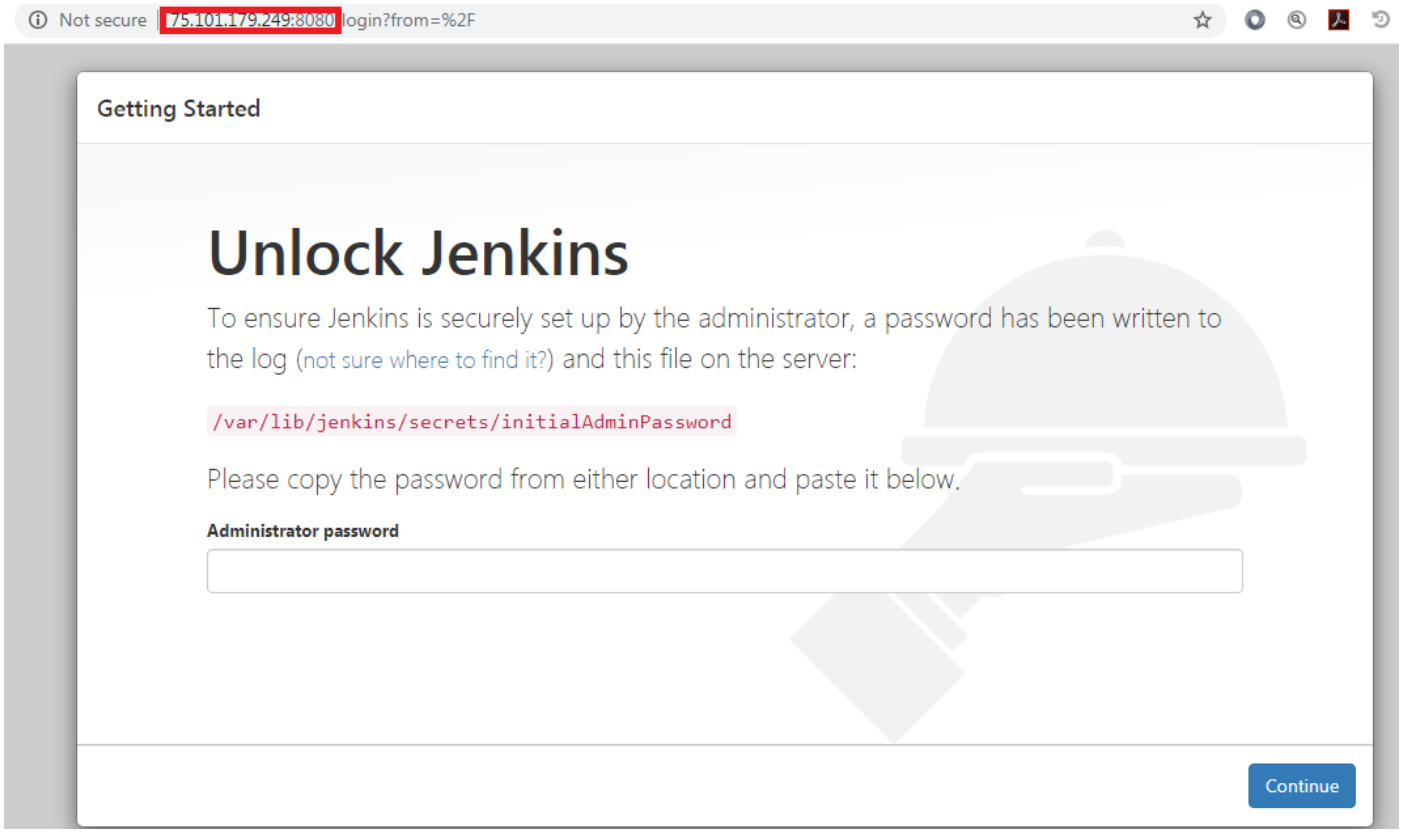
Start Jenkins: You can start the Jenkins service with the command:

sudo service jenkins start

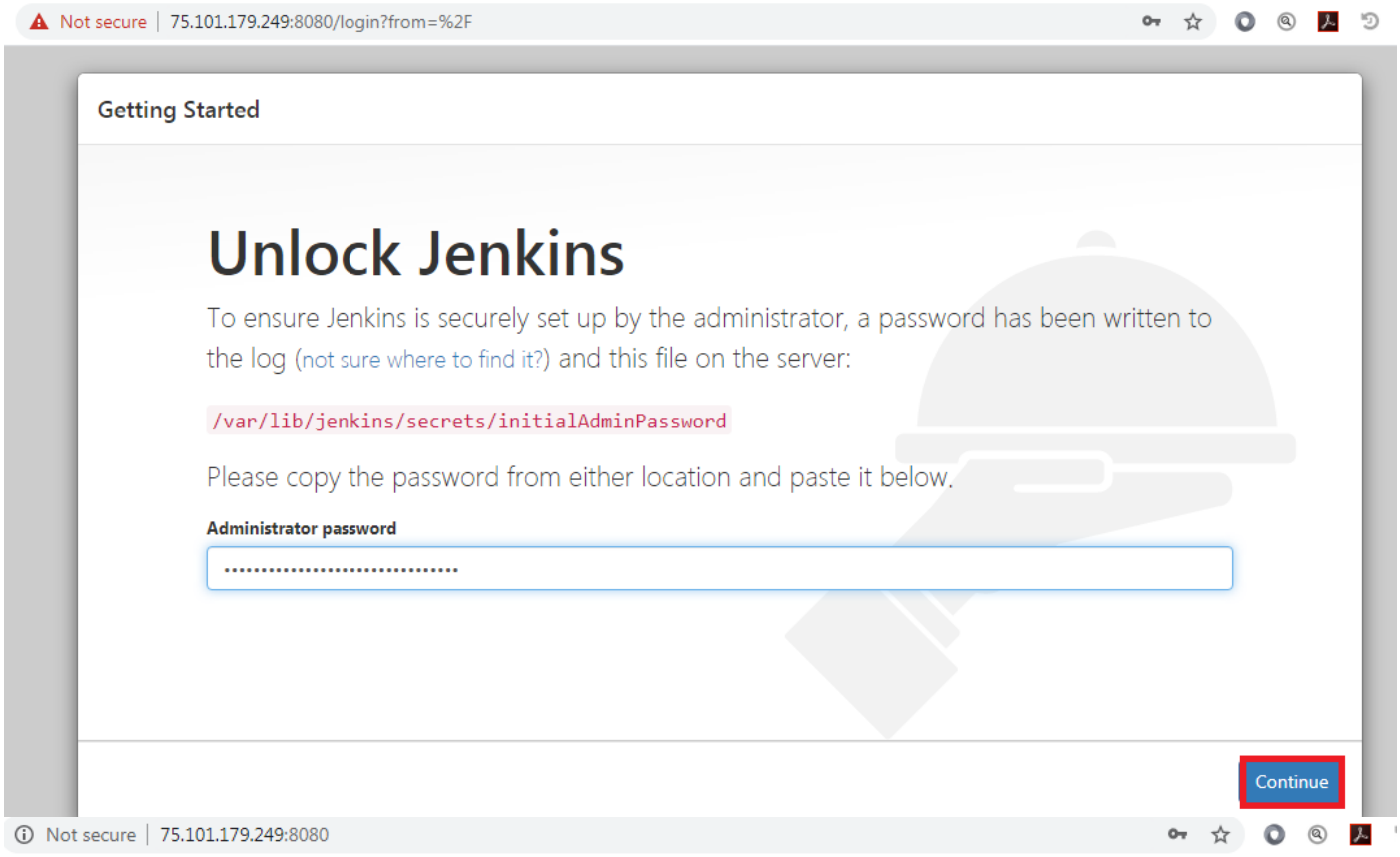
You can check the status of the Jenkins service using the command:

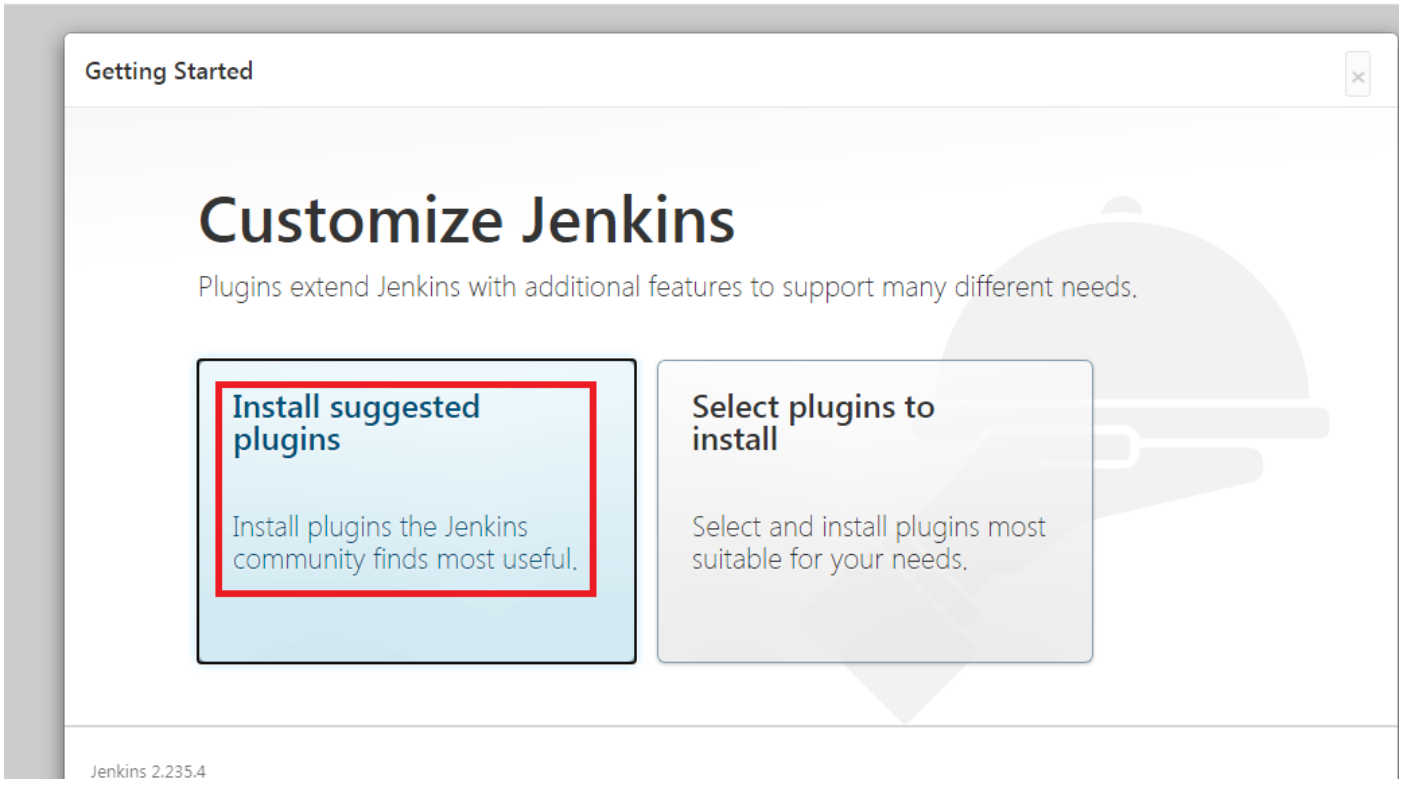
sudo service jenkins status

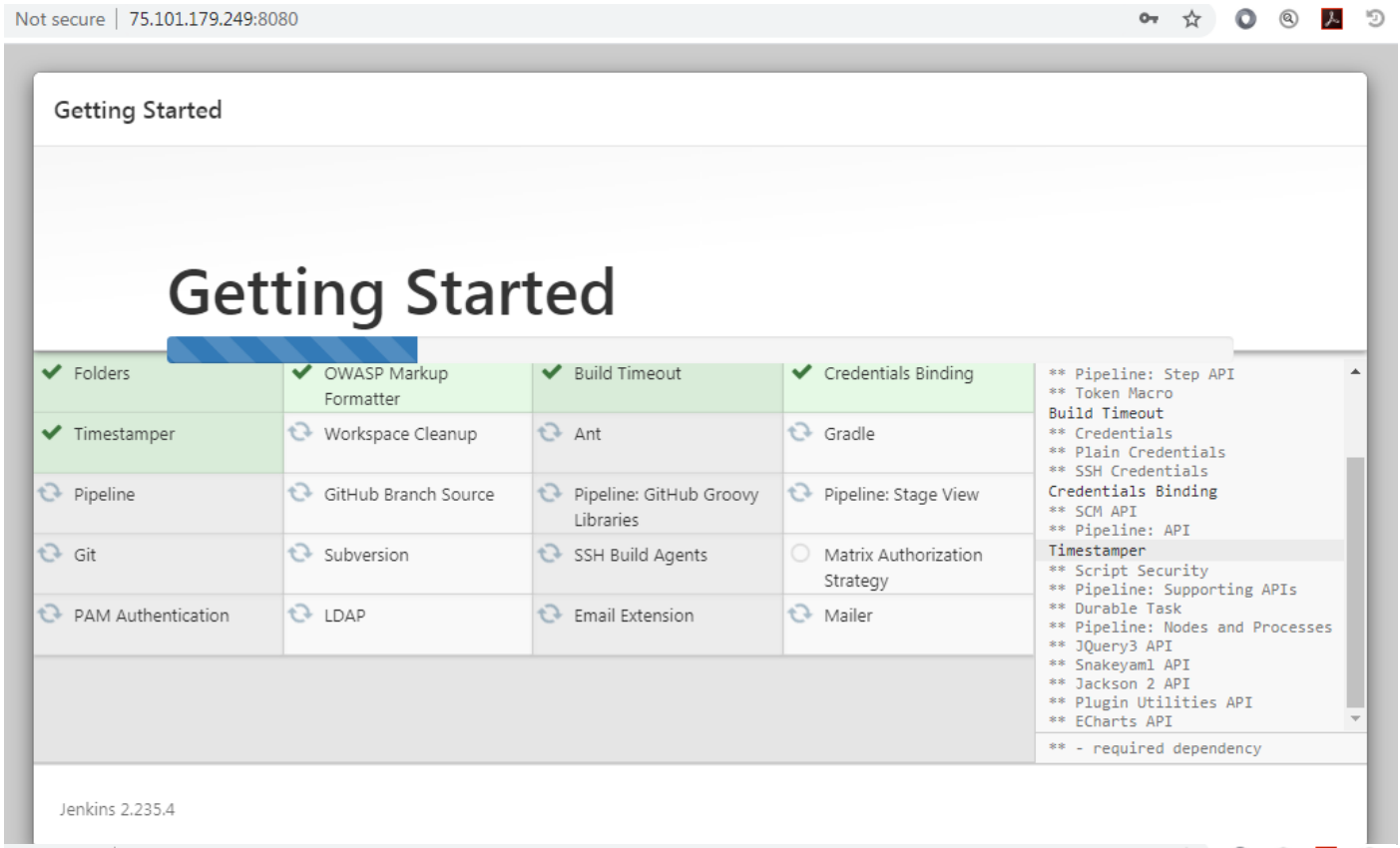
3. Launch Jenkins, and on the home page, choose **Manage Jenkins**.

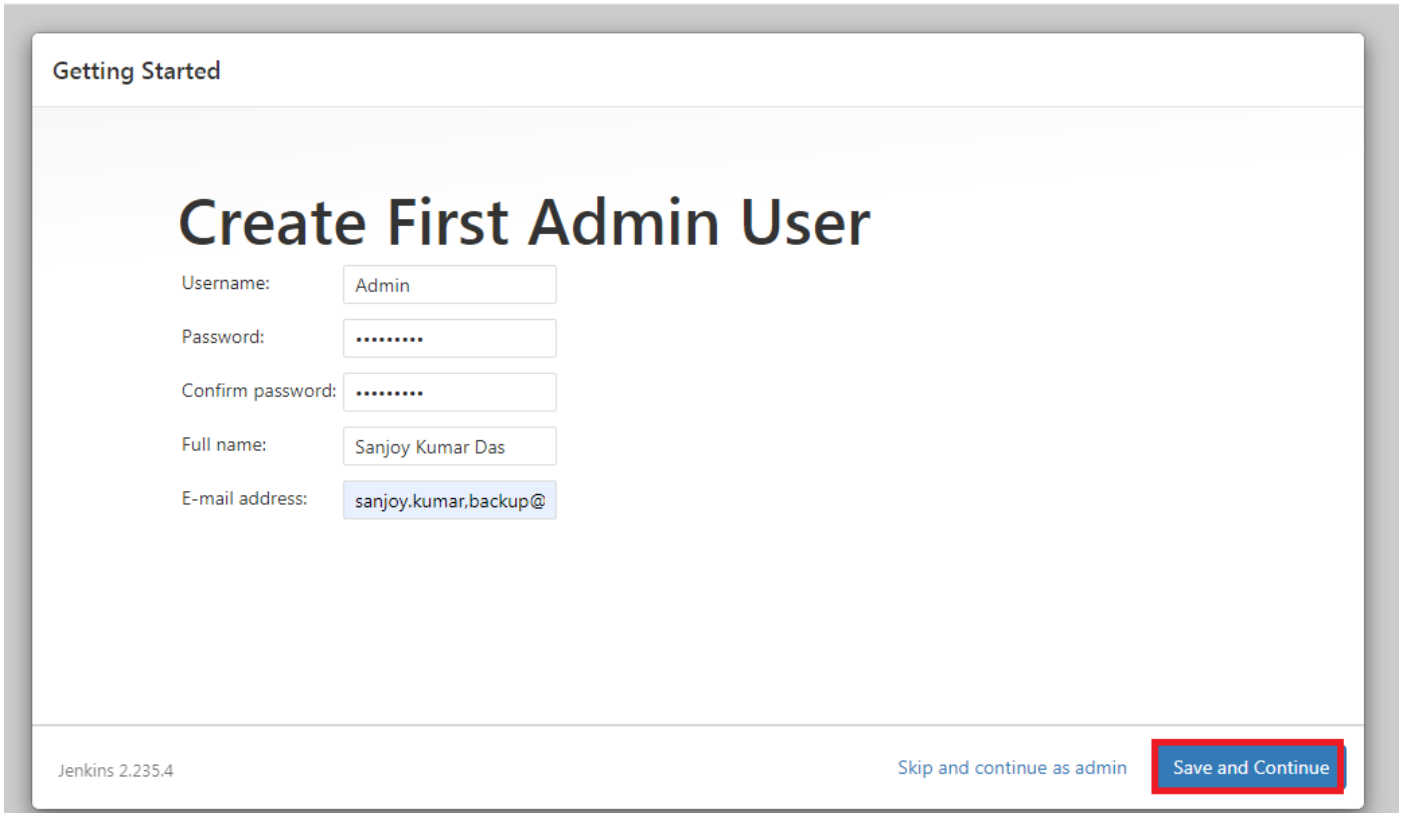


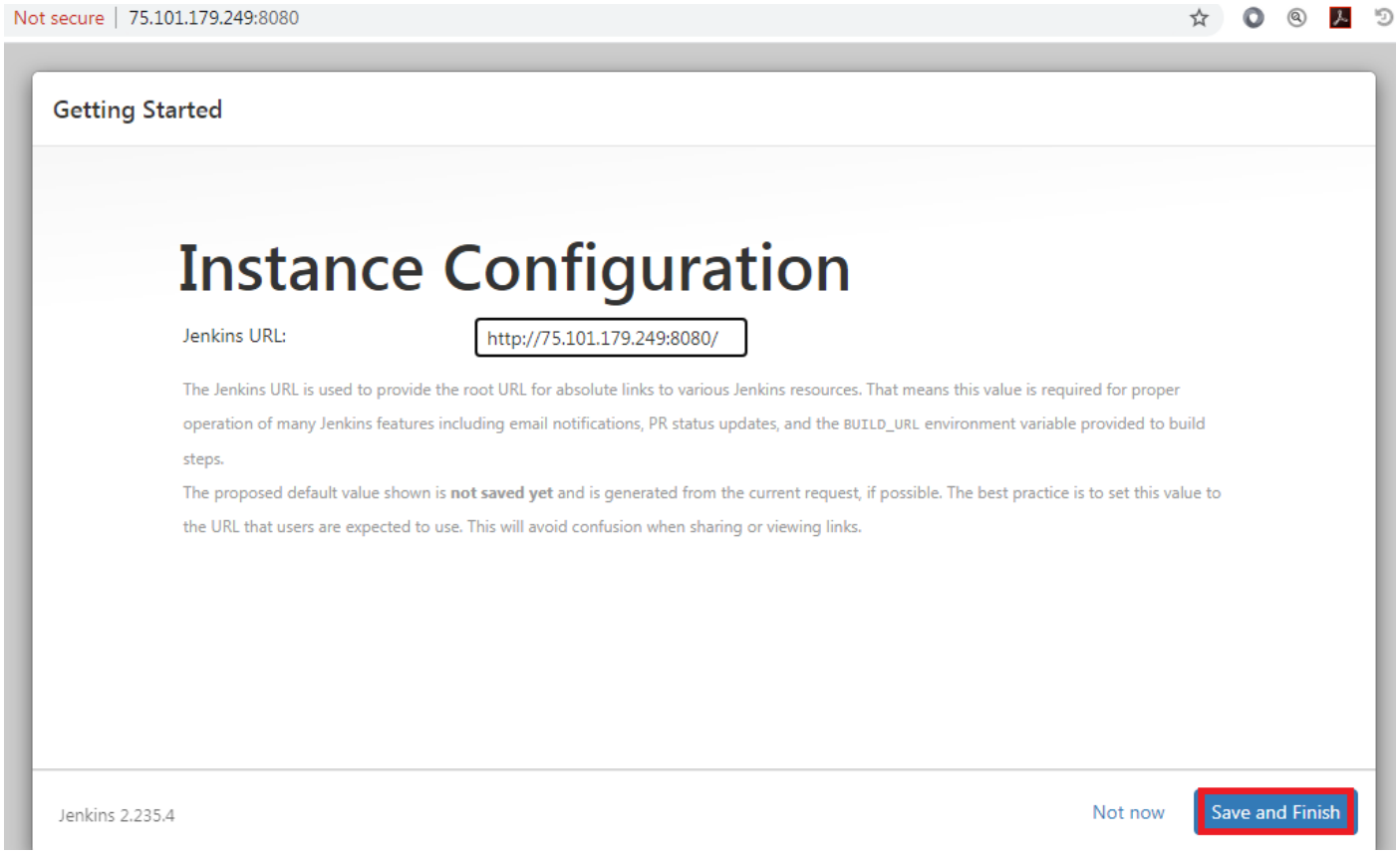
[root@ip-172-31-13-175 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword


