

Individual Assignment

XDCS2054N Cloud Computing



August 2, 2024

nan Hat SHang Kham

Student ID- 0136015

**Table of Contents**

1. Introduction
2. Setup Process
3. Testing
4. Challenges and Solutions
5. Conclusion
6. References

**Introduction**

A simple html website which will later be utilized as a resume site is deployed using git for version control, Docker for containerization, Jenkins for Automation and Docker Hub as a repository to manage Docker images created. The main purpose of this project is to showcase the deployment of a website with continuous integration and continuous deployment (CI/CD). With CI/CD, the process of the deployment of website becomes easier with less human intervention which allows the reduction of errors along the deployment process if the prerequisites are fulfilled.

**Setup Process**

Prerequisite- a local machine or virtual machine

1. Setting up AWS EC2 instance

Firstly, an AWS account is a must to create an EC2 instance. Thus, it is required to sign up using a valid Gmail account of your choice. After that, through the console of AWS, the EC2 Dashboard can be reached where you will be able to launch and set up your EC2 instance with respect to AMI or Amazon Machine Image of your choice. If this is your first time using EC2 service, it is recommended to create a key pair and save it in your Virtual Machine or local computer which can later be used if you want to connect your EC2 instance and Virtual Machine using ssh command. The most important key point in setting up an AWS instance is to allow SSH or port 22 for remote access in your security group. Without this inbound rule, your instance will have SSH error. It is also recommended to allow port 80 or HTTP for your web traffics and additional ports based on your services. After that, your EC2 instance is now ready to be connected.

Important points

The public DNS and IP address of your instance would change every time you pause your instance. Moreover, if you have connection issues with your instance, it is recommended to reboot your instance.

1. Connecting EC2 instance with VM or local machine

If the EC2 instance can be connected successfully, it is optional for you to connect your virtual machine or local computer with EC2 using SSH command through terminal. If your local machine or VM has never been connected to an EC2 instance with a particular public IP address, it will not recognize the instance and the machine will double confirm you for connecting with that EC2 instance.

A computer screen with text

Description automatically generated

SSH command to connect with EC2

Important Points

It is always recommended to check the available updates and upgrades.

1. Installing and setting up Git

For the installation of Git, it is required for your EC2 instance and VM to be connected unless you are using the EC2 terminal directly. Firstly, check the available updates and install Git using “$ sudo apt-get install git -y” command. After that, you can choose to configure your username and email address of your git which is supposed to be the same as your GitHub user name and email address. Thus, if you haven’t registered for GitHub, it is required to do at this stage. After the successful installation, you can check the version of Git using “$ git -version” or check the status of Git using systemctl command. It is suggested to create SSH key for GitHub which is optional.

A screenshot of a computer program

Description automatically generated

Git Installation and Setting Up

A computer screen with white and orange text

Description automatically generated

Generation of SSH Key

A screenshot of a computer

Description automatically generated

SSH key for GitHub and Jenkins

1. Cloning the repository from Git

First, you will need to create a directory in your EC2 instance to clone your GitHub Repository into the directory. After that, using your git commands, you can commit and push the changes to GitHub. In this scenario, my GitHub repository is created first and through vscode, I connected my GitHub to commit to my GitHub.

A screen shot of a computer

Description automatically generated

Example of Pushing to GitHub

1. Installing and setting up Docker

At this stage, you can install docker to your EC2 instance and based on the AMI of your choice, your installation process might be different. In this project, Ubuntu is used as an AMI and thus the following commands are to be executed to install Docker. After that, the docker should be started and enabled to test. At this point, it is possible to create a docker image and a docker container using the docker commands and deployed it on the EC2 instance.

A screenshot of a computer

Description automatically generated

Status Of Docker Running

1. Signing up Docker Hub

In this case, it is required to sign up for the Docker Hub first as the Jenkins Pipeline is integrated with Docker Hub and its credentials would be stored in Jenkins. After that, it is recommended to retrieve the Personal Access Tokens to login on your EC2 instance. These tokens will later be used in Jenkins as Docker Hub credentials for security purposes. The following command line is executed and the Personal Access Tokens (PAT) are provided in order to connect to Docker Hub.

A black screen with white text

Description automatically generated

Access to Docker Hub from EC2 with PAT

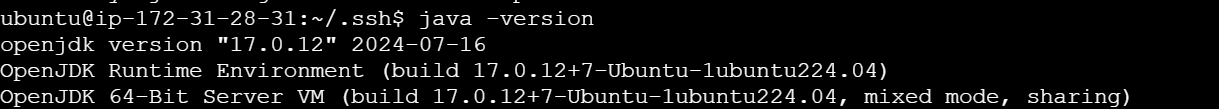
1. Installing and setting up Jenkins

It is required to install OpenJDK first into your EC2 instance and it is recommended to install OpenJDK 17 because OpenJDK 11 would not be supported anymore by the end of this year for the latest version of Jenkins. If Java is already in your EC2, you can check with the command “$java version”. By following the commands, the installation of Jenkins could be done.

