Individual Assignment

COMP 3104 – DevOps

Total Points (10%)

Submission Deadline: Sunday, 26th Nov 2023, 11:59 PM (Week 12)

Not submission extension as it might affect other coursework

Please do research if any command is not working. Command list are only for you reference.

Todo Checklist and Rubric

Sr. #	Task	Description	Marks

1	Create Azure / AWS Ubuntu VM instance	Install Java, Jenkins, Docker, and Git
		*Submit Screenshots of all command list you have executed with proper name. For example, java installation command list screenshot name like **java.png.***
		Share Jenkins URL
			Java = 10
			Docker = 10
			Git = 05
			GitHub Repo = 05
			Jenkins = 30
			Only Azure VM OR AWS EC2 instance=05

2	Create Azure/AWS Ubuntu VM instance	Install Java (***prerequisite***), Postgres and SonarQube Server *Submit Screenshots of all command list you have executed with proper name. For example, postgres installation command list screenshot name like **postgres.png.*** ***Share SonarQube URL***	Postgres = 10 SonarQube Server = 20 **Only Azure VM OR AWS EC2 instance=05**

If you are on AWS account, then follow below steps to create EC2 instance otherwise refer another document or move to java installation given in these document.

****

*********** Creating to AWS EC2 *********

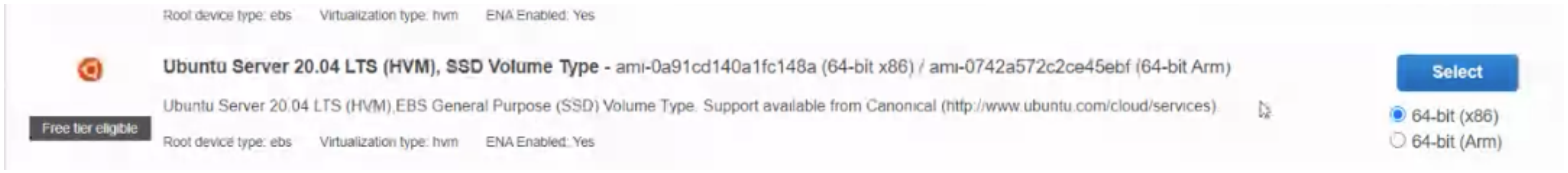
****

NOTE: Students are free to use nay cloud services (AWS, GCS, MS Azure, etc.) to complete the assignment. Please submitted relevant screens shots to validate your work. Below steps are only to create AWS EC2 instances. All other steps remains same for Ubuntu OS.

Login to AWS Console https://aws.amazon.com/console/
Student Account: https://www.awseducate.com/student/s/classrooms

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html

Select Ubuntu server on EC2 as below

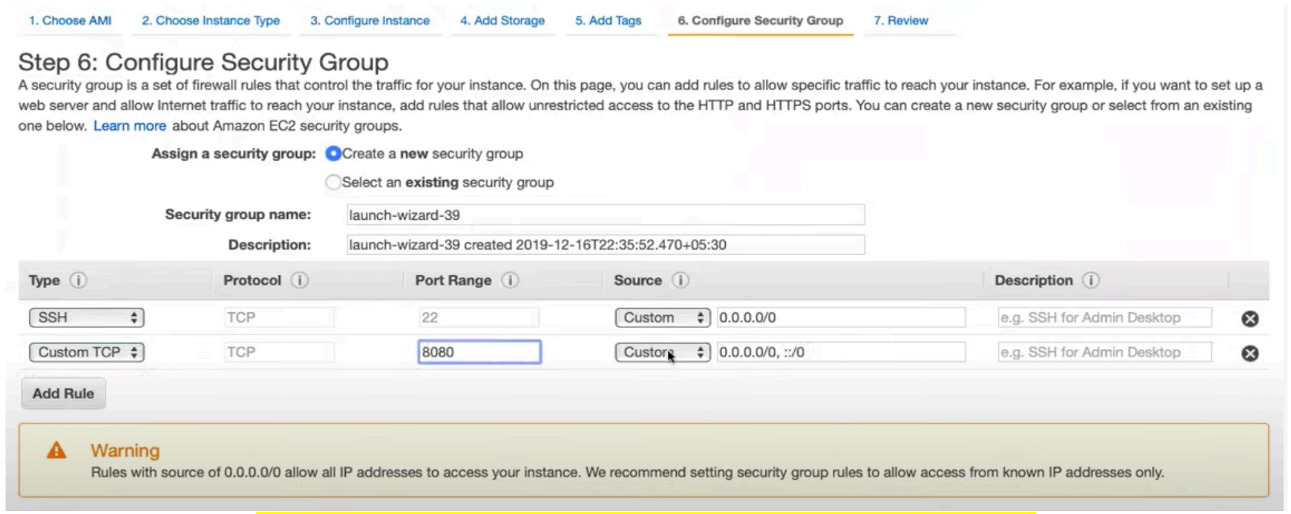


Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-0a91cd140a1fc148a (64-bit x86) / ami-0742a572c2ce45ebf (64-bit Arm)

Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services)

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes



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: launch-wizard-39

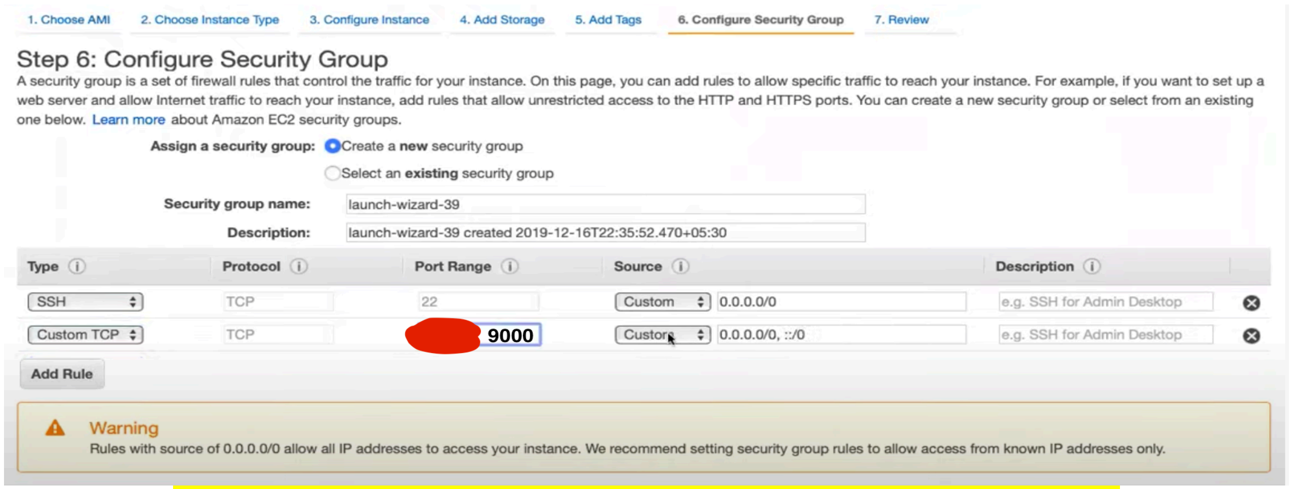
Description: launch-wizard-39 created 2019-12-16T22:35:52.470+05:30

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP	TCP	8080	Custom	0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Add Rule

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Add Custom TCP having port 8080 for Jenkins



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: launch-wizard-39

Description: launch-wizard-39 created 2019-12-16T22:35:52.470+05:30

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
Custom TCP	TCP	[redacted] 9000	Custom	0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Add Rule

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Add Custom TCP having port 9000 for SonarQube Server

********* Connecting to AWS EC2 *******

To access your instance:

Open an SSH client. (Find out how to connect using PuTTY on windows)
Locate your private key file (pritesh_keypair.pem). The wizard automatically detects the key you used to launch the instance.
Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
$ chmod 400 pritesh_keypair.pem
```

Connect to your instance using its Public DNS: ec2-100-25-191-138.compute-1.amazonaws.com

Example:

```
$ ssh -i "pritesh_keypair.pem" ubuntu@ec2-100-25-191-138.compute-1.amazonaws.com
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.
If you need any assistance connecting to your instance, please see our connection documentation.

Reference: https://www.youtube.com/watch?v=hHZoV3LBIwE&t=5121s

****
********* Installing Java *********
****

https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-18-04

```
$ java -version
```