Not secure | 75.101.179.249:8080/login?from=%2F
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/var/lib/jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
Continue

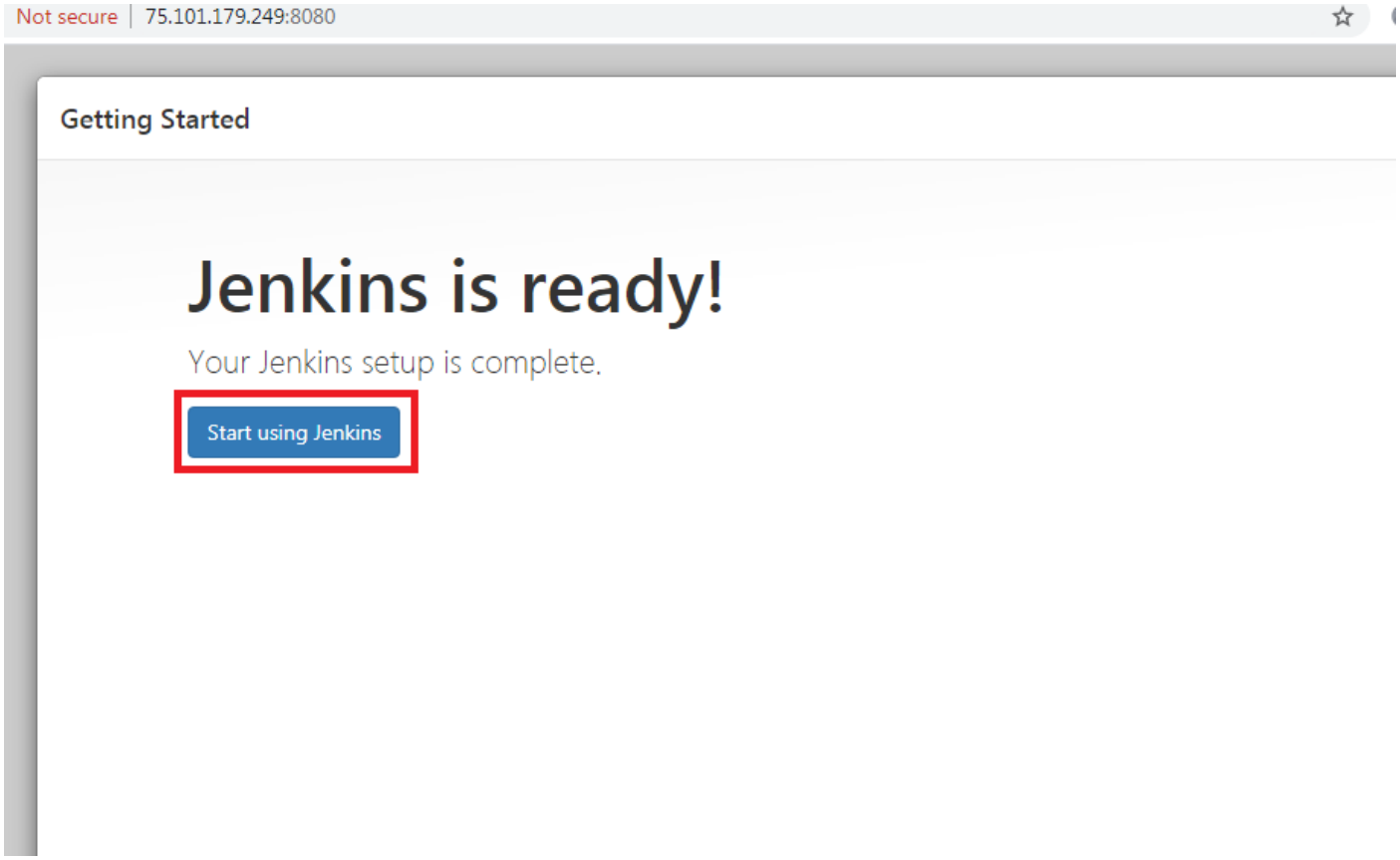

Not secure | 75.101.179.249:8080
Getting Started
Customize Jenkins
Plugins extend Jenkins with additional features to support many different needs.
Install suggested plugins
Install plugins the Jenkins community finds most useful.
Select plugins to install
Select and install plugins most suitable for your needs.
Jenkins 2.235.4

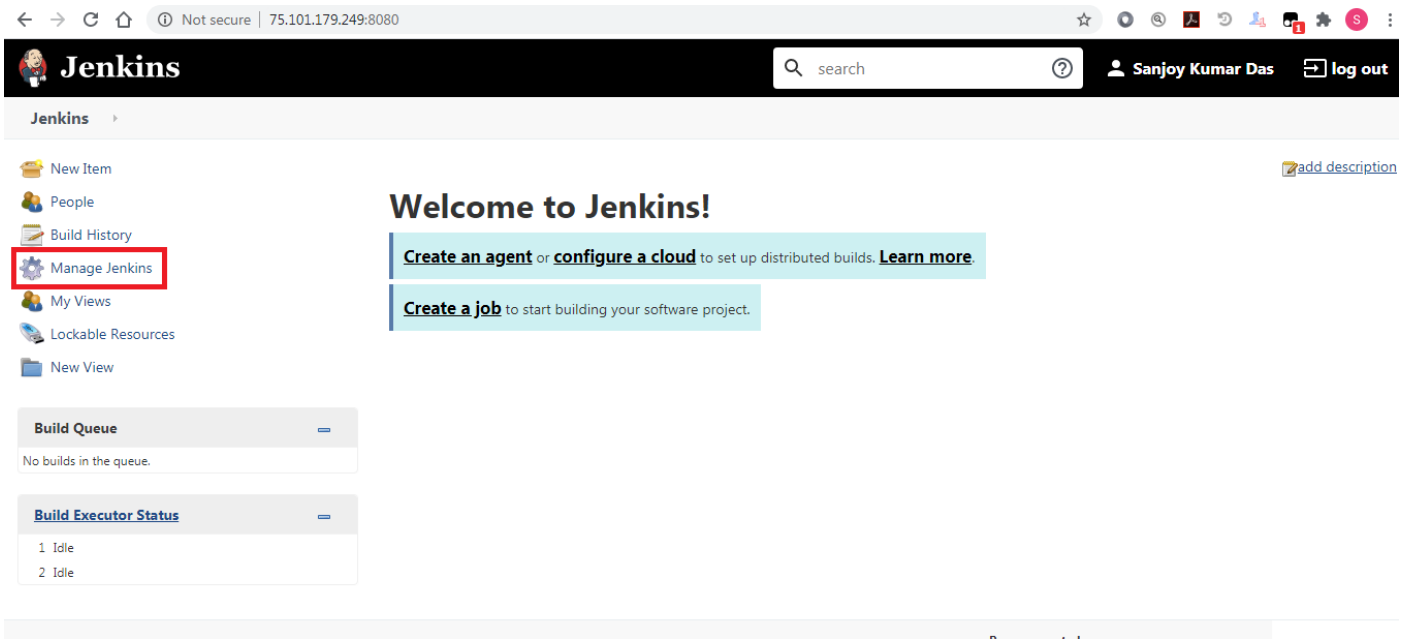

Not secure | 75.101.179.249:8080
Getting Started
Getting Started
Folders
OWASP Markup Formatter
Build Timeout
Credentials Binding
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git
Subversion
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
** Pipeline: Step API
** Token Macro
Build Timeout
** Credentials
** Plain Credentials
** SSH Credentials
Credentials Binding
** SCM API
** Pipeline: API
Timestamper
** Script Security
** Pipeline: Supporting APIs
** Durable Task
** Pipeline: Nodes and Processes
** JQuery3 API
** Snakeyaml API
** Jackson 2 API
** Plugin Utilities API
** ECharts API
** - required dependency
Jenkins 2.235.4


Not secure | 75.101.179.249:8080
Getting Started
Create First Admin User
Username:
Admin
Password:
Confirm password:
Full name:
Sanjoy Kumar Das
E-mail address:
sanjoy.kumar.backup@
Jenkins 2.235.4
Skip and continue as admin
Save and Continue

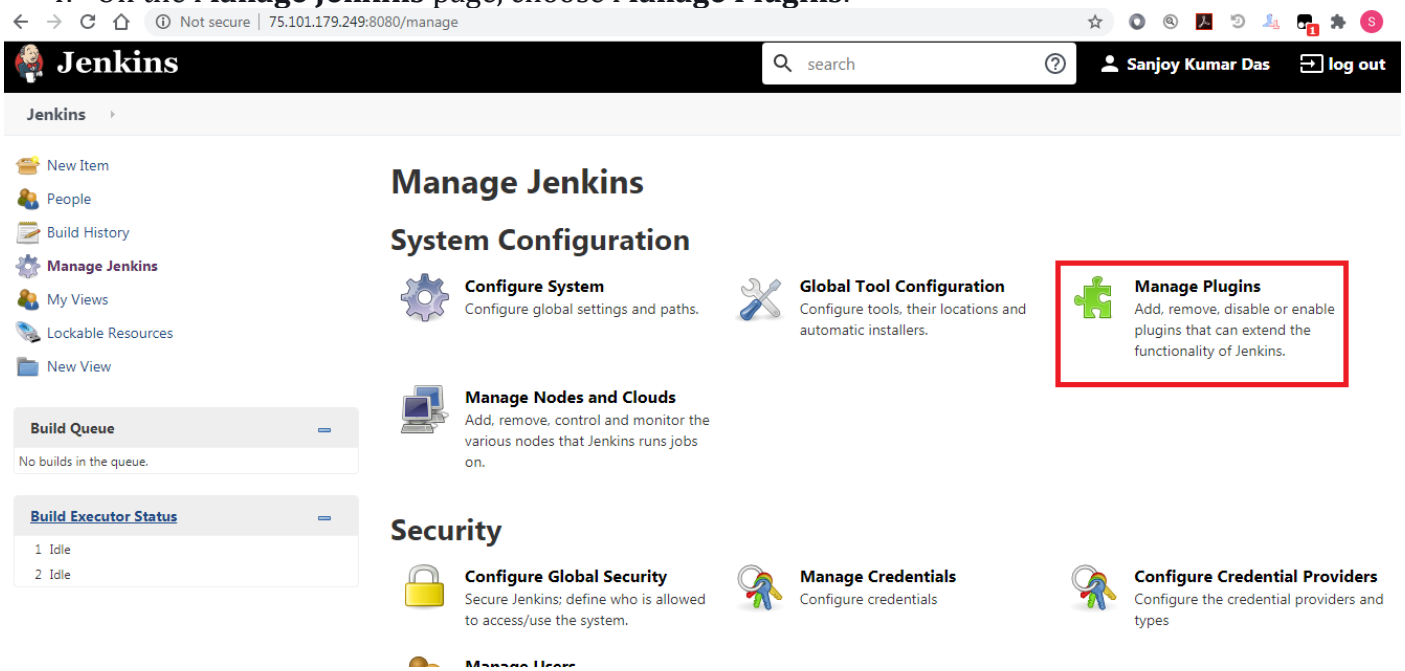


http://75.101.179.249:8080/

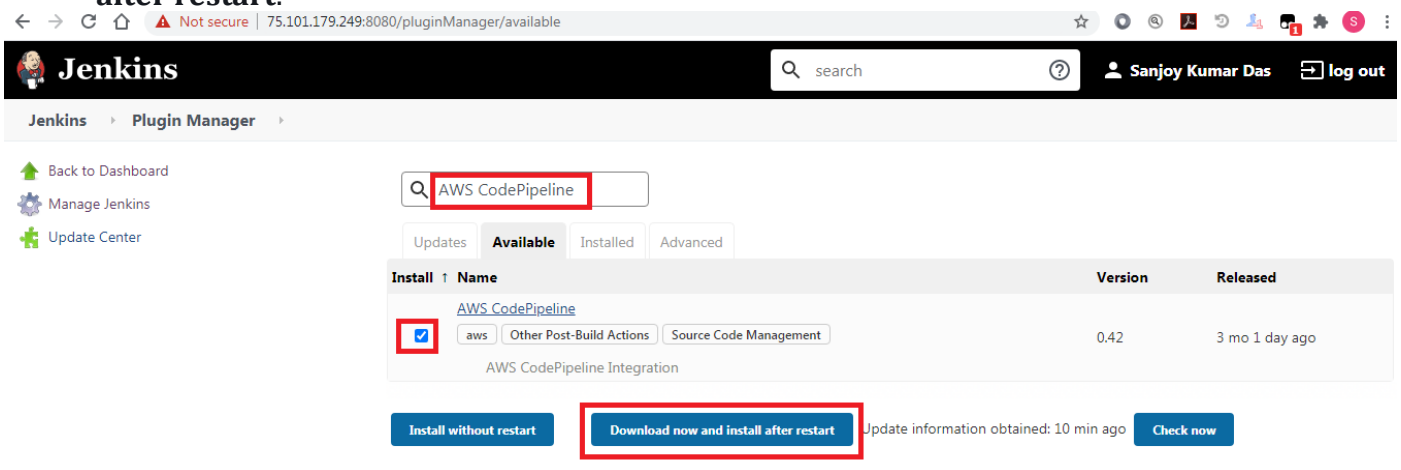




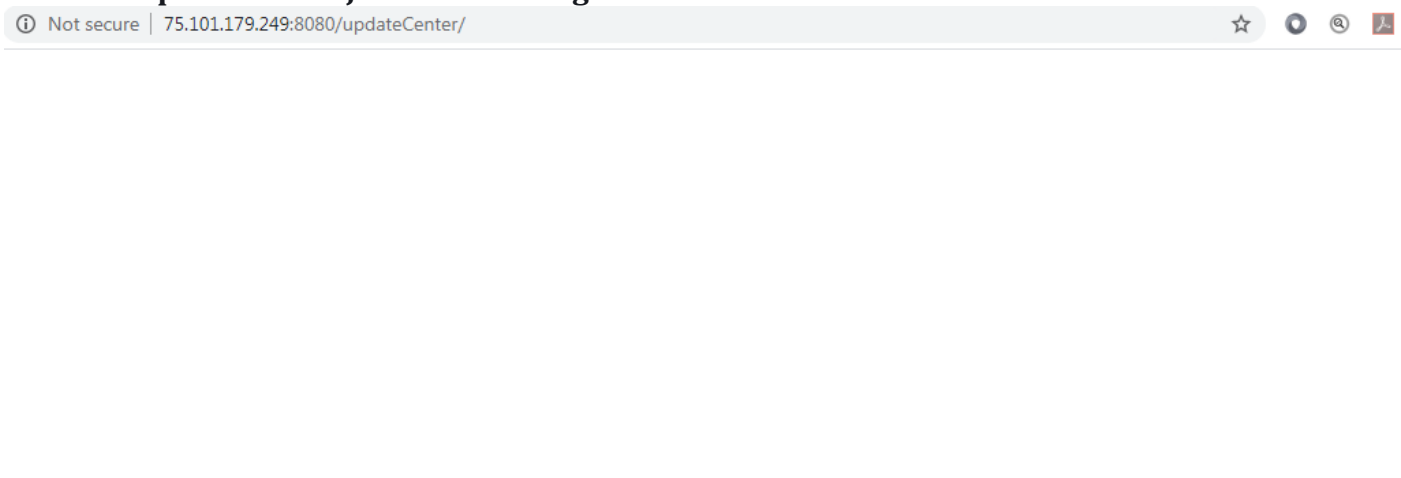
4. On the **Manage Jenkins** page, choose **Manage Plugins**.



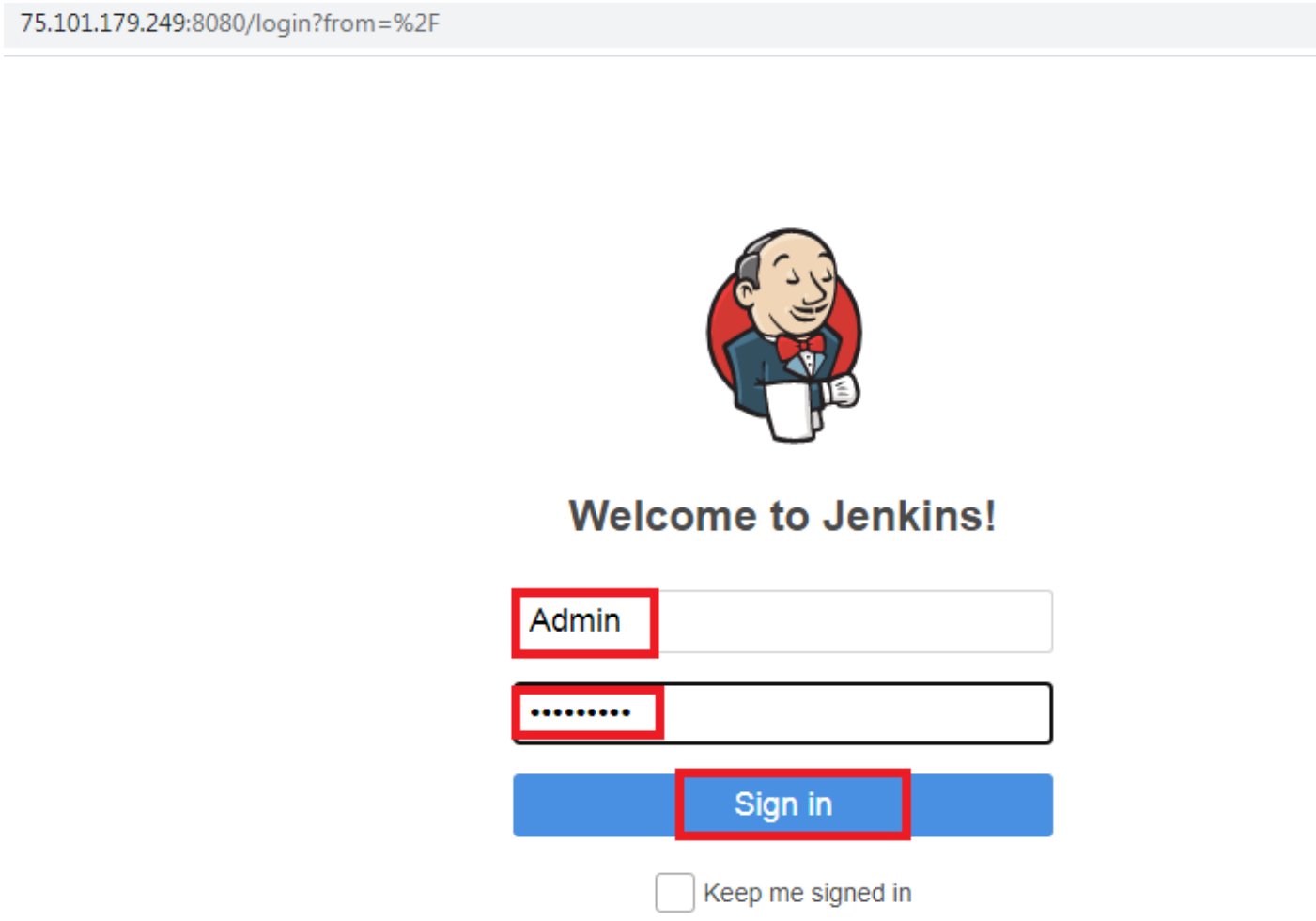
5. Choose the **Available** tab, and in the **Filter** search box, enter `AWS CodePipeline`.
   Choose **CodePipeline Plugin for Jenkins** from the list and choose **Download now and install after restart**.