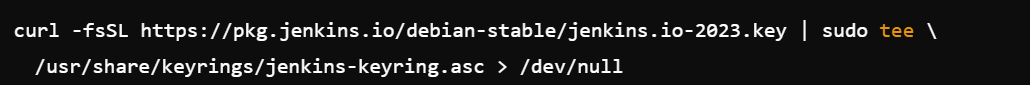
A black screen with white text

Description automatically generated

Installation of Java



Checking Java Version



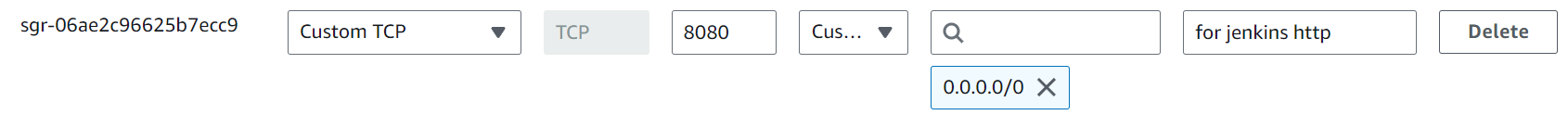
Installation of Jenkins from Debian with GPG as key verification

A computer screen shot of a computer code

Description automatically generated

Adding Jenkins Repository to Sources List

After that, add an inbound rule for Jenkins in EC2 and access to Jenkins using that port number 8080 and set up Jenkins with necessary plugins.



Additional Inbound Rule in EC2

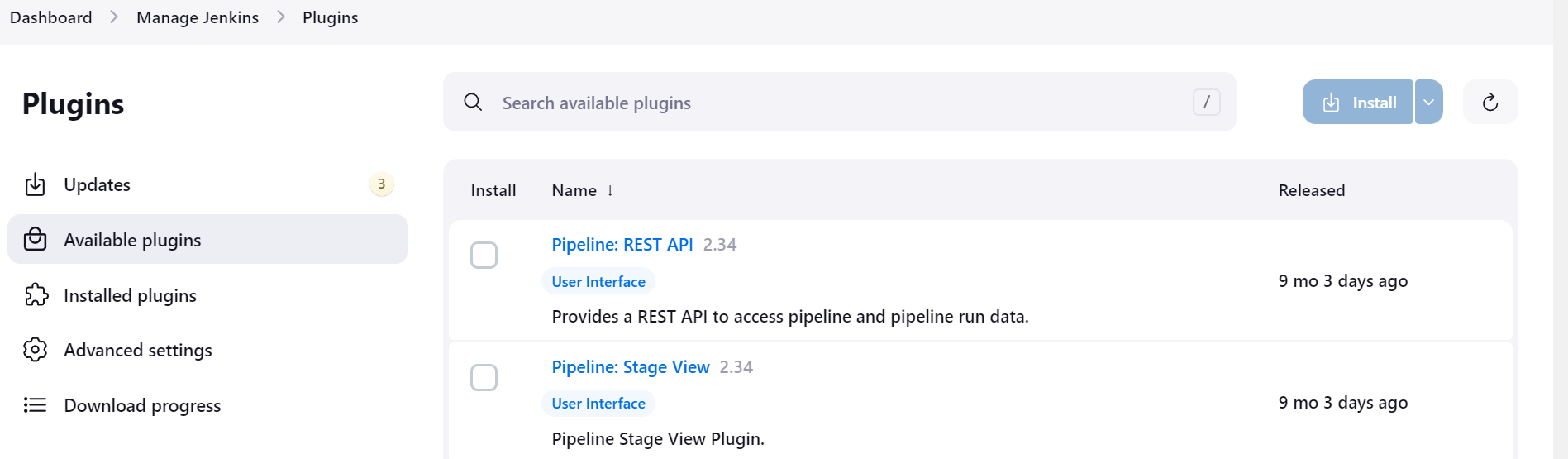
After successfully downloading Jenkins, you should see the Dashboard of Jenkins like the following image and to create a new Jenkins pipeline or any project, you can click “New Item” on the left side of the dashboard.

A screenshot of a computer

Description automatically generated

Jenkins Dashboard

Set up the Jenkins by adding plugins based on your pipeline. In this case, Git, Docker Pipeline are necessary. After that, the credential for Docker Hub is added to Jenkins Credentials and the Jenkin pipeline for automation purpose is set up. The reason why Docker Hub credential is added is because of the additional security reasons of your private information and the PAT retrieved from Docker Hub will be used as the Docker Hub credentials here. The SSH key is also added just in case. After that, the pipeline can be saved.



Manage Jenkins tab to add plugins

A screenshot of a computer

Description automatically generated

Credentials in Jenkins

The pipeline of Jenkins in this project is built in order to get the Docker Hub credentials first from Jenkins in order to access to Docker Hub and then, check the updates in GitHub in order to update the pipeline accordingly.

