AWS-ASSESSMENT

Set up a CI/CD pipeline with Jenkins and an EC2 instance. The source will be on GitHub.

**Steps:**

1. Sign in to the AWS Management Console.

2. Open the Amazon EC2 console by selecting EC2 under Compute.

3. From the Amazon EC2 dashboard, select Launch Instance.

4. The Choose an Amazon Machine Image (AMI) page displays a list of basic

configurations called Amazon Machine Images (AMIs) that serve as templates for

your instance. Select the HVM edition of the Amazon Linux AMI.

5. Scroll down and select the key pair you created in the creating a key pair section

above or any existing key pair you intend to use.

**SECURITY GROUP:**

1. In Security group name, enter jenkinserver57-sg.

2. On the Inbound tab, add the rules as follows:

3. Select Add Rule, and then select SSH from the Type list.

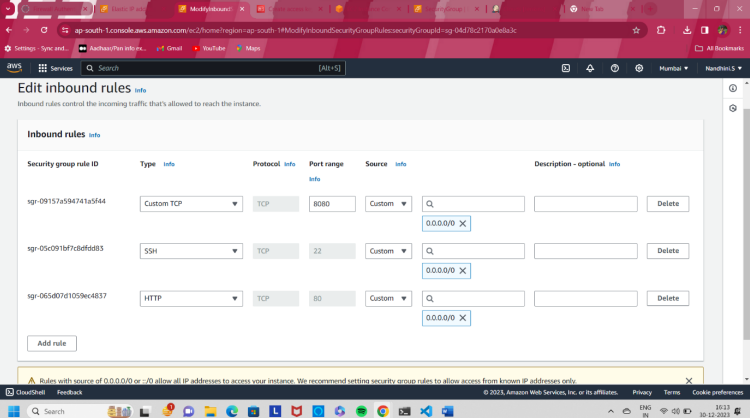
4. Under Source, select Anywhere IP

5. Select Add Rule, and then select HTTP from the Type list.

6. Select Add Rule, and then select Custom TCP Rule from the Type list.

7. Under Port Range, enter 8080.

8. Select Create



**INSTALLING JENKINS**

• Using SSH to connect to your instance

• Use the ssh command to connect to the instance. You will specify the private key (.pem) file and ec2-user@public\_dns\_name

**Configuring Jenkins**

STEPS:

• Jenkins is now installed and running on your EC2 instance.To configure Jenkins:

• Connect to http://ec2-15-207-223-126.ap-south-1.compute.amazonaws.com:8080

from your browser.

The Jenkins installation script directs you to the Customize Jenkins page. Click Install

suggested plugins.

• Once the installation is complete, the Create First Admin User will open. Enter your

information, and then select Save and Continue.

• On the left-hand side, select Manage Jenkins, and then select Manage Plugins.