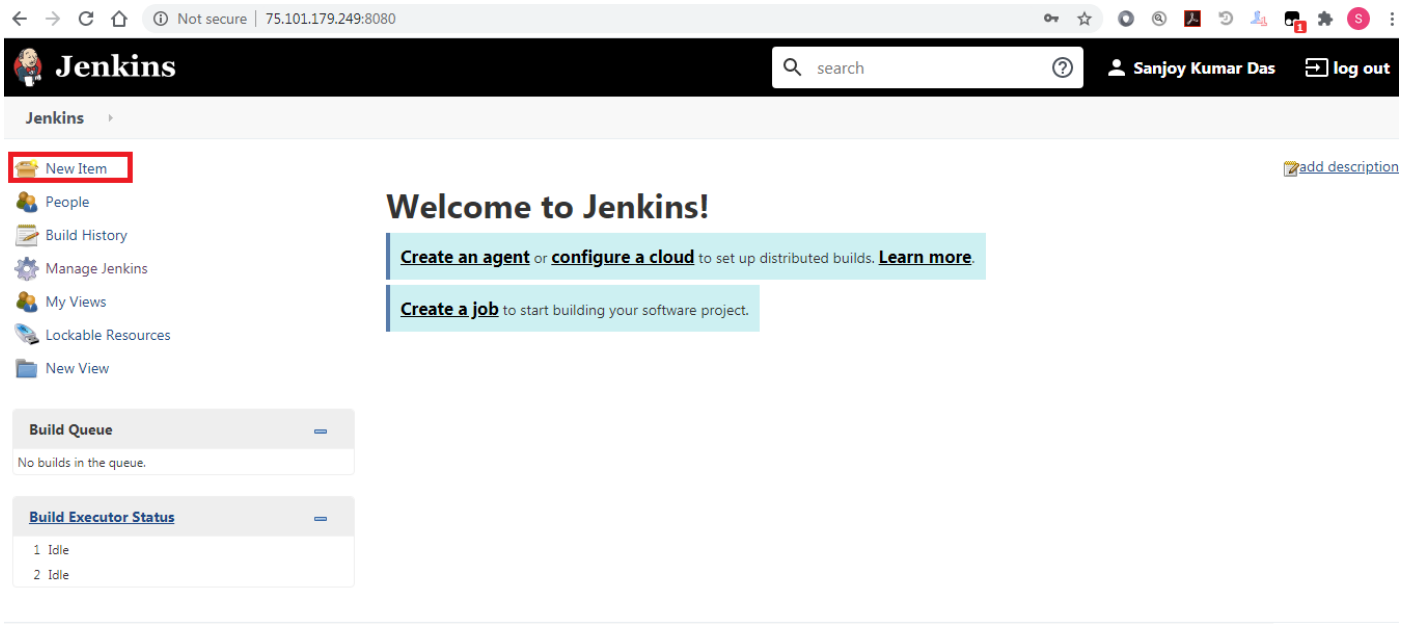
6. On the **Installing Plugins/Upgrades** page, select **Restart Jenkins when installation is complete and no jobs are running**.



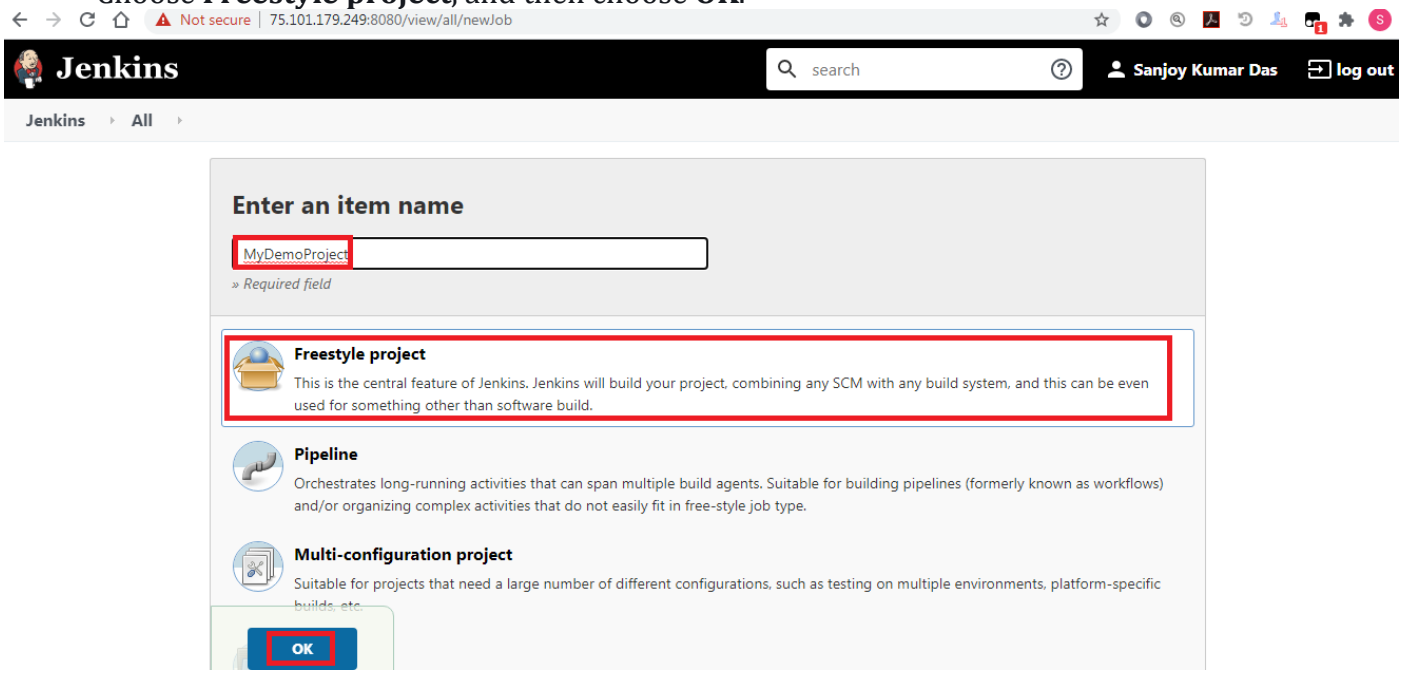
7. Choose **Back to Dashboard**.



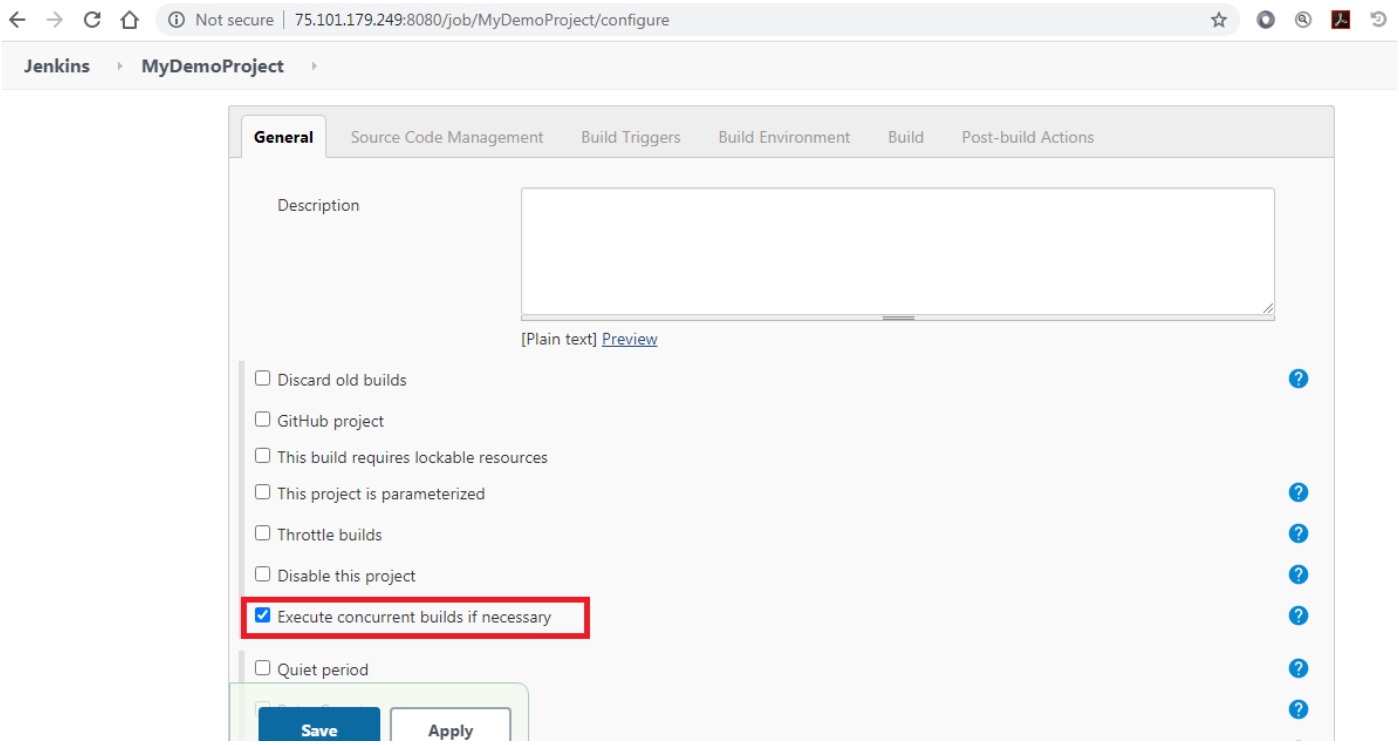
8. On the main page, choose **New Item**.

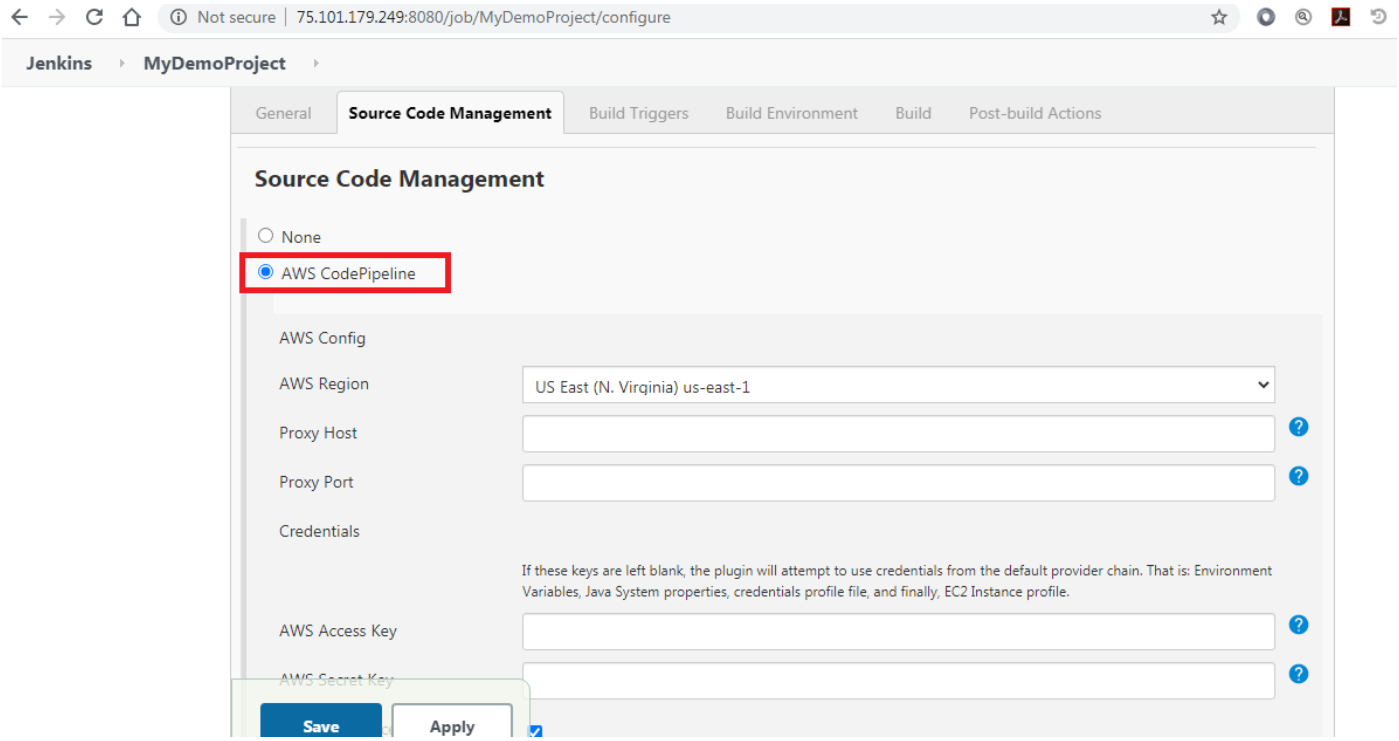


9. In **Item Name**, enter a name for the Jenkins project (for example, `MyDemoProject`). Choose **Freestyle project**, and then choose **OK**.

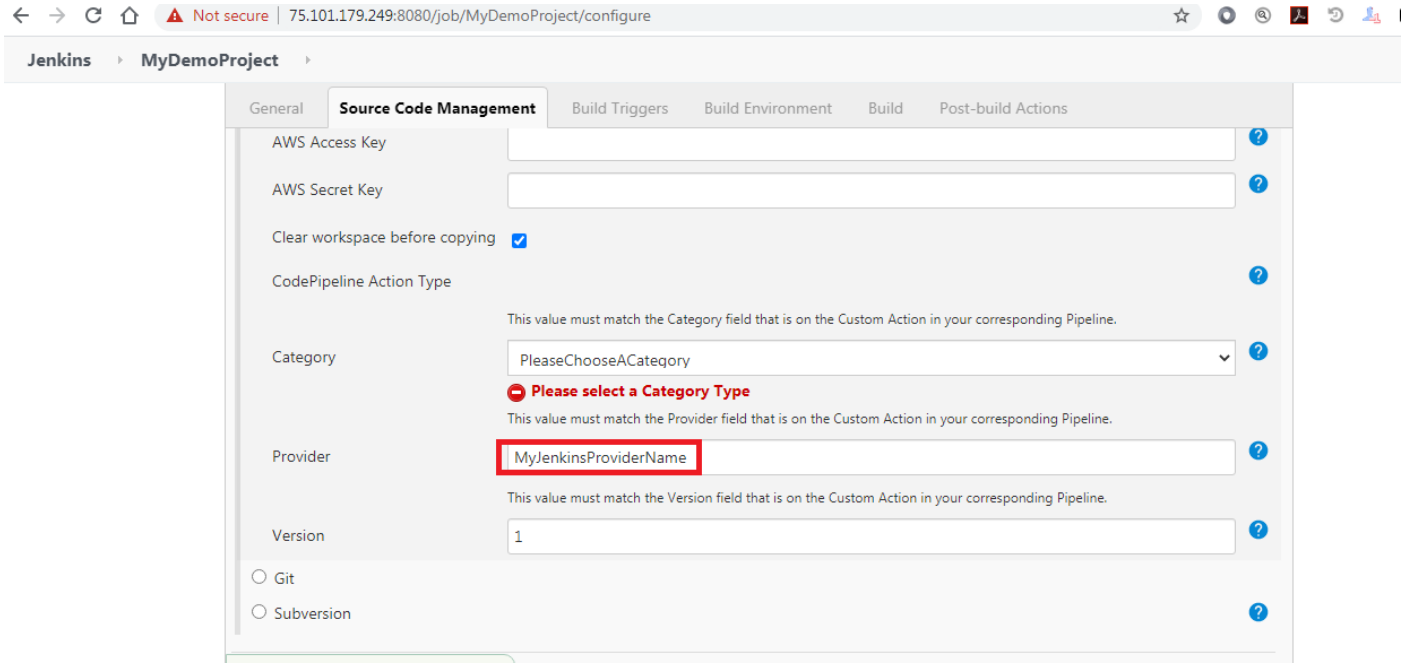


10. On the configuration page for the project, select the **Execute concurrent builds if necessary** check box. In **Source Code Management**, choose **AWS CodePipeline**. If you have installed Jenkins on an EC2 instance and configured the AWS CLI with the profile for the IAM user you created for integration between CodePipeline and Jenkins, leave all of the other fields empty.





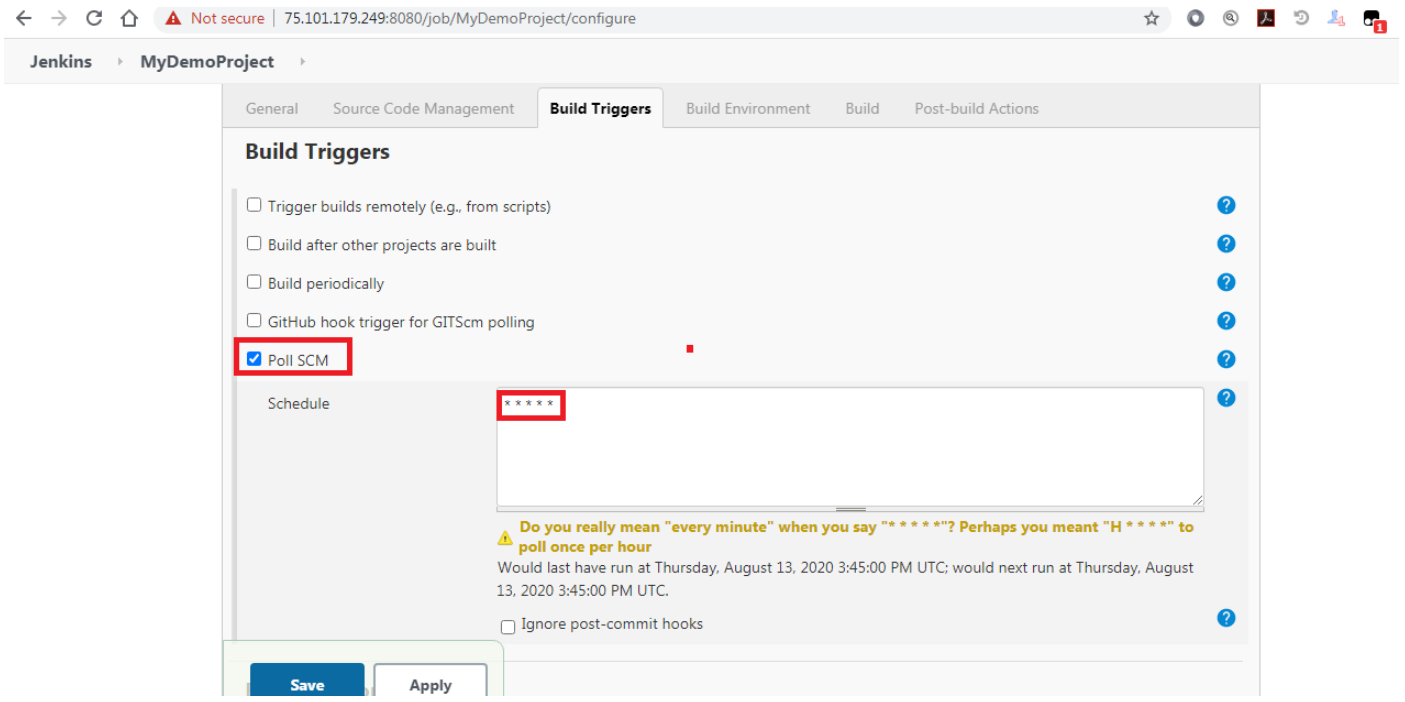
11. Choose **Advanced**, and in **Provider**, enter a name for the provider of the action as it will appear in CodePipeline (for example, `MyJenkinsProviderName`). Make sure that this name is unique and easy to remember. You will use it when you add a build action to your pipeline later in this tutorial, and again when you add a test action.