Perform a system update

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

Installing JAVA

```
$ sudo apt install openjdk-17-jdk -y
$ java -version
```

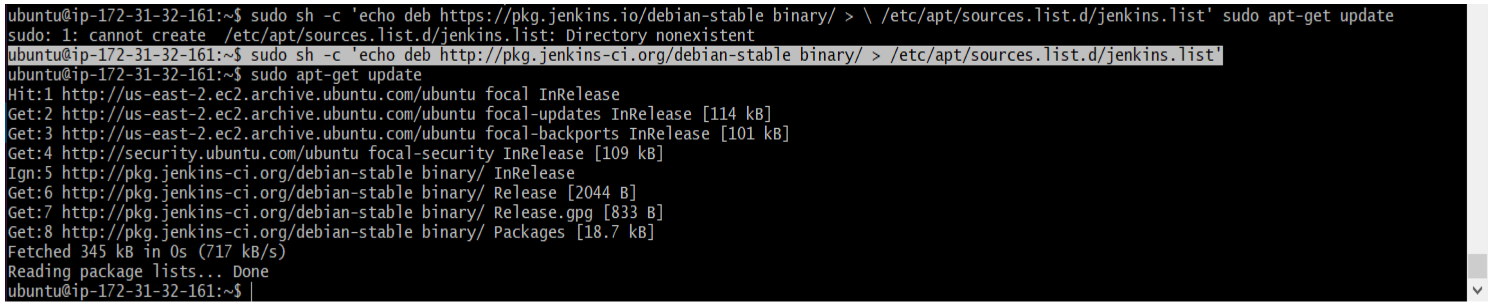
****

******** Installing Jenkins ********

****

Downloading and Installing Jenkins (**Research if these URL not working**)

```
$ sudo wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \
/etc/apt/sources.list.d/jenkins.list'
```



```
ubuntu@ip-172-31-32-161:~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \ /etc/apt/sources.list.d/jenkins.list' sudo apt-get update
sudo: 1: cannot create  /etc/apt/sources.list.d/jenkins.list: Directory nonexistent
ubuntu@ip-172-31-32-161:~$ sudo sh -c 'echo deb http://pkg.jenkins-ci.org/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
ubuntu@ip-172-31-32-161:~$ sudo apt-get update
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [109 kB]
Ign:5 http://pkg.jenkins-ci.org/debian-stable binary/ InRelease
Get:6 http://pkg.jenkins-ci.org/debian-stable binary/ Release [2044 B]
Get:7 http://pkg.jenkins-ci.org/debian-stable binary/ Release.gpg [833 B]
Get:8 http://pkg.jenkins-ci.org/debian-stable binary/ Packages [18.7 kB]
Fetched 345 kB in 0s (717 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-32-161:~$
```

```
$ sudo apt-get update
$ sudo apt-get install jenkins -y
```

Start Jenkins Services

```
$ sudo systemctl start jenkins
$ sudo systemctl status jenkins
```

Open web browser and enter **http://<ip address>:8080/**
Install on browser and fetch initial password from below given location

```
$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

****

********* Installing Docker *********

****

https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04

Perform a system update

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

Install Docker

```
$ sudo apt install docker.io -y
```

Add Jenkins user to docker

```
$ sudo usermod -aG docker jenkins
```

Restart Jenkins

```
$ sudo systemctl restart jenkins
```

Check that it's running:

```
$ sudo systemctl status docker
```

Working with Docker Images

```
$ sudo docker --version
$ sudo docker run hello-world
```

****
********* Installing Git *********
****

https://www.digitalocean.com/community/tutorials/how-to-install-git-on-ubuntu-20-04

Perform a system update

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

Install Git

```
$ sudo apt install git
```

Version Check

```
$ sudo git –version or git --version
```

Git user setup

```
$ git config --global user.name "Your Name"
$ git config --global user.email "youremail@domain.com"
```

Verify Git user setup

```
$ git config --list
```

Create Git repository name studentID_**COMP_3104_assignment2** and commit hello.txt file with your name, student ID and "Git repository created from AWS EC2 instance". *Don't forget to* ***submit GitHub repo link***

Refer previous weeks exercise to work with GitHub repositories using CLI commands

****
**** Installing Postgres ******
****

https://www.fosstechnix.com/how-to-install-sonarqube-on-ubuntu-20-04/
https://www.vultr.com/docs/how-to-install-sonarqube-on-ubuntu-16-04

Perform a system update

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

Install and configure PostgreSQL

Install the PostgreSQL repository (**Research if these URL not working**)

```
$ sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" >>
/etc/apt/sources.list.d/pgdg.list'
$ sudo wget -q https://www.postgresql.org/media/keys/ACCC4CF8.asc -O - | sudo apt-key add -
```

Install the PostgreSQL database server by running:

```
$ sudo apt-get -y install postgresql postgresql-contrib
```

Start PostgreSQL server and enable it to start automatically at boot time by running:

```
$ sudo systemctl start postgresql
$ sudo systemctl enable postgresql
```

Change the password for the default PostgreSQL user

```
$ sudo passwd postgres
```

Set password of postgres as **gbcdevops**

Switch to the postgres user

```
$ su - postgres
```

Create a new user by typing:

```
$ createuser sonar
```

Switch to the PostgreSQL shell

```
$ psql
```

Set a password for the newly created user for SonarQube database

```
$ ALTER USER sonar WITH ENCRYPTED password 'DevOps@gbc.ca';
```

Create a new database for PostgreSQL database by running:

```
$ CREATE DATABASE sonar OWNER sonar;
```

Exit from the psql shell:

```
\q
```

Switch back to the sudo user by running the **exit** command

****
****** Installing SonarQube ********
****

Download and configure SonarQube

Download the SonarQube installer files archive. (**Research if these URL not working**)

```
$ sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-6.4.zip
```

You can always look for the link to the latest version of the application on the SonarQube download page

Install unzip by running:

```
$ sudo apt-get -y install unzip
```

Unzip the archive using the following command

```
$ sudo unzip sonarqube-6.4.zip -d /opt
```

Rename the directory:

```
$ sudo mv /opt/sonarqube-6.4 /opt/sonarqube
```

Open the SonarQube configuration file using your favorite text editor

```
$ sudo nano /opt/sonarqube/conf/sonar.properties
```

Find the following lines

```
#sonar.jdbc.username=
#sonar.jdbc.password=
```

Uncomment and provide the PostgreSQL username and password of the database that we have created earlier. It should look like:

```
sonar.jdbc.username=sonar
sonar.jdbc.password=DevOps@gbc.ca
```

Next, find:

```
#sonar.jdbc.url=jdbc:postgresql://localhost/sonar
```

Uncomment the line, save the file and exit from the editor.

Configure Systemd service

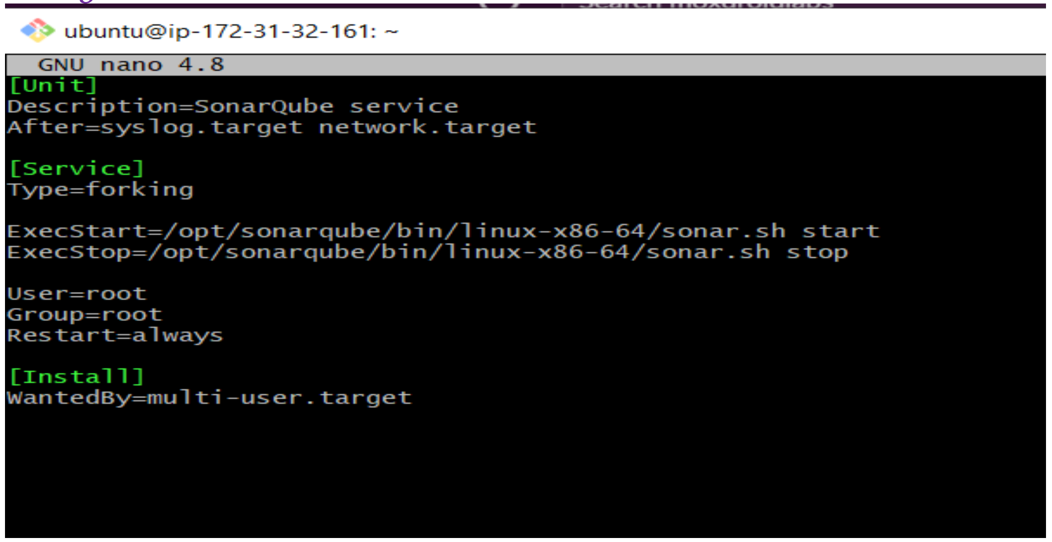
SonarQube can be started directly using the startup script provided in the installer package. As a matter of convenience, you should setup a Systemd unit file for SonarQube.

```
$ sudo nano /etc/systemd/system/sonar.service
```

Populate the file with:

```
[Unit]
Description=SonarQube service
After=syslog.target network.target

[Service]
Type=forking

ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop

User=root
Group=root
Restart=always

[Install]
```

WantedBy=multi-user.target



```
  GNU nano 4.8
[Unit]
Description=SonarQube service
After=syslog.target network.target

[Service]
Type=forking

ExecStart=/opt/sonarqube/bin/linux-x86-64/sonar.sh start
ExecStop=/opt/sonarqube/bin/linux-x86-64/sonar.sh stop

User=root
Group=root
Restart=always

[Install]
WantedBy=multi-user.target
```

Start the application by running:

```
$ sudo systemctl start sonar
```

Enable the SonarQube service to automatically start at boot time.

```
$ sudo systemctl enable sonar
```

To check if the service is running, run:

```
$ sudo systemctl status sonar
```

SonarQube is installed on your server, access the dashboard at the following address

http://<ip address>:9000

Log in using the initial administrator account, **admin** and **admin**. You can now use SonarQube to continuously analyze the code you have written.

Submission

- Upload Doc file containing URL of your SonarQube and Jenkins server which you installed on EC2 Ubuntu instance.
- Also put all your screenshots of commands on terminal as a proof of your work with proper label.