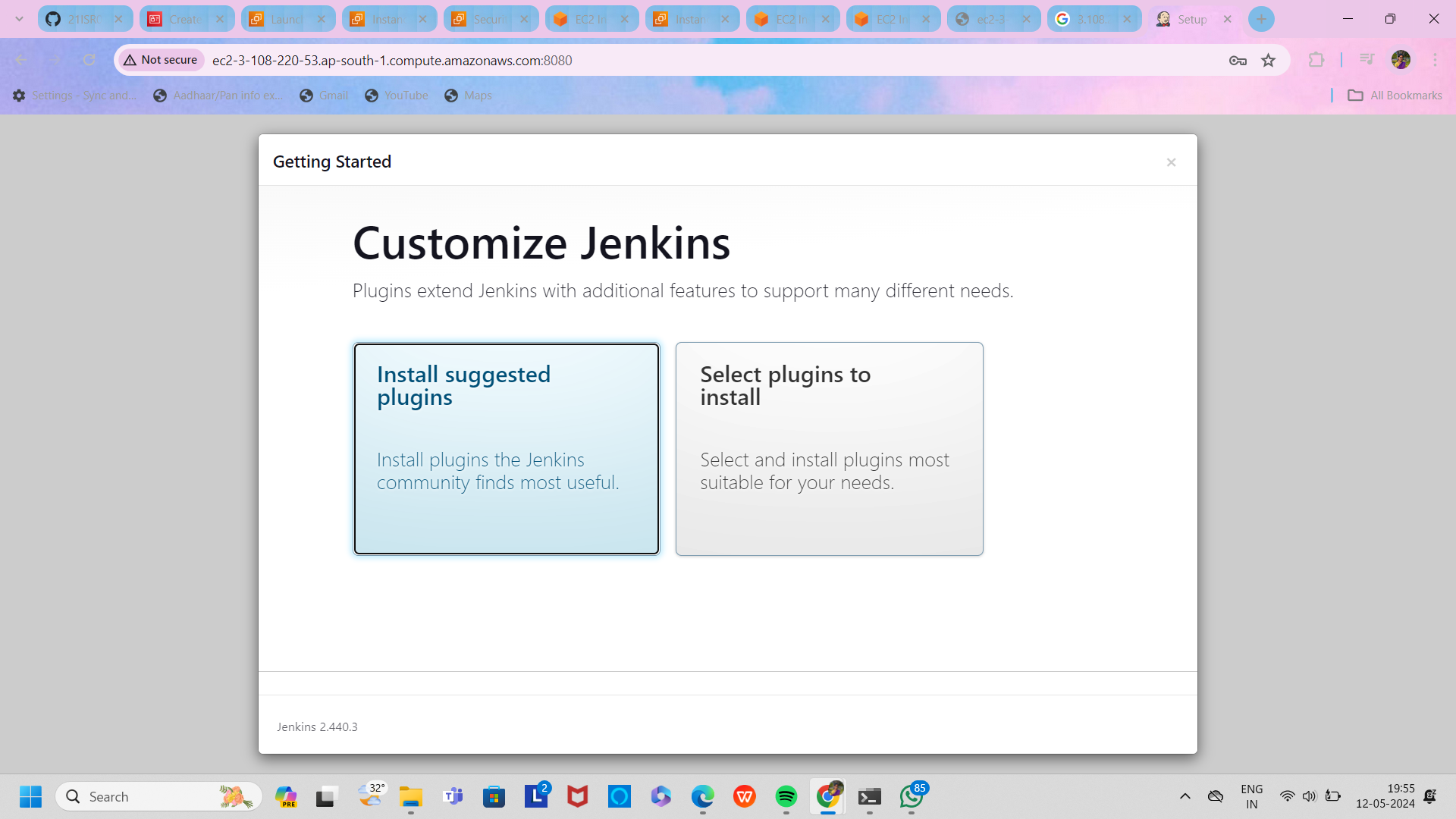
• Select the Available tab, and then enter Amazon EC2 plugin at the top right.