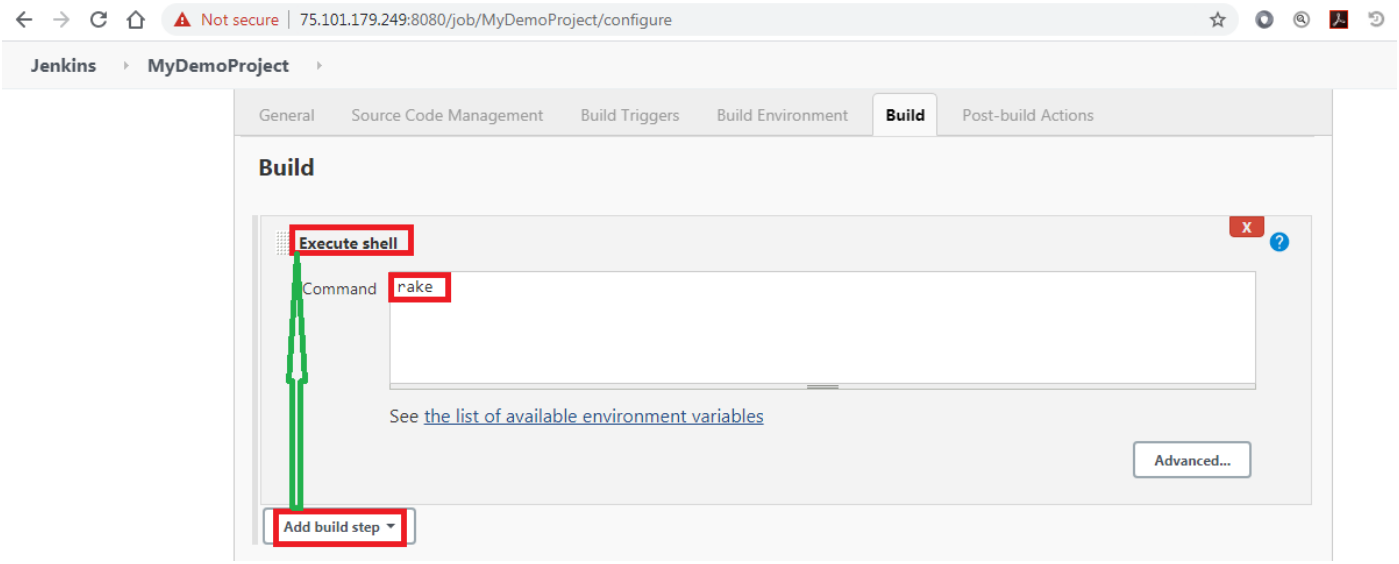
12. In **Build Triggers**, clear any check boxes, and then select **Poll SCM**. In **Schedule**, enter five asterisks separated by spaces, as follows:

```
* * * * *
```

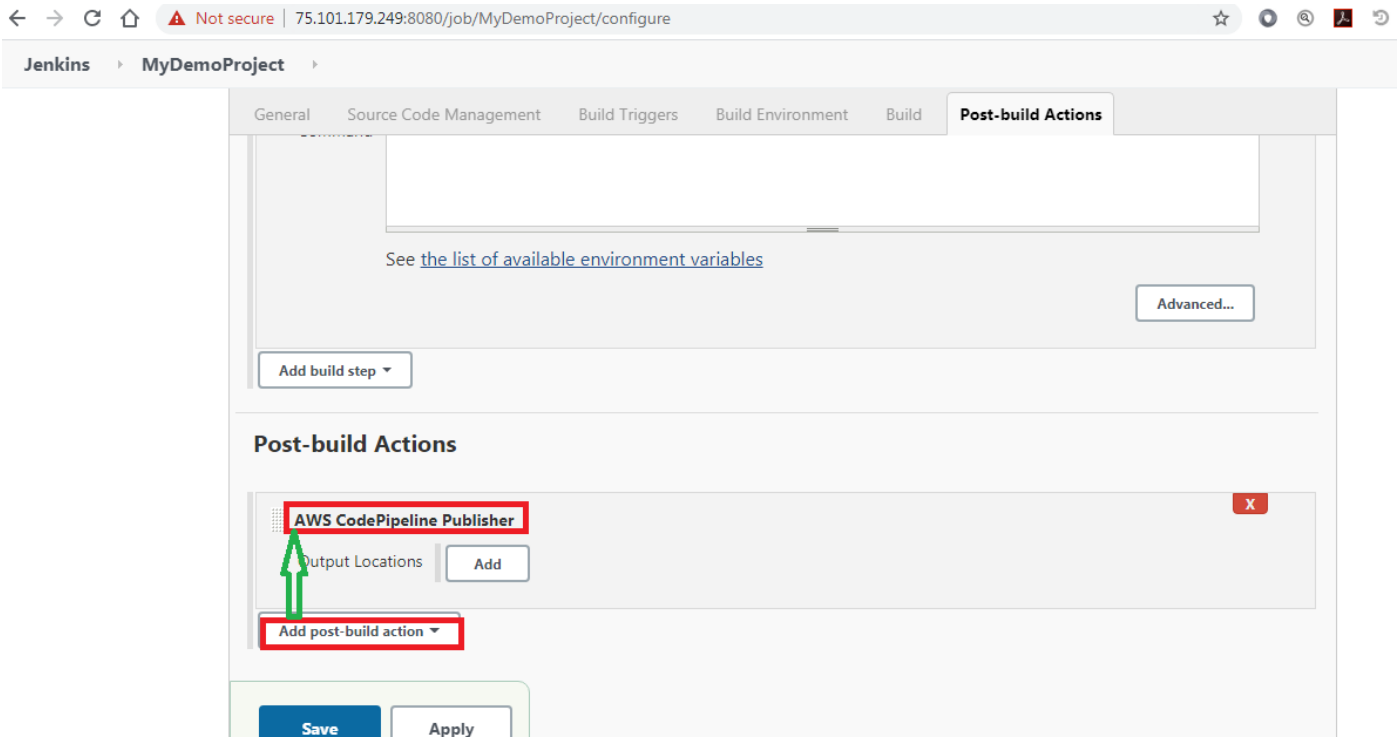
This polls CodePipeline every minute.



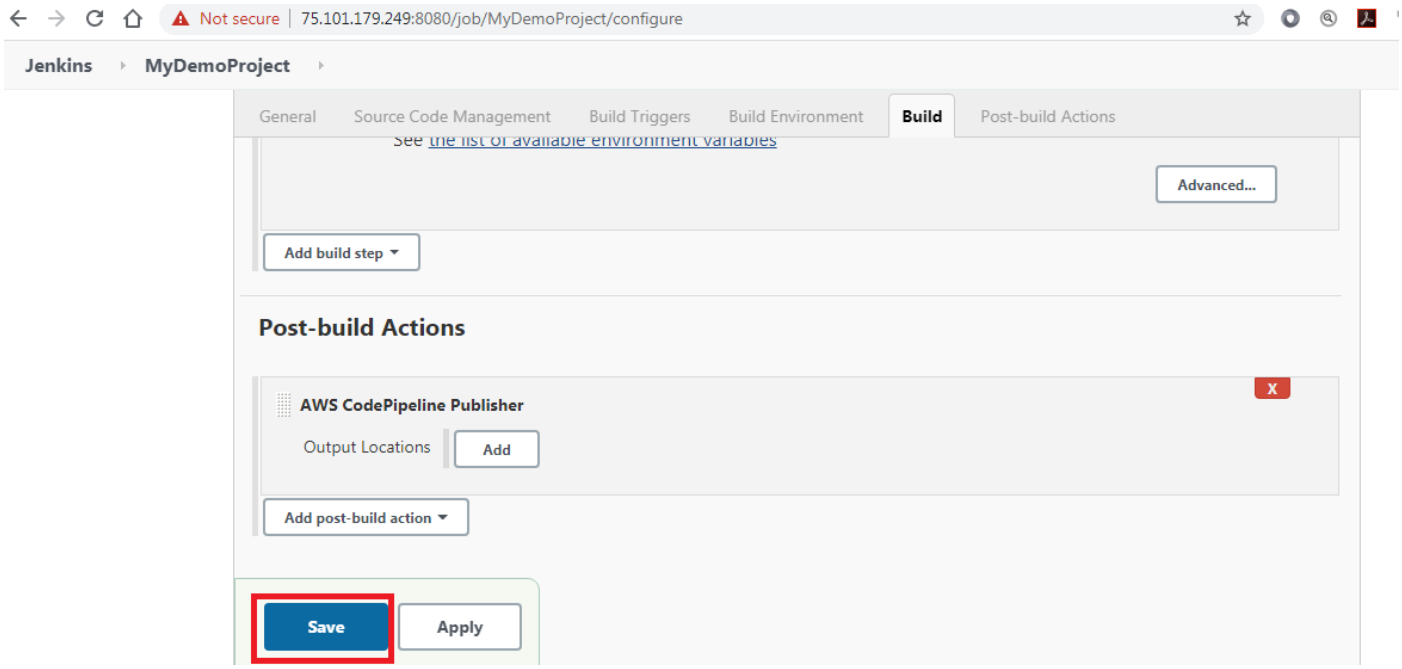
13. In **Build**, choose **Add build step**. Choose **Execute shell** (Amazon Linux, RHEL, or Ubuntu Server) **Execute batch command** (Windows Server), and then enter the following:



14. Choose **Add post-build action**, and then choose **AWS CodePipeline Publisher**. Choose **Add**, and in **Build Output Locations**, leave the location blank. This configuration is the default. It will create a compressed file at the end of the build process.



15. Choose **Save** to save your Jenkins project.



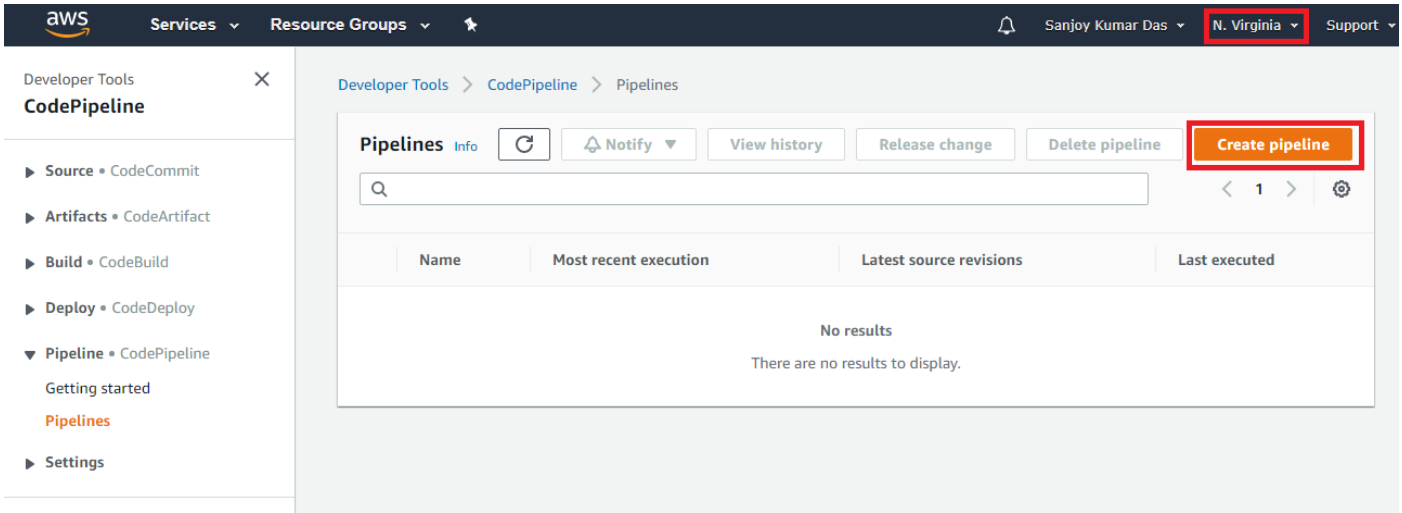
Step 2: Create a pipeline in CodePipeline

To create a CodePipeline automated release process

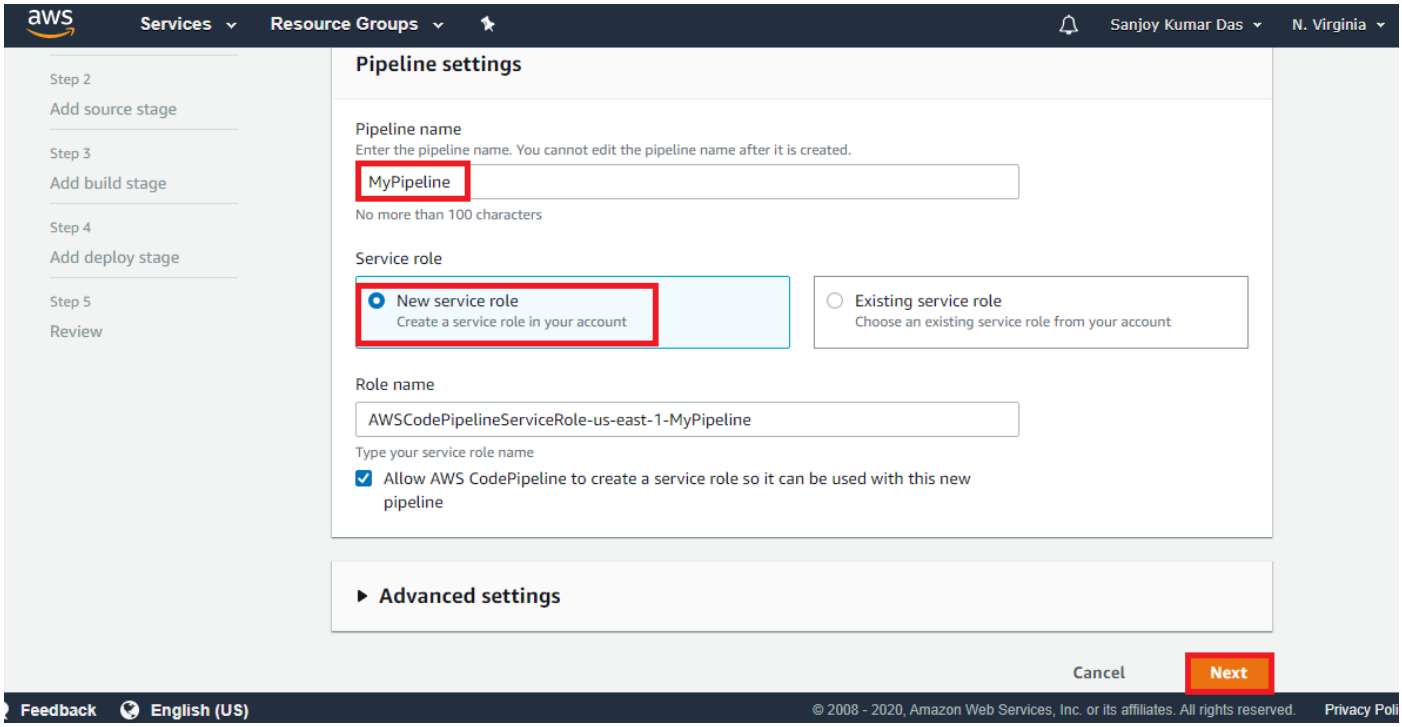
1. Sign in to the AWS Management Console and open the CodePipeline console at http://console.aws.amazon.com/codesuite/codepipeline/home.

2. If necessary, use the Region selector to change the Region to the one where your pipeline resources are located. For example, if you created resources for the previous tutorial in `us-east-1`, make sure that the Region selector is set to N. Virginia.

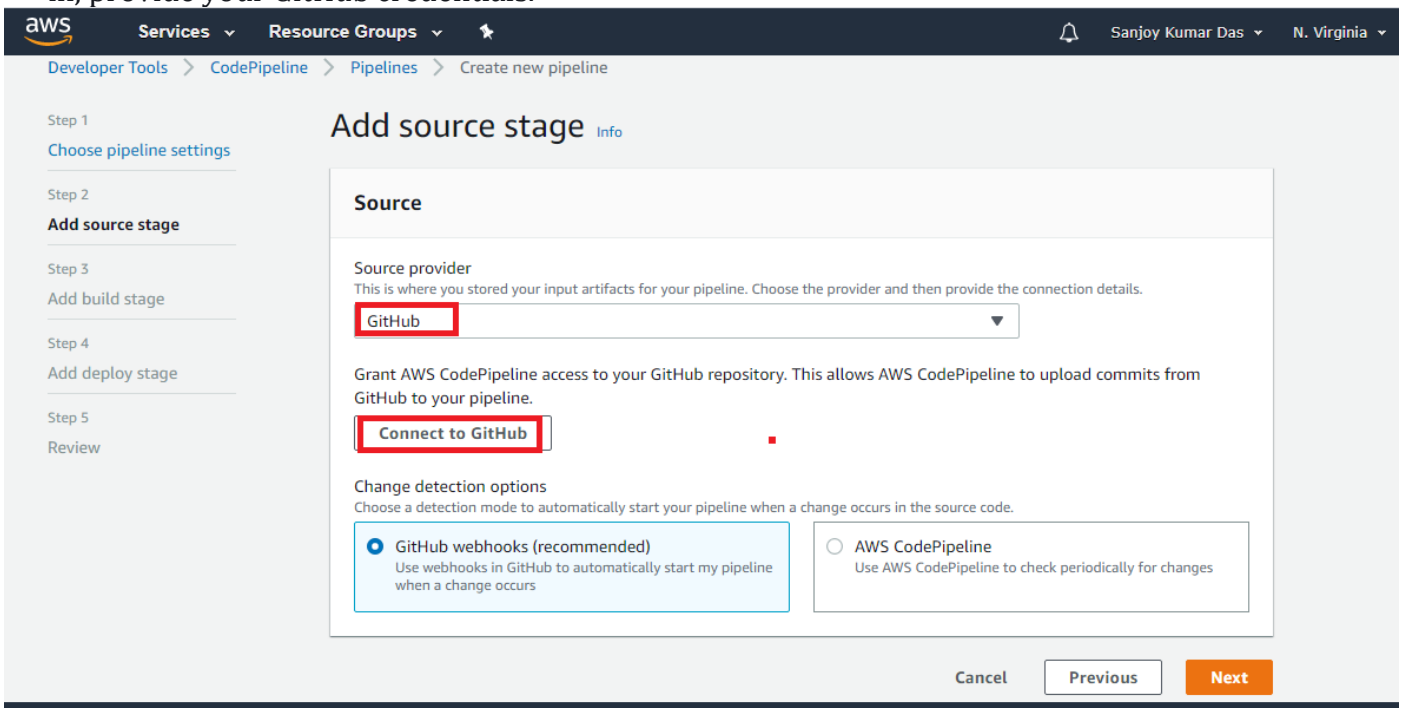
3. On the **Welcome** page, **Getting started** page, or the **Pipelines** page, choose **Create pipeline**.

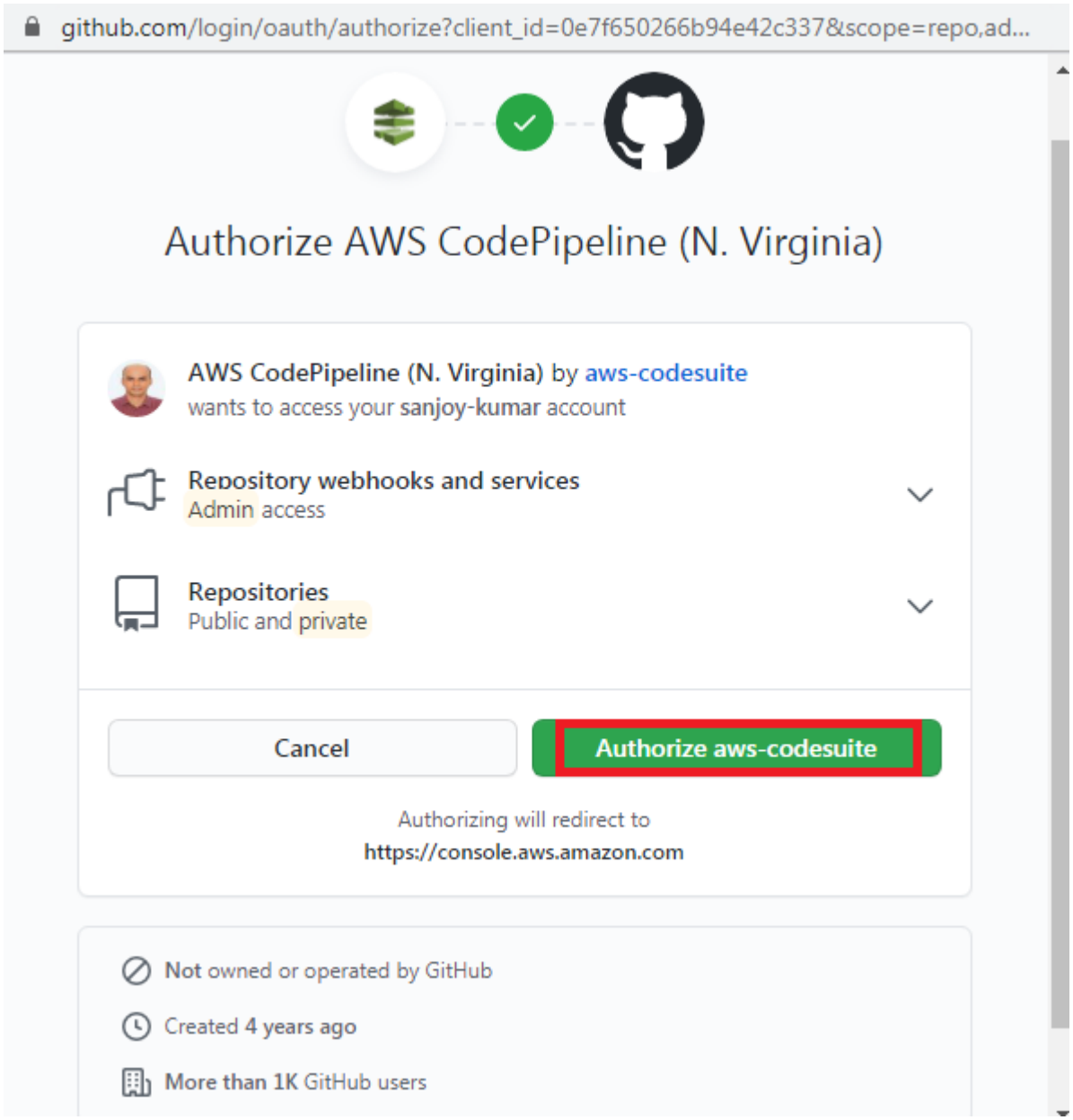


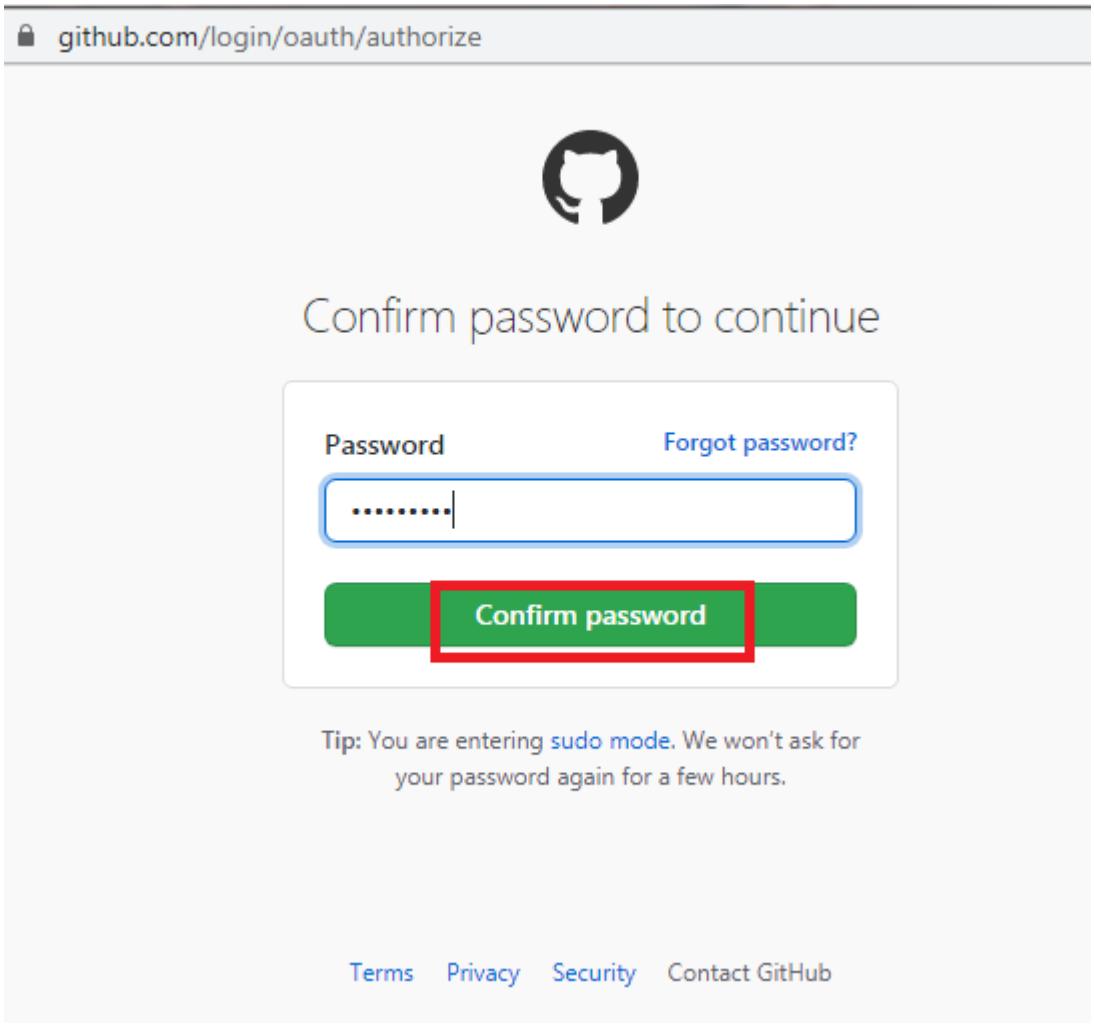
4. On the **Step 1: Choose pipeline settings** page, in **Pipeline name**, enter the name for your pipeline.
5. In **Service role**, choose **New service role** to allow CodePipeline to create a service role in IAM.
6. Leave the settings under **Advanced settings** at their defaults, and choose **Next**.