A computer screen shot of a computer

Description automatically generated

Jenkins Pipeline part-1

After that, the pipeline would build the Docker Image from the newest update of GitHub with the image name “mulan20/Jenkins-integration:latest”. The image will then be pushed onto the Docker Hub with the latest tag.

A screen shot of a computer code

Description automatically generated

Jenkins Pipeline part-2

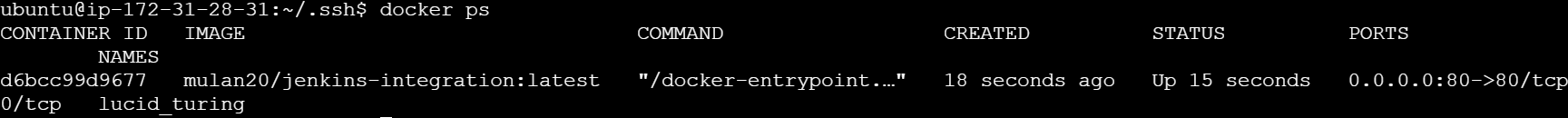
After that, the docker image would be deployed into the docker container on port 80:80 because port 8080 is being used by Jenkins now. In this stage, you can choose to delete the previous Jenkins container running with the same name so that you don’t need to manually delete the former built container. In this case, the pipeline would be successfully only after the previous container built from the pipeline is required to be deleted first.

A close-up of a computer code

Description automatically generated

Jenkins Pipeline part-3

After that, you can just the docker running on your EC2 by using the command “docker ps”. If the docker container is already, you have successfully deployed your website and you can check it on any search engine using your Public IP address.



Running Docker Container

A screenshot of a computer

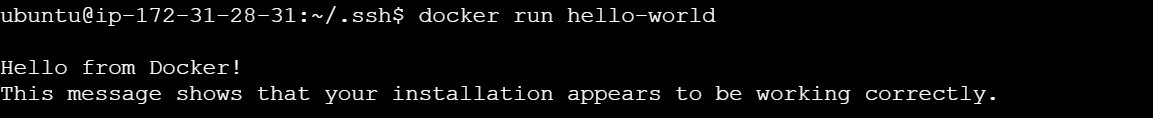
Description automatically generated

Running Website

**Testing**

1. Testing Docker

In order to test whether is docker is running or not, you will need to test by running the hello-world image from Docker Hub. If you docker is running successfully, it should show the following message. Moreover, you can also check the status of your docker by “sudo systemctl status docker”.



Docker Running Test

1. Testing Jenkins

In order to test Jenkins, you must have successfully installed Java and Jenkins first. After that, Jenkins status could be checked and if the Jenkins is not running, you can start and enable Jenkins using systemctl commands. If the Jenkins is running, you can access to the Jenkins through the port 8080.

A computer screen with white text

Description automatically generated

Running Jenkins

**Challenges and Solutions**

1. EC2 instance ssh problem

It is crucially for your EC2 instance to have a SSH inbound rule added to it as this could lead to the SSH problem which will cause your troubles in connecting to your EC2 instance.

1. Pipeline failures

For a novice, it is normal to have failures for the first few build-ups of your Jenkins pipeline. It can be solved by reviewing the console output of your failed pipeline built where they would provide the detailed process of your pipeline with the reason of failure or error.

1. Open JDK 17 vs Open JDK 11

When working with the latest version of Jenkins or 2.470, it is suggested for the JDK 17 to be used. Even though JDK 11 is still supported until September of this year, there are struggles when it comes to pipeline built-up. Thus, it is required to install the JDK 17 as provided in the “Setting Up” session.

1. EC2 Billing Issues

As a free-tier user, it is necessary to read the agreements which will be pushed by AWS regarding the monthly free tier limitations. However, more than one EC2 instance was created in this scenario and the free-tier limit almost exceeded accidentally. Thus, this issue is solved by opening a new Gmail account with which I used to sign up again or AWS.

**Conclusion**

To conclude, the entire process of deployment is enjoyable with many new things to learn. Chances are given to learn new technologies like Docker, integration of pipeline for automation. By setting up an EC2 instance, I got to learn how AWS free tier and EC2 works. Moreover, it is very powerful to be working with Git which allows you to update and your project with very little effort and automation allows you to simplify your process of deployment even though the initial setup of pipeline could be difficult for novice. If I could redo the project, I would try to connect my pipeline with SSH instead of Docker Hub which I failed to implement in my pipeline and I will try to study about CI/CD first before starting the project which I couldn’t do because of time limitations.

**References**

GeeksforGeeks. (2024, July 8). *Building with Docker Using Jenkins Pipelines*. GeeksforGeeks. https://www.geeksforgeeks.org/building-with-docker-using-jenkins-pipelines/