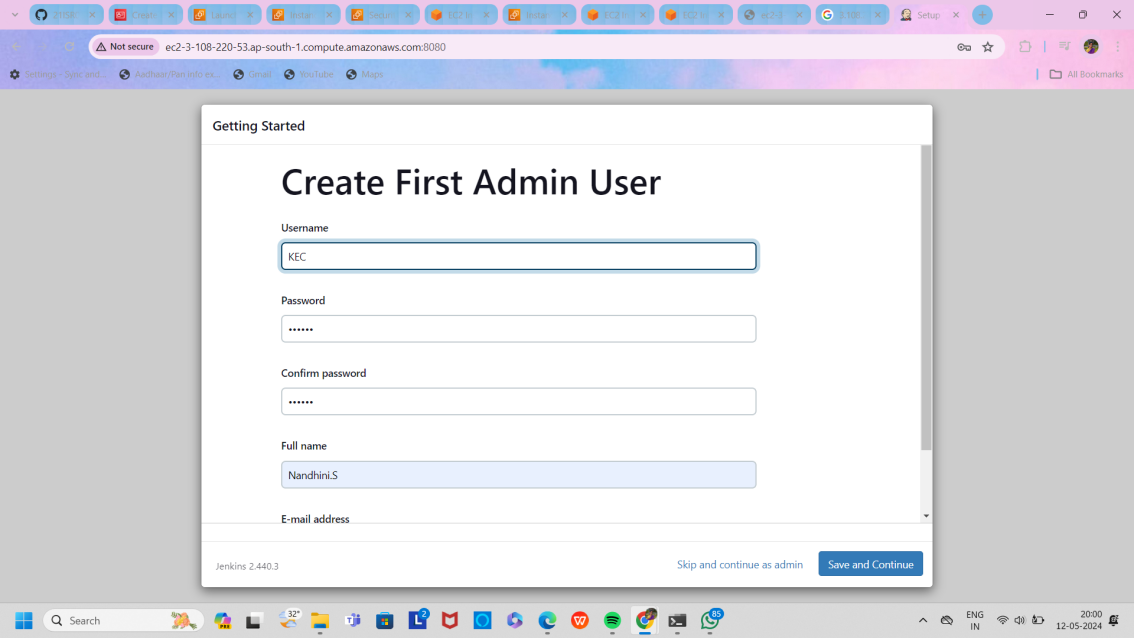
• Select the checkbox next to Amazon EC2 plugin, and then select Install without

restart.

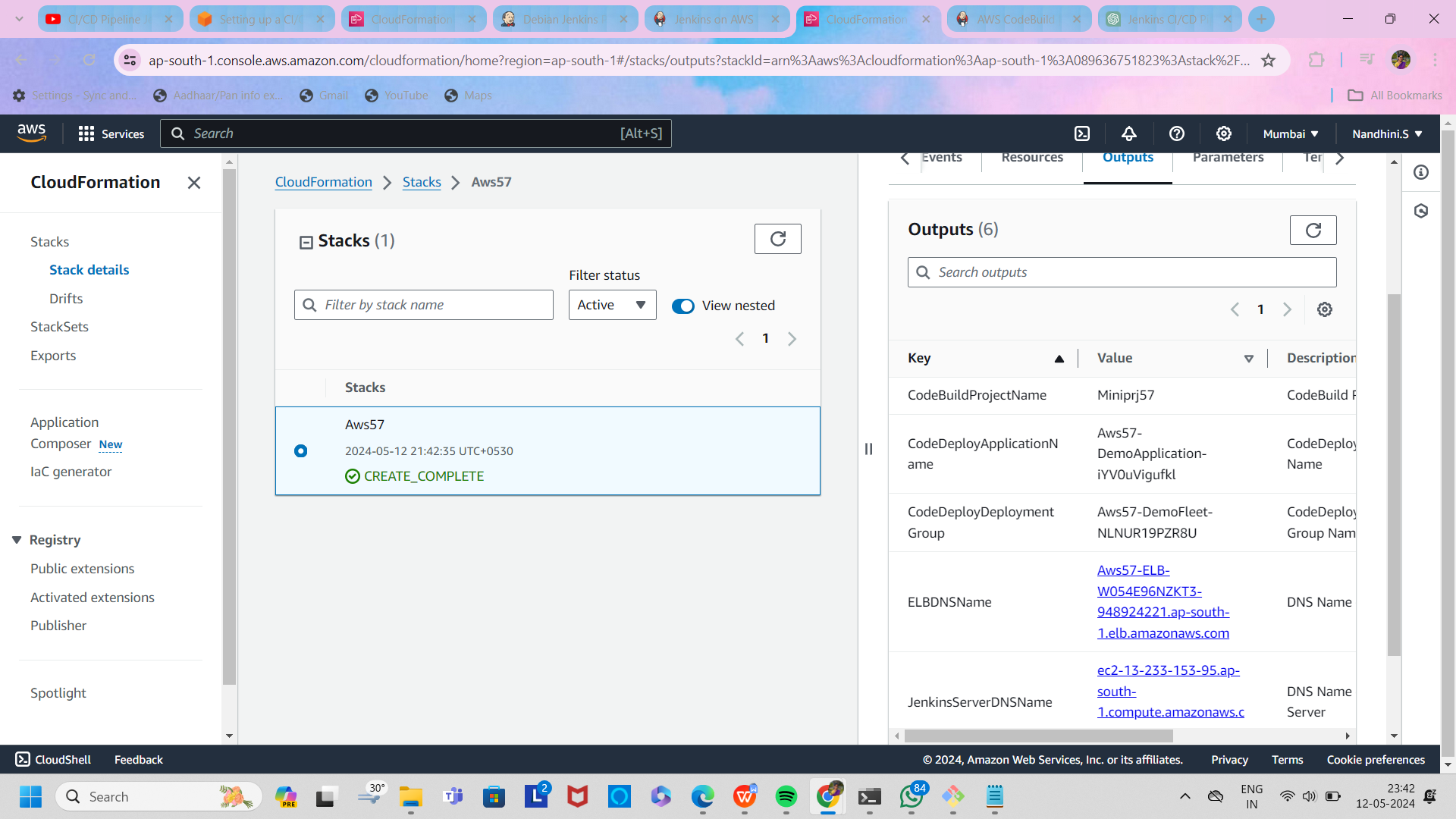
• Select Add a new cloud, and select Amazon EC2. A collection of new fields appears.