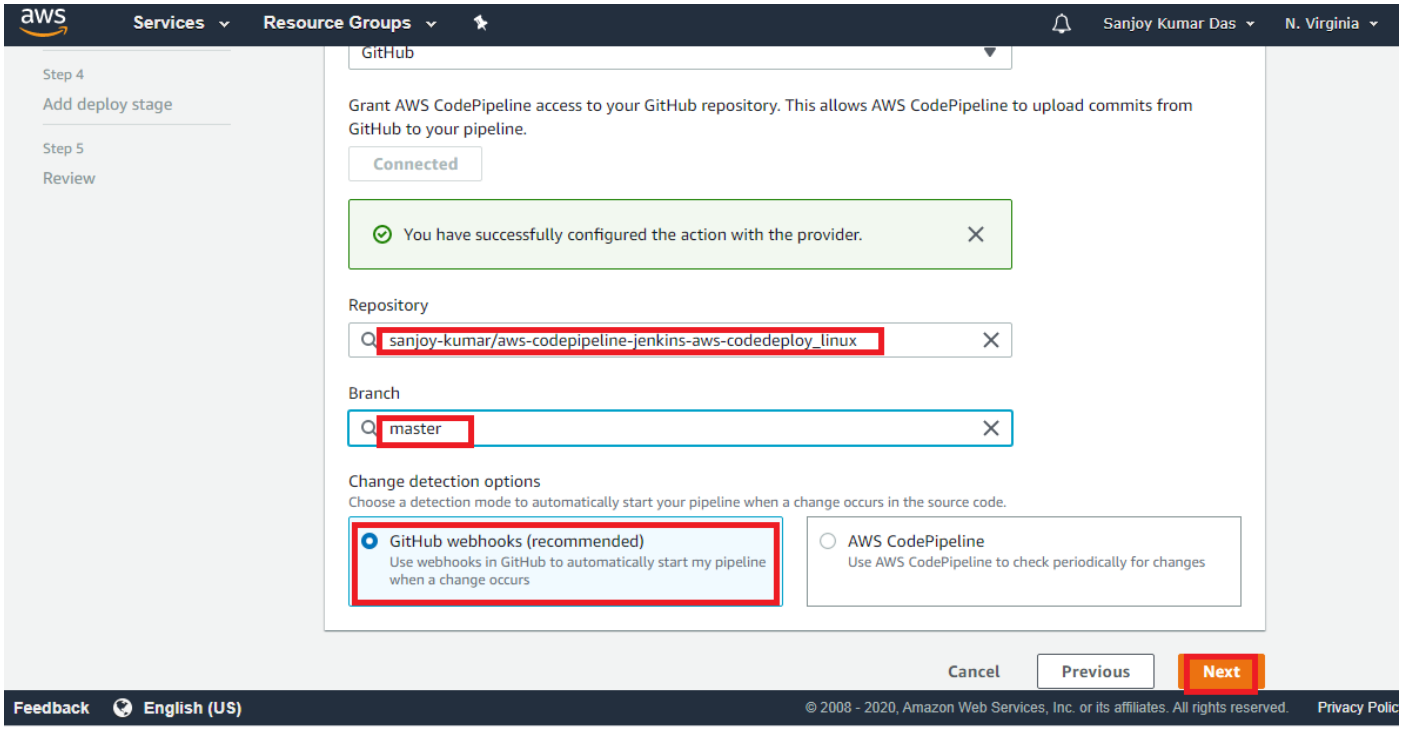


7. In **Step 2: Add source stage**, in **Source provider**, choose **GitHub**, and then choose **Connect to GitHub**. This will open a new browser window that will connect you to GitHub. If prompted to sign in, provide your GitHub credentials.



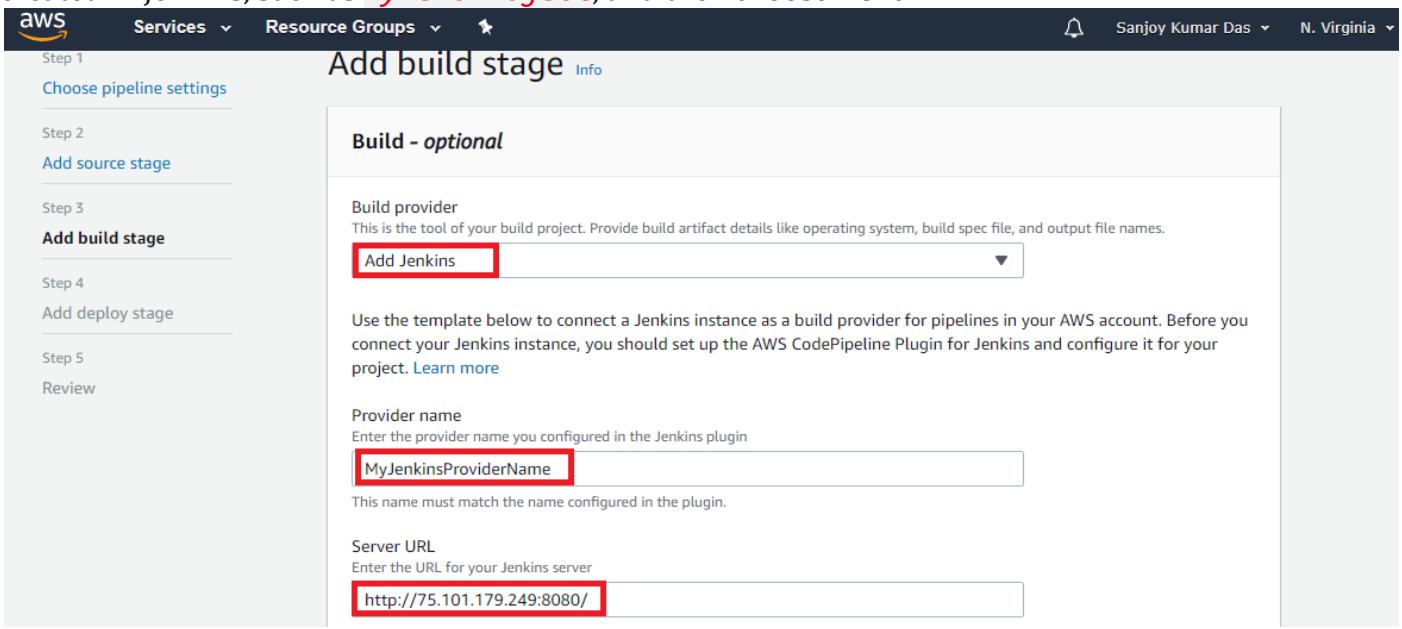

github.com/login/oauth/authorize?client_id=0e7f650266b94e42c337&scope=repo,ad...
Authorize AWS CodePipeline (N. Virginia)
AWS CodePipeline (N. Virginia) by aws-codesuite
wants to access your sanjoy-kumar account
Repository webhooks and services
Admin access
Repositories
Public and private
Cancel
Authorize aws-codesuite
Authorizing will redirect to
https://console.aws.amazon.com
Not owned or operated by GitHub
Created 4 years ago
More than 1K GitHub users

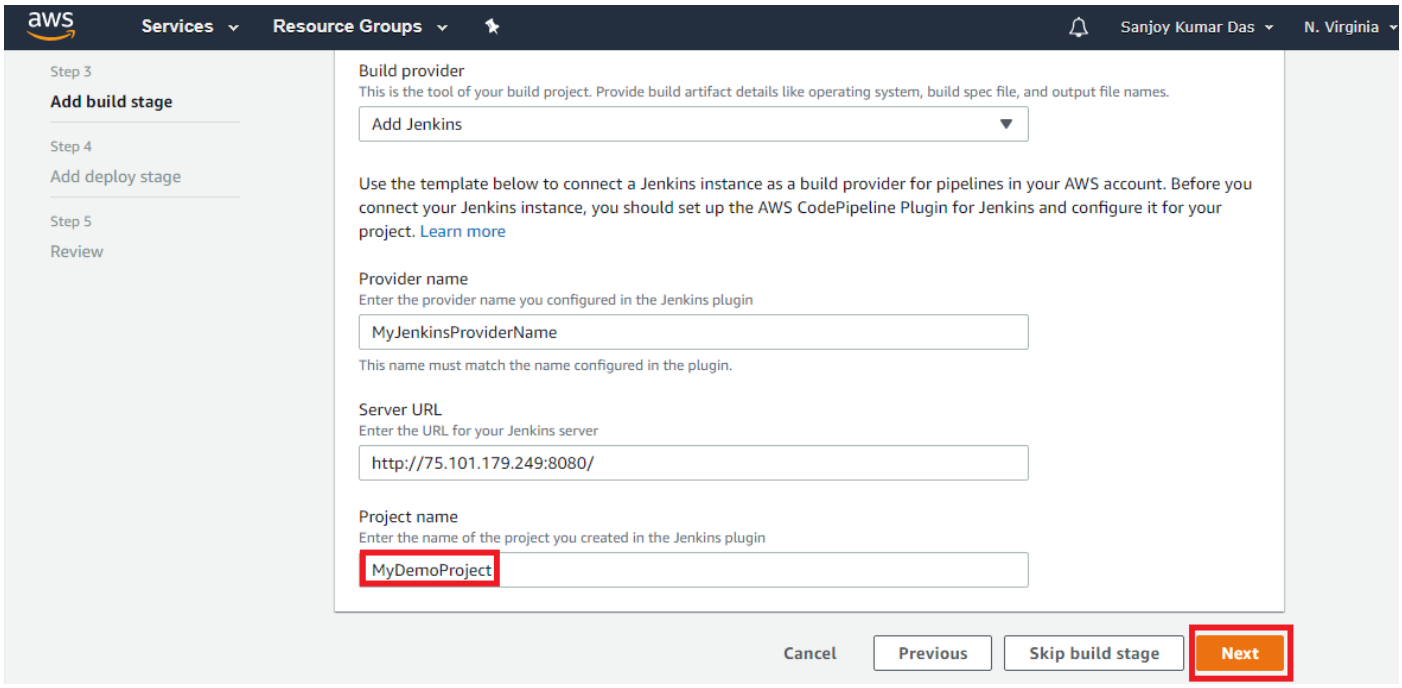




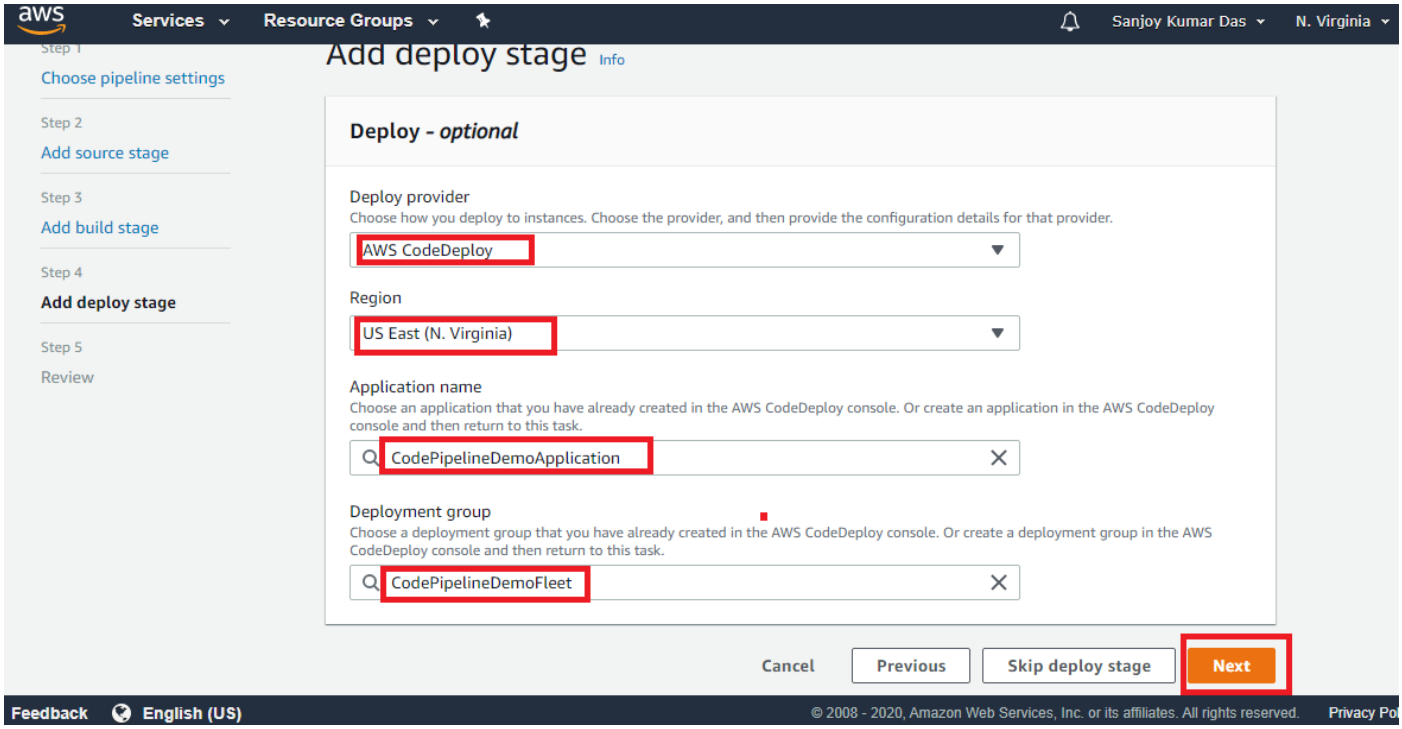
8. In **Step 3: Add build stage**, choose **Add Jenkins**. In **Provider name**, enter the name of the action you provided in the CodePipeline Plugin for Jenkins (for example `MyJenkinsProviderName`). This name must exactly match the name in the CodePipeline Plugin for Jenkins. In **Server URL**, enter the URL of

the EC2 instance where Jenkins is installed. In **Project name**, enter the name of the project you created in Jenkins, such as `MyDemoProject`, and then choose **Next**.





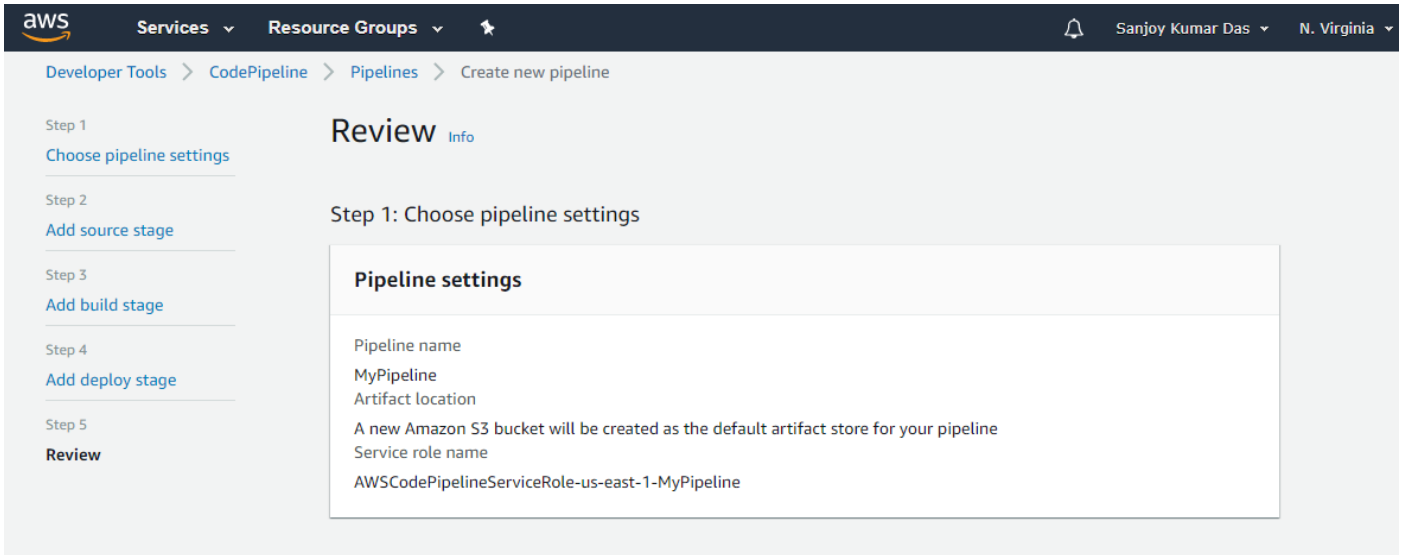
9. In **Step 4: Add deploy stage**, reuse the CodeDeploy application and deployment group you created in Tutorial: Create a simple pipeline (S3 bucket). In **Deploy provider**, choose **CodeDeploy**. In **Application name**, enter `CodePipelineDemoApplication`, or choose the refresh button, and then choose the application name from the list. In **Deployment group**, enter `CodePipelineDemoFleet`, or choose it from the list, and then choose **Next**.

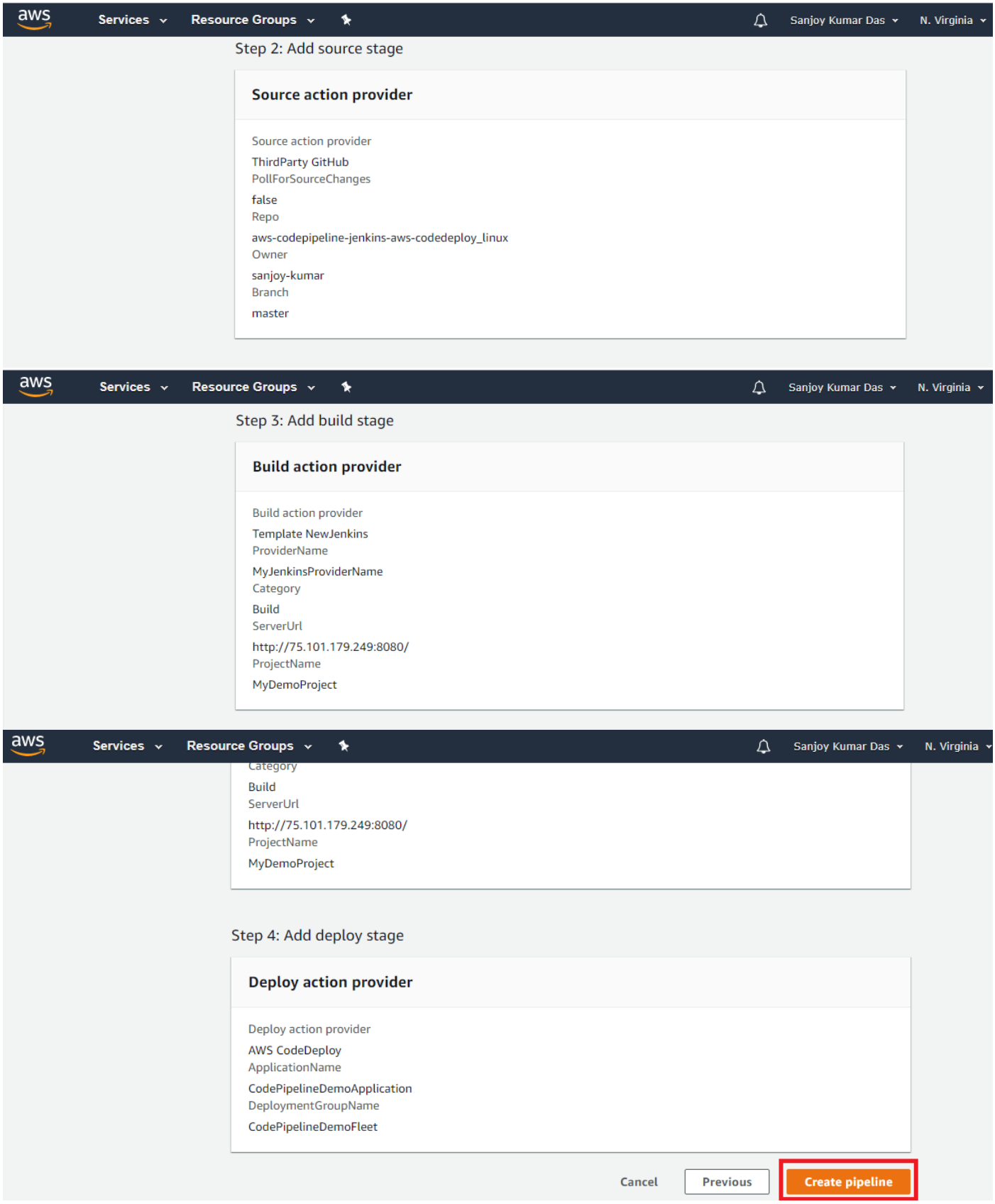


Note

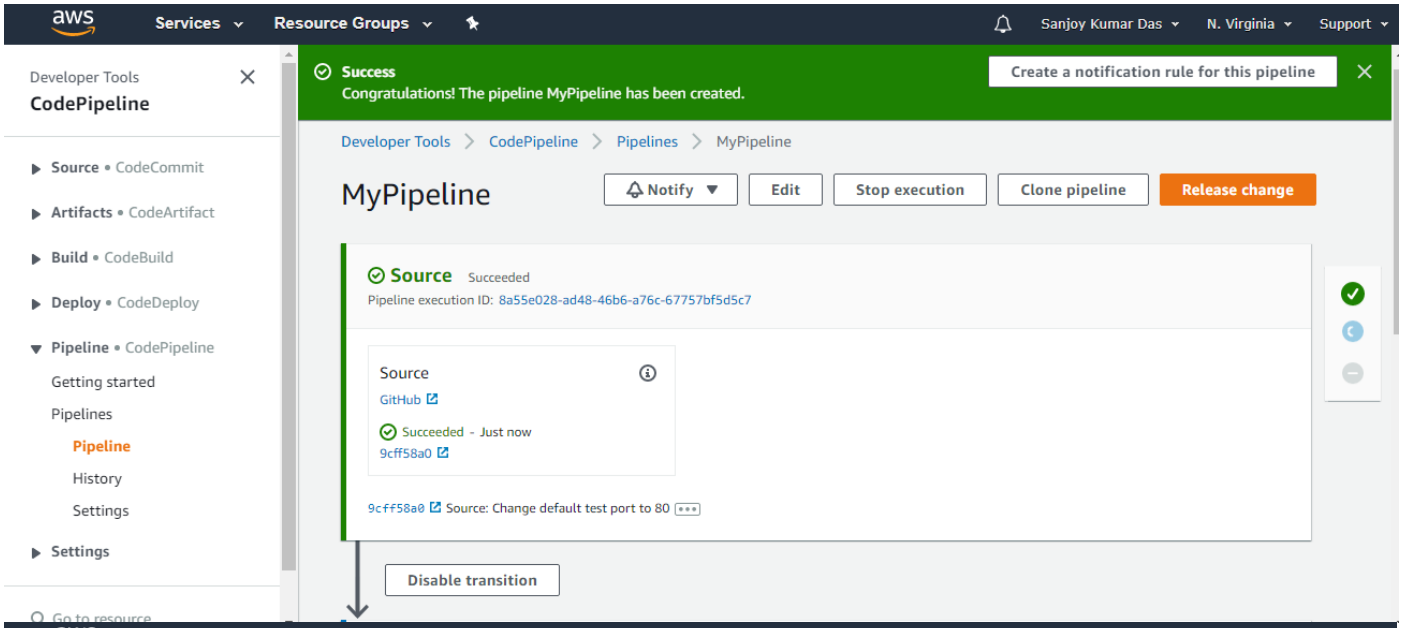
You can use your own CodeDeploy resources or create new ones, but you might incur additional costs.

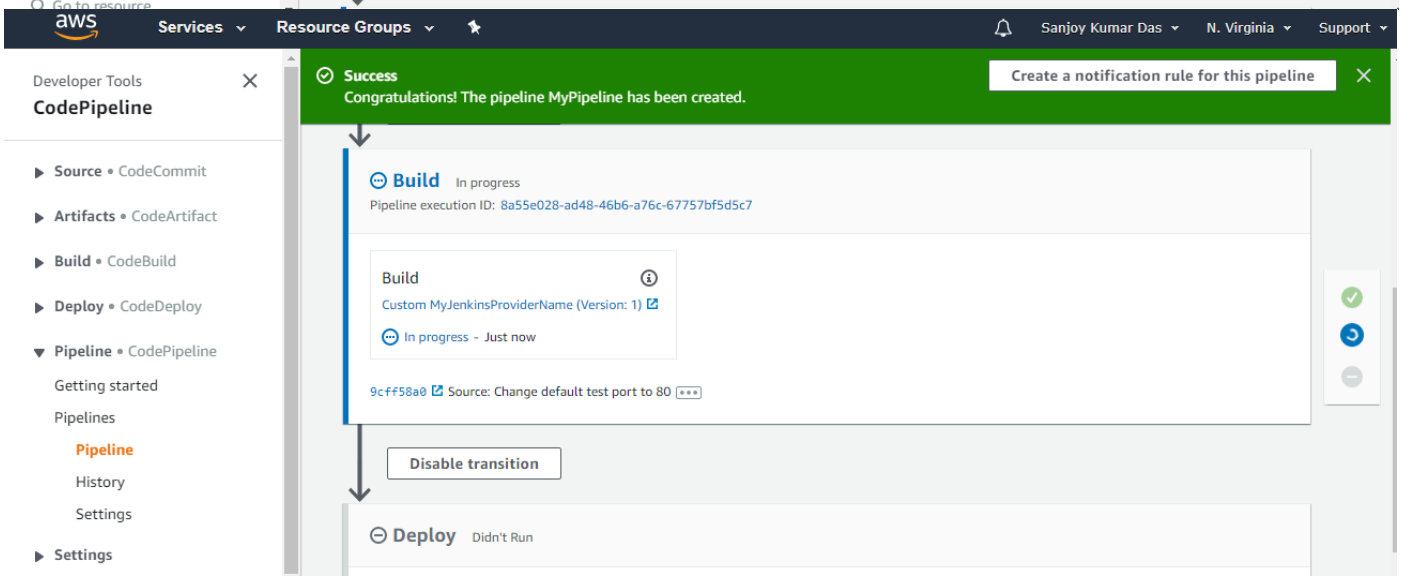
10. In **Step 5: Review**, review the information, and then choose **Create pipeline**.

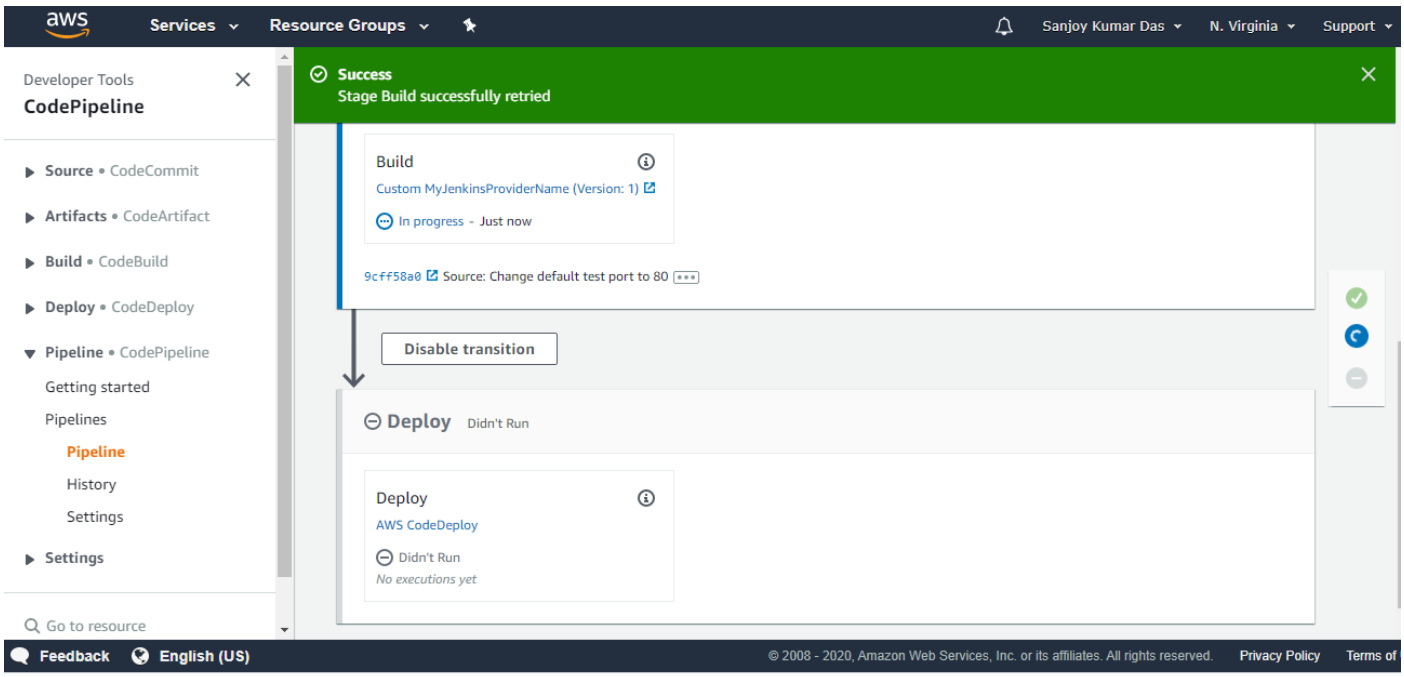




11. The pipeline automatically starts and runs the sample through the pipeline. You can view progress and success and failure messages as the pipeline builds the Haml sample to HTML and deploys it a webpage to each of the Amazon EC2 instances in the CodeDeploy deployment.


aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Success
Congratulations! The pipeline MyPipeline has been created.
Create a notification rule for this pipeline
Developer Tools > CodePipeline > Pipelines > MyPipeline
MyPipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Source Succeeded
Pipeline execution ID: 8a55e028-ad48-46b6-a76c-67757bf5d5c7
Source
GitHub
Succeeded - Just now
9cff58a0
9cff58a0 Source: Change default test port to 80
Disable transition


aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Success
Congratulations! The pipeline MyPipeline has been created.
Create a notification rule for this pipeline
Build In progress
Pipeline execution ID: 8a55e028-ad48-46b6-a76c-67757bf5d5c7
Build
Custom MyJenkinsProviderName (Version: 1)
In progress - Just now
9cff58a0 Source: Change default test port to 80
Disable transition
Deploy Didn't Run


aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Success
Stage Build successfully retried
Build
Custom MyJenkinsProviderName (Version: 1)
In progress - Just now
9cff58a0 Source: Change default test port to 80
Disable transition
Deploy Didn't Run
Deploy
AWS CodeDeploy
Didn't Run
No executions yet
Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of

Step 3: Add another stage to your pipeline

Now you will add a test stage and then a test action to that stage that uses the Jenkins test included in the sample to determine whether the webpage has any content. This test is for demonstration purposes only.

Add a test stage to your pipeline

Topics

Look up the IP address of an instance

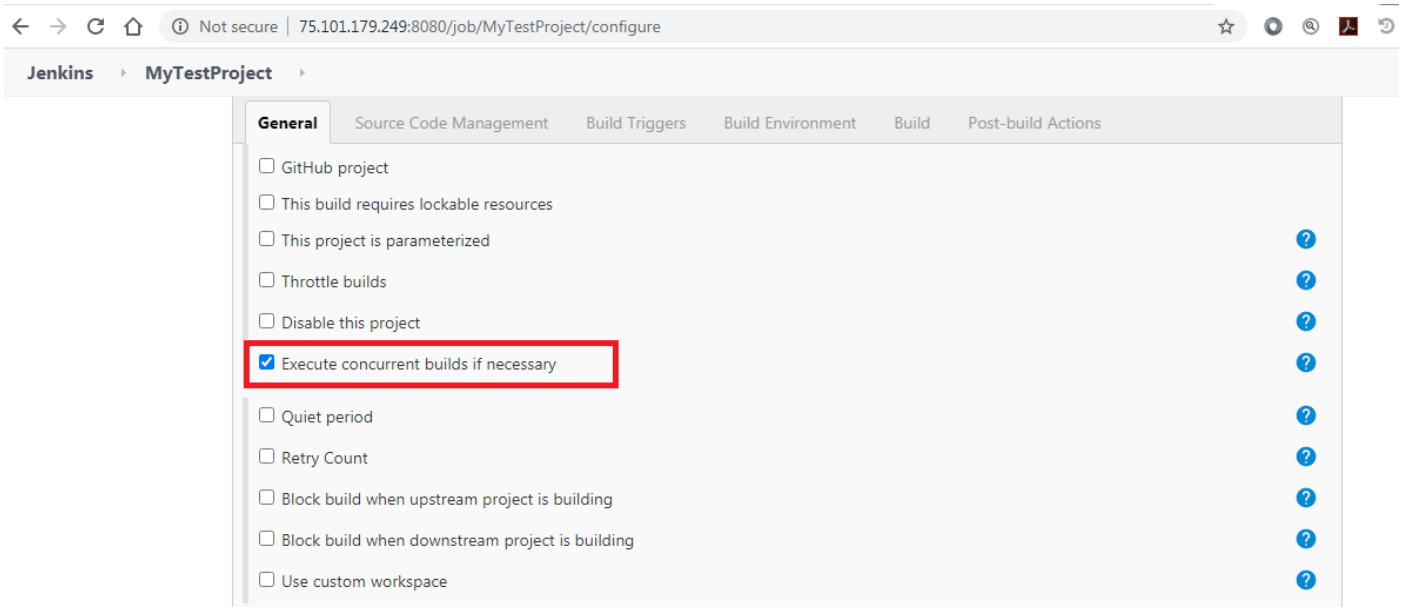
To verify the IP address of an instance where you deployed your code

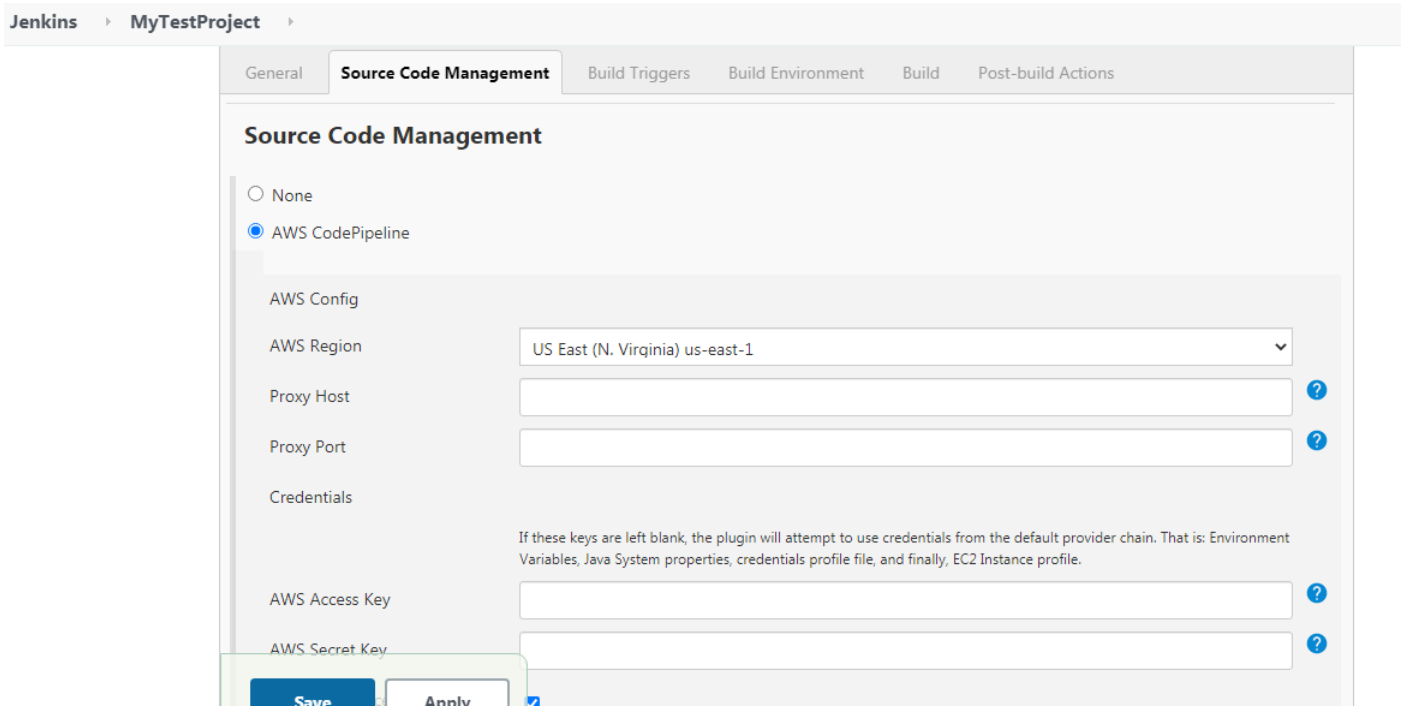
1. After **Succeeded** is displayed for the pipeline status, in the status area for the Staging stage, choose **Details**.
2. In the **Deployment Details** section, in **Instance ID**, choose the instance ID of one of the successfully deployed instances.
3. Copy the IP address of the instance (for example, `172.31.13.175`). You will use this IP address in your Jenkins test.

Create a Jenkins project for testing the deployment

To create the Jenkins project

1. On the instance where you installed Jenkins, open Jenkins and from the main page, choose **New Item**.
2. In **Item Name**, enter a name for the Jenkins project (for example, `MyTestProject`). Choose **Freestyle project**, and then choose **OK**.
3. On the configuration page for the project, select the **Execute concurrent builds if necessary** check box. In **Source Code Management**, choose **AWS CodePipeline**. If you have installed Jenkins on an EC2 instance and configured the AWS CLI with the profile for the IAM user you created for integration between CodePipeline and Jenkins, leave all the other fields empty.

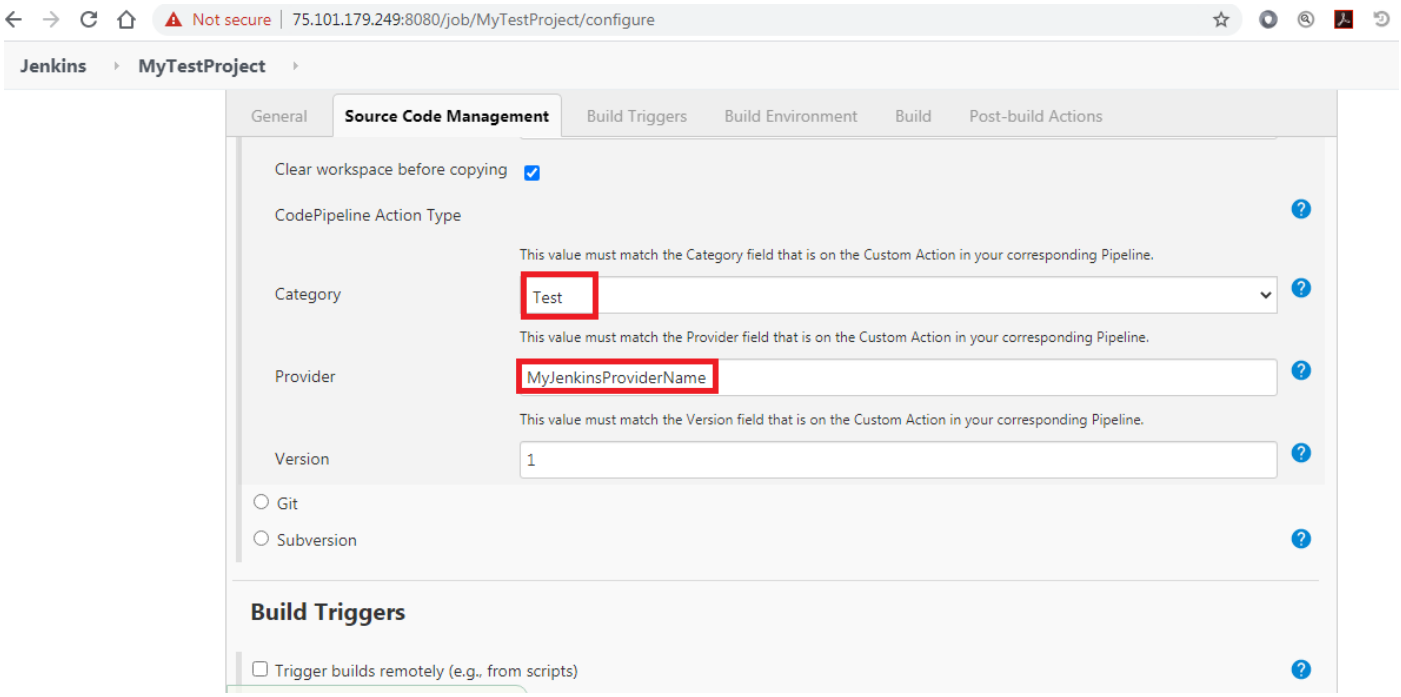




Important

If you are configuring a Jenkins project and it is not installed on an Amazon EC2 instance, or it is installed on an EC2 instance that is running a Windows operating system, complete the fields as required by your proxy host and port settings, and provide the credentials of the IAM user you configured for integration between Jenkins and CodePipeline.