• Select Save when done.





* To create the CloudFormation stack (for example in the AWS Frankfurt Region) click the below link:
* Choose Next and provide the following values on the Specify Details page:
* For Stack name, name your stack as you prefer.
* For CodedeployInstanceCount, choose the default of t2.medium.
* To check the supported instance types by AWS Region, see Supported Regions.
* For InstanceCount, keep the default of 3, to launch three EC2 instances for CodeDeploy.
* For JenkinsInstanceType, keep the default of t2.medium.
* For KeyName, choose an existing EC2 key pair in your AWS account. Use this to connect by using SSH to the Jenkins server and the CodeDeploy EC2 instances. Make sure that you have access to the private key of this key pair.
* For PublicSubnet1, choose a public subnet from which the load balancer, Jenkins server, and CodeDeploy web servers launch.
* For PublicSubnet2, choose a public subnet from which the load balancers and CodeDeploy web servers launch.
* For VpcId, choose the VPC for the public subnets you used in PublicSubnet1 and PublicSubnet2.
* For YourIPRange, enter the CIDR block of the network from which you connect to the Jenkins server using HTTP and SSH. If your local machine has a static public IP address, go to https://www.whatismyip.com/ to find your IP address, and then enter your IP address followed by /32. If you don’t have a static IP address (or aren’t sure if you have one), enter 0.0.0.0/0. Then, any address can reach your Jenkins server.



**Create a project and configure the CodeDeploy Jenkins plugin**

Enter the following values from the Outputs tab of your CloudFormation stack and leave the other settings at their default (blank):

For AWS CodeDeploy Application Name, enter the value of CodeDeployApplicationName.

For AWS CodeDeploy Deployment Group, enter the value of CodeDeployDeploymentGroup.

For AWS CodeDeploy Deployment Config, enter CodeDeployDefault.OneAtATime.

For AWS Region, choose the Region where you created the CodeDeploy environment.

For S3 Bucket, enter the value of S3BucketName.

The CodeDeploy plugin uses the Include Files option to filter the files based on specific file names existing in your current Jenkins deployment workspace directory. The plugin zips specified files into one file. It then sends them to the location specified in the S3 Bucket parameter for CodeDeploy to download and use in the new deployment.

In the optional Include Files field, I used (\*\*) so all files in the workspace directory get zipped.

Unzip the application files and send them to your GitHub repository, run the following git commands from the path where you placed your sample application:

$ git add -A

$ git commit -m 'Your first application'

$ git push