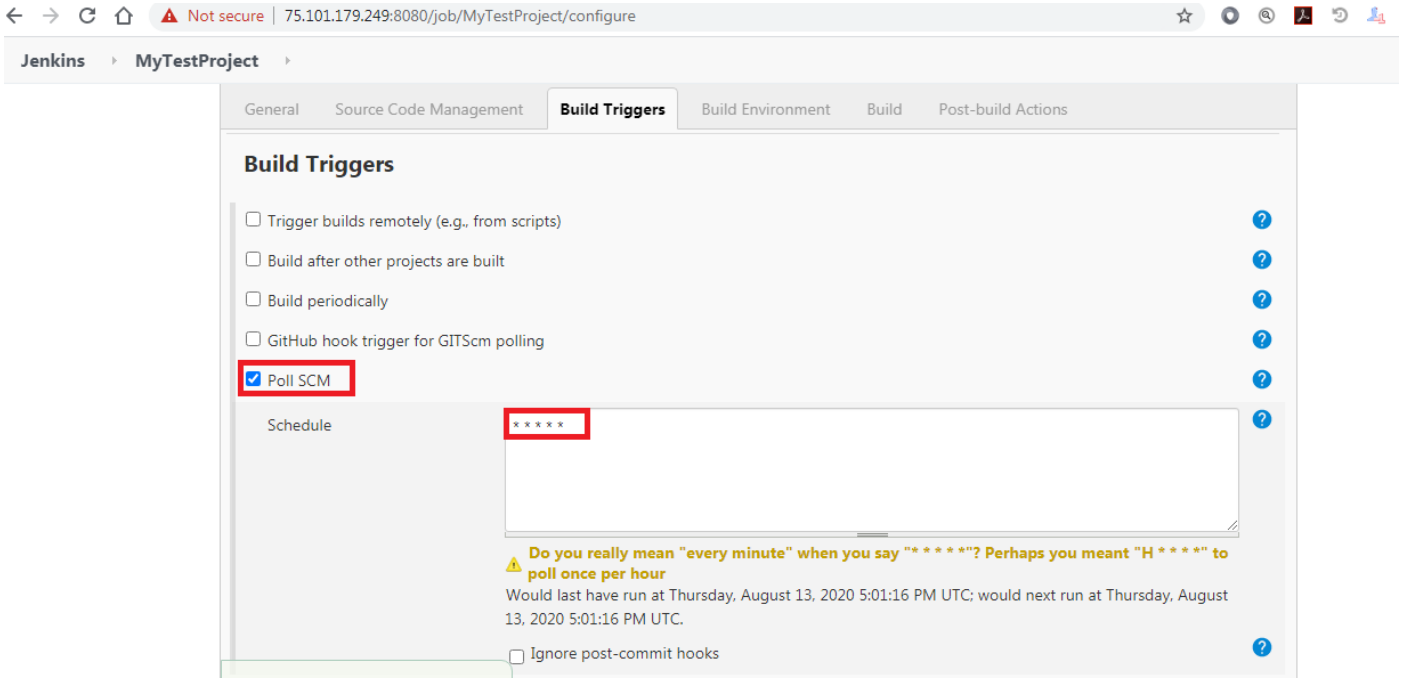
4. Choose **Advanced**, and in **Category**, choose **Test**.
5. In **Provider**, enter the same name you used for the build project (for example, `MyJenkinsProviderName`). You will use this name when you add the test action to your pipeline later in this tutorial.



6. In **Build Triggers**, clear any check boxes, and then select **Poll SCM**. In **Schedule**, enter five asterisks separated by spaces, as follows:

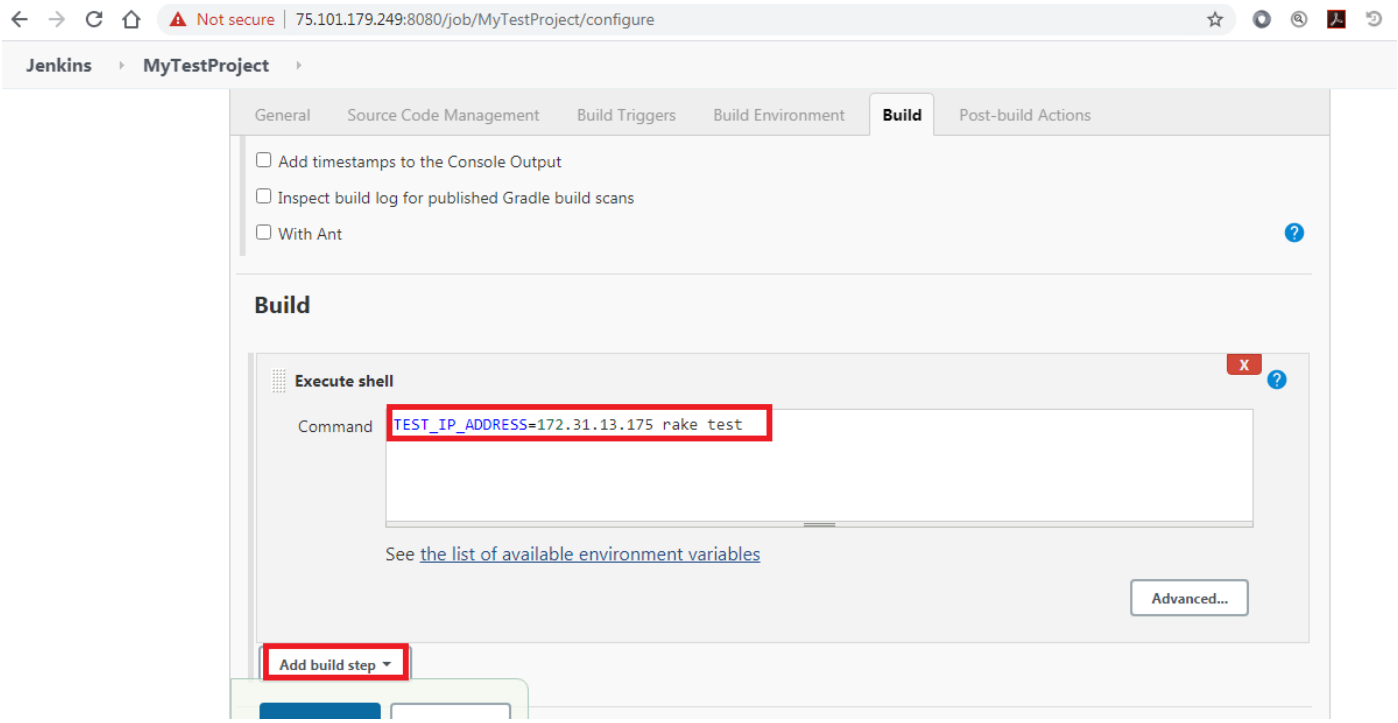
```
* * * * *
```

This polls CodePipeline every minute.



7. In **Build**, choose **Add build step**. If you are deploying to Amazon Linux, RHEL, or Ubuntu Server instances, choose **Execute shell** . Then enter the following, where the IP address is the address of the EC2 instance you copied earlier:

```
TEST_IP_ADDRESS= 172.31.13.175 rake test
```



If you are deploying to Windows Server instances, choose **Execute batch command**, and then enter the following, where the IP address is the address of the EC2 instance you copied earlier:

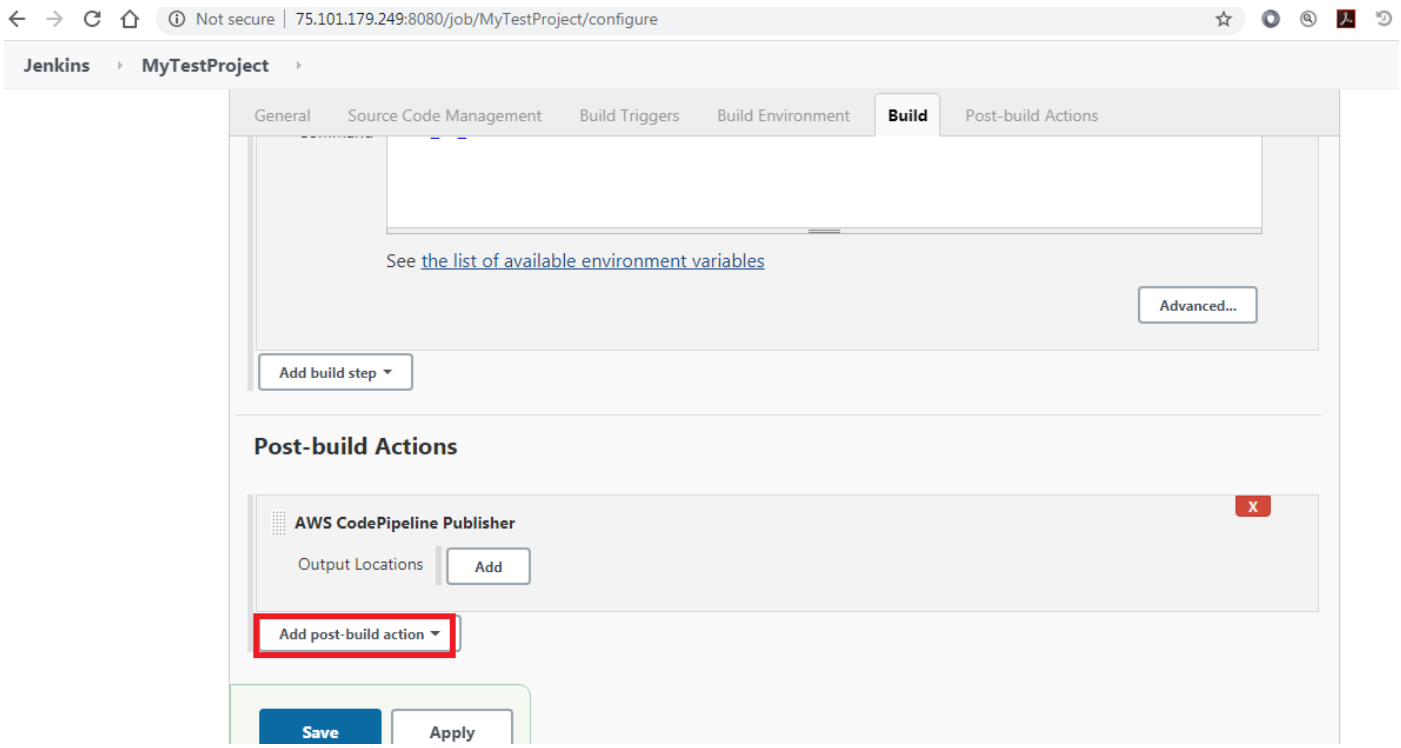
```
set TEST_IP_ADDRESS=172.31.13.175 rake test
```

Note

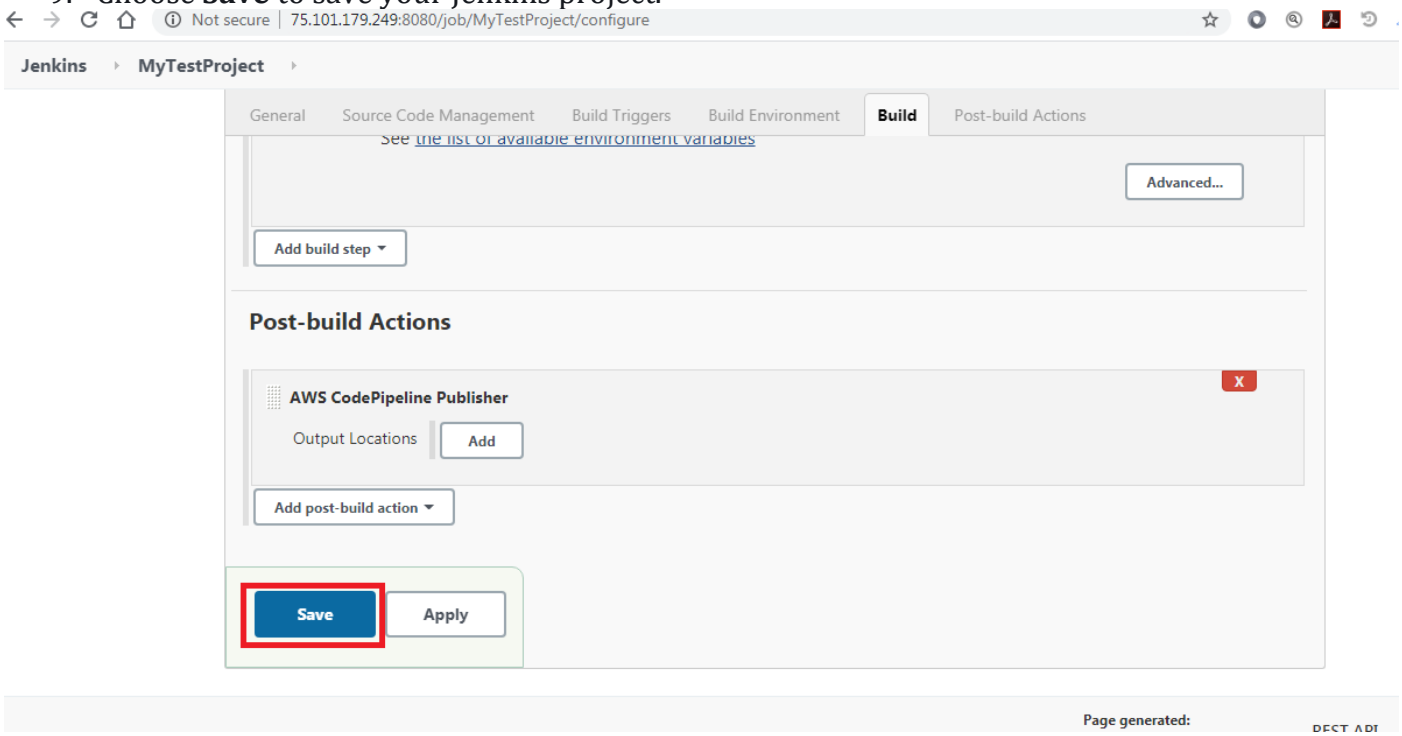
The test assumes a default port of 80. If you want to specify a different port, add a test port statement, as follows:

```
TEST_IP_ADDRESS= 172.31.13.175 TEST_PORT=8000 rake test
```

8. Choose **Add post-build action**, and then choose **AWS CodePipeline Publisher**. Do not choose **Add**.



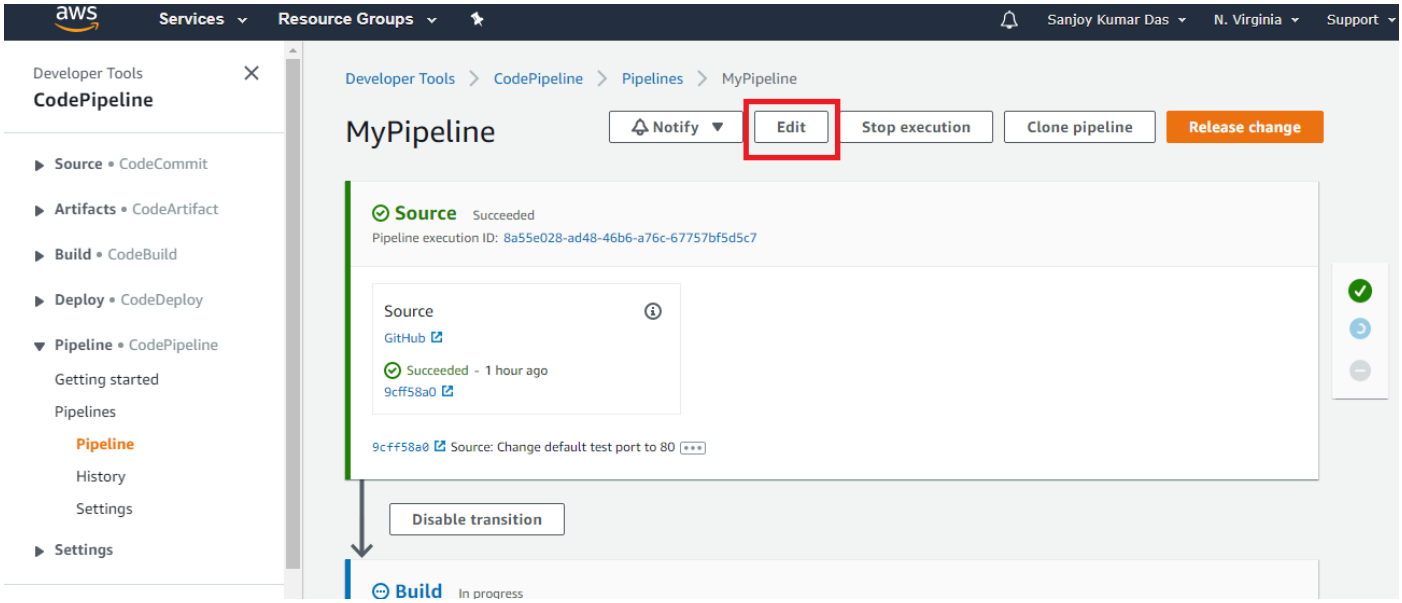
9. Choose **Save** to save your Jenkins project.



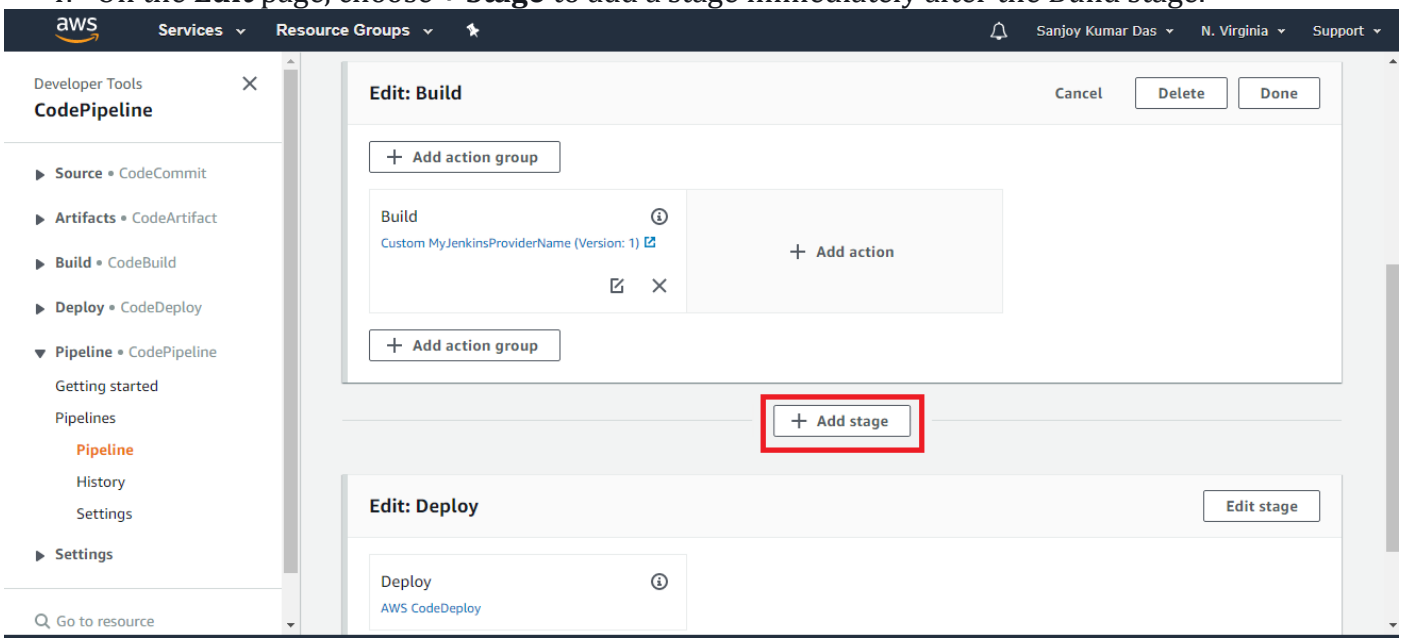
Create a fourth stage

To add a stage to your pipeline that includes the Jenkins test action

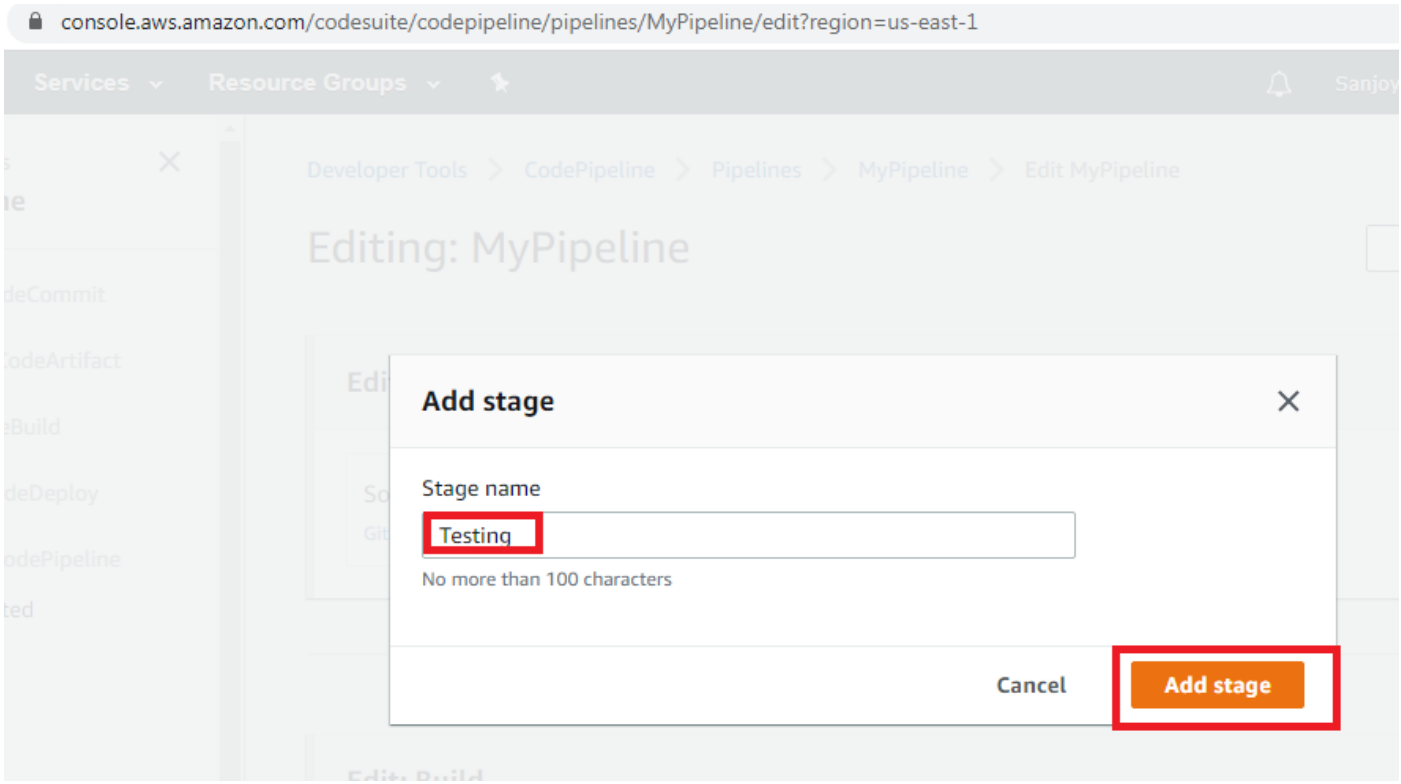
1. Sign in to the AWS Management Console and open the CodePipeline console at http://console.aws.amazon.com/codesuite/codepipeline/home.
2. In **Name**, choose the name of the pipeline you created, MyPipeline.
3. On the pipeline details page, choose **Edit**.

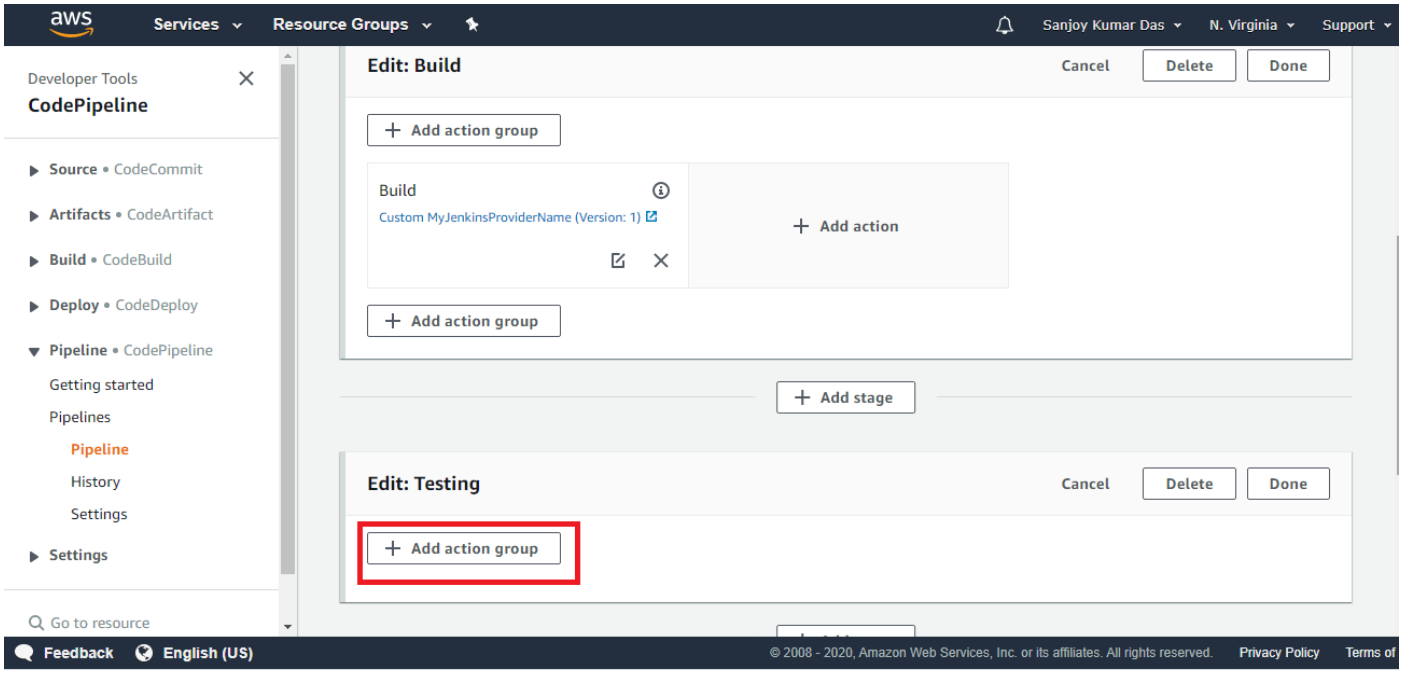


4. On the **Edit** page, choose **+ Stage** to add a stage immediately after the Build stage.

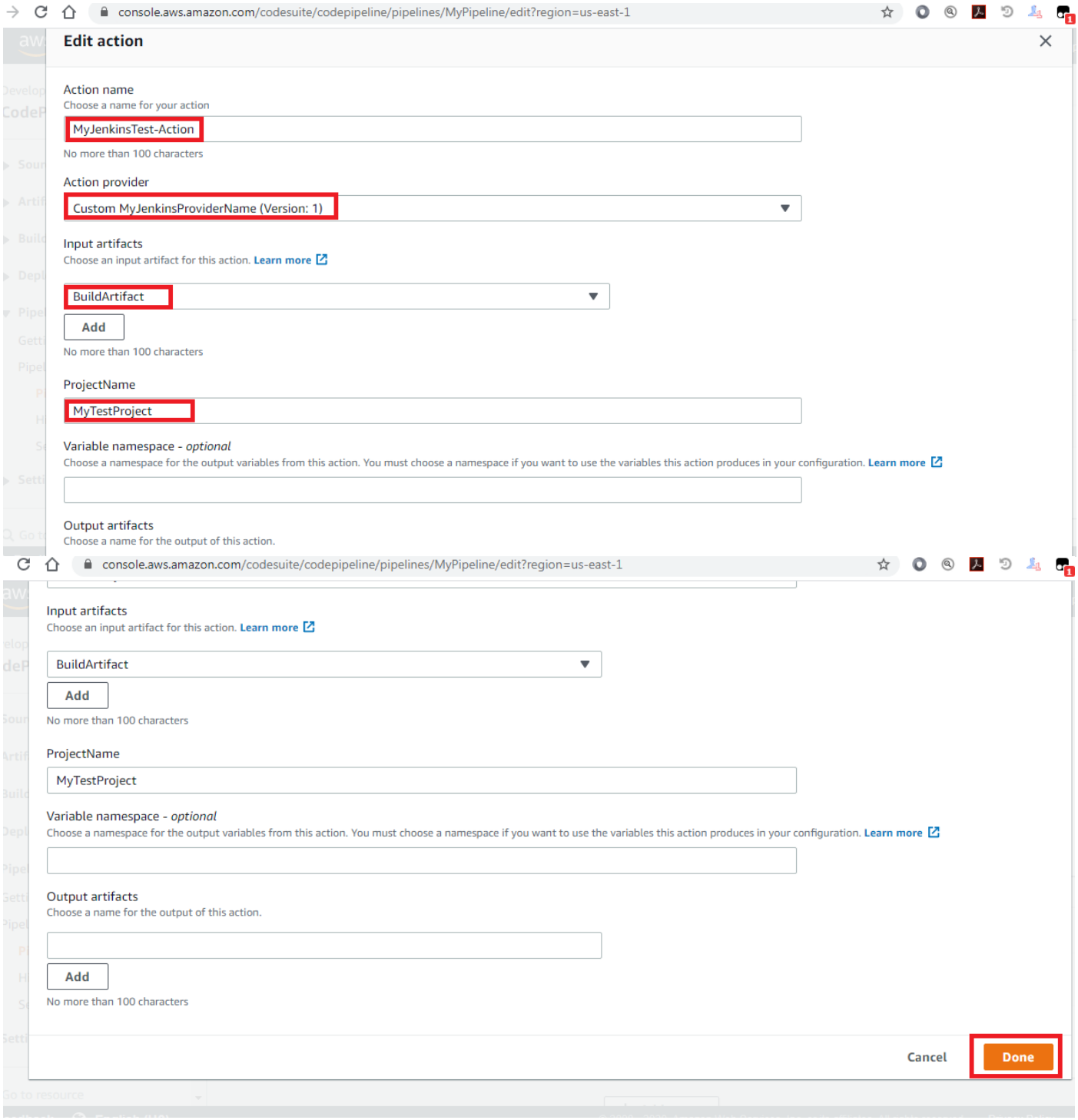


5. In the name field for the new stage, enter a name (for example, `Testing`), and then choose **+ Add action group**.

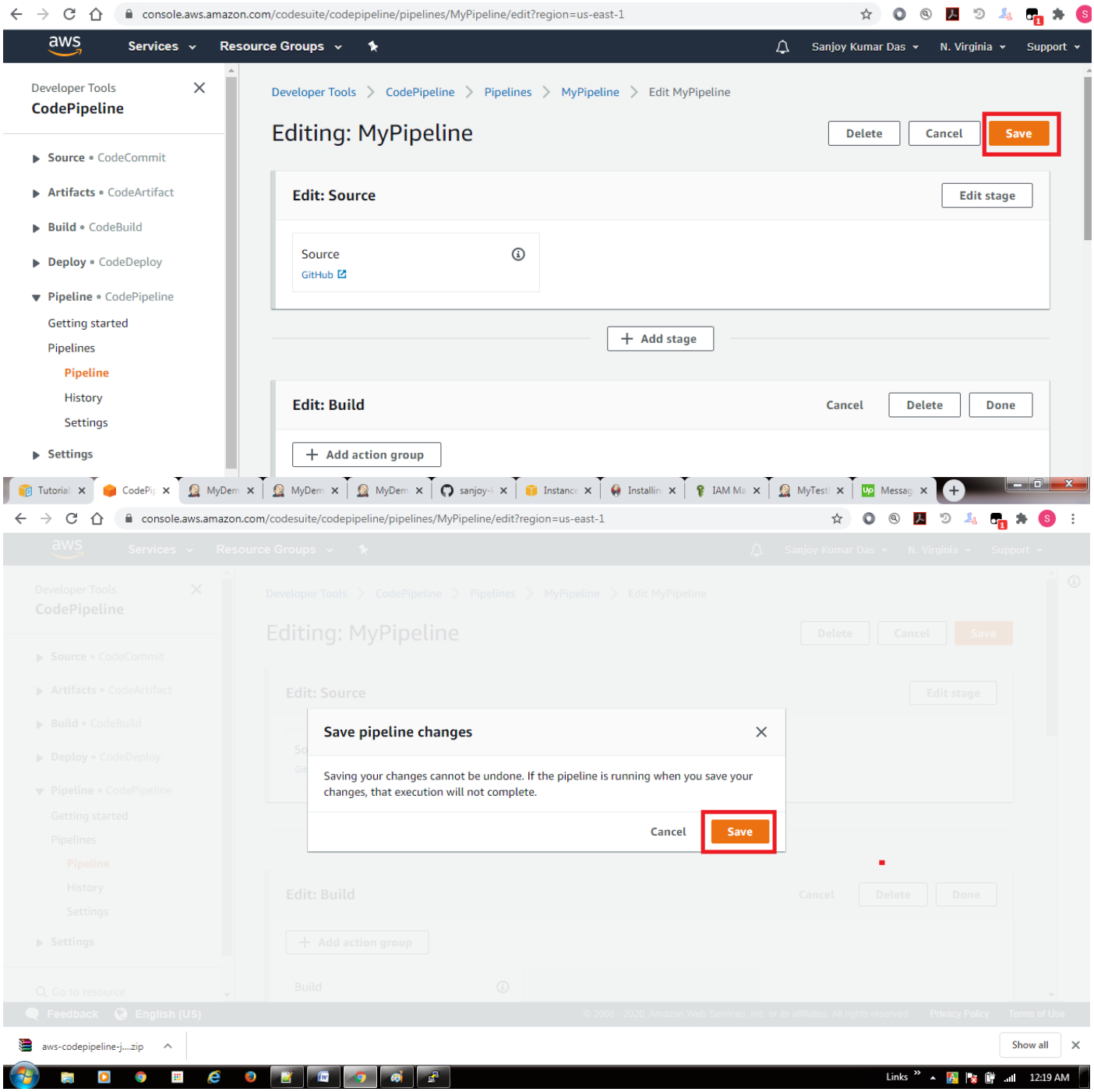




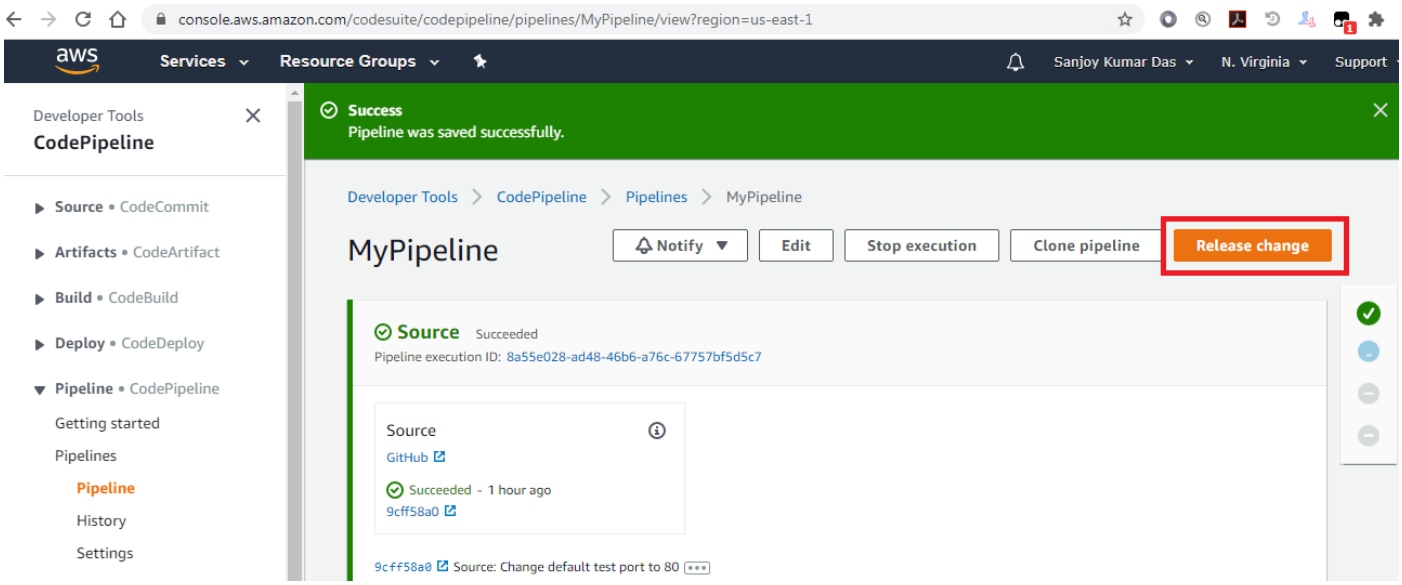
6. In **Action name**, enter *`MyJenkinsTest-Action`*. In **Test provider**, choose the provider name you specified in Jenkins (for example, *`MyJenkinsProviderName`*). In **Project name**, enter the name of the project you created in Jenkins (for example, *`MyTestProject`*). In **Input artifacts**, choose the artifact from the Jenkins build whose default name is *`BuildArtifact`*, and then choose **Done**.

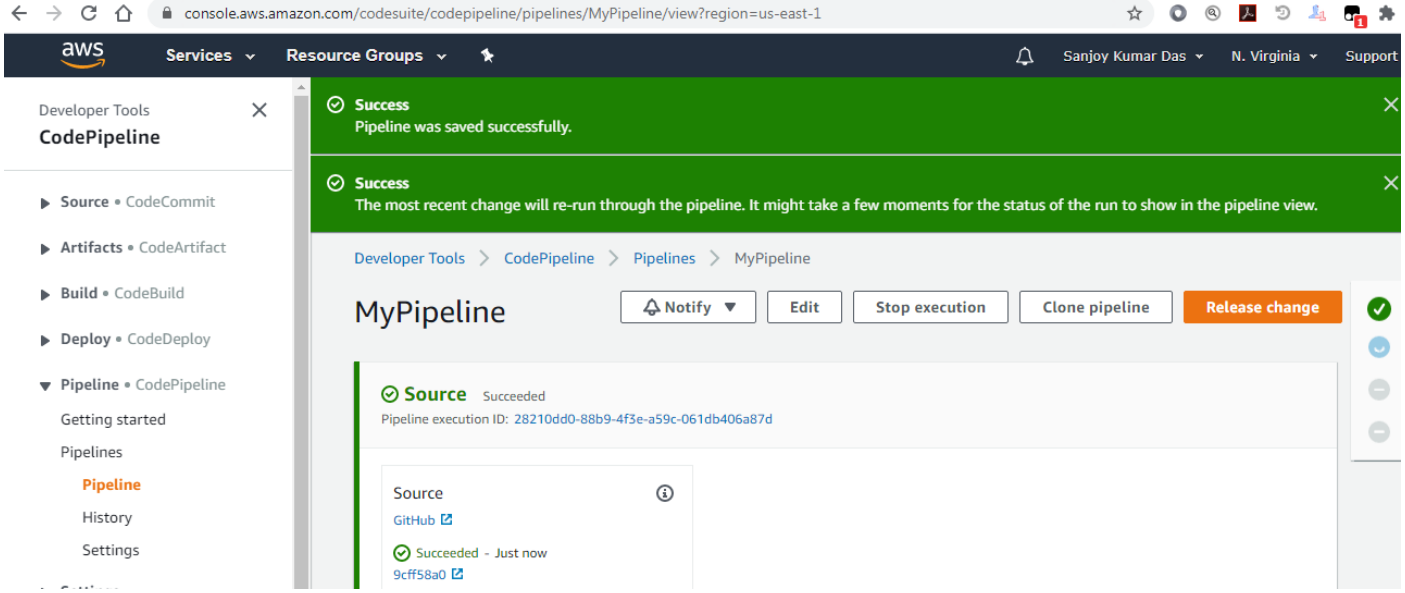


7. On the **Edit** page, choose **Save pipeline changes**. In the **Save pipeline changes** dialog box, choose **Save and continue**.



8. Although the new stage has been added to your pipeline, a status of **No executions yet** is displayed for that stage because no changes have triggered another run of the pipeline. To run the sample through the revised pipeline, on the pipeline details page, choose **Release change**.


console.aws.amazon.com/codesuite/codepipeline/pipelines/MyPipeline/view?region=us-east-1
aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Success
Pipeline was saved successfully.
Developer Tools > CodePipeline > Pipelines > MyPipeline
MyPipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Source Succeeded
Pipeline execution ID: 8a55e028-ad48-46b6-a76c-67757bf5d5c7
Source
GitHub
Succeeded - 1 hour ago
9cff58a0
9cff58a0 Source: Change default test port to 80


console.aws.amazon.com/codesuite/codepipeline/pipelines/MyPipeline/view?region=us-east-1
aws
Services
Resource Groups
Sanjoy Kumar Das
N. Virginia
Support
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Success
Pipeline was saved successfully.
Success
The most recent change will re-run through the pipeline. It might take a few moments for the status of the run to show in the pipeline view.
Developer Tools > CodePipeline > Pipelines > MyPipeline
MyPipeline
Notify
Edit
Stop execution
Clone pipeline
Release change
Source Succeeded
Pipeline execution ID: 28210dd0-88b9-4f3e-a59c-061db406a87d
Source
GitHub
Succeeded - Just now
9cff58a0