Step by step instructions:

1. First, I installed vagrant and VirtualBox. This step is straight forward. Just click the exe and hit next.
2. The second step is to run vagrant init, and modify the vagrant file. I have all the file mentioned below at https://github.com/Dedsec-Xu/devops/tree/main/final

```
Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/focal64"
  config.vm.hostname = "sonarqube.box"
  config.vm.provision "docker"

  config.vm.provider "virtualbox" do |v|
    v.memory = 7792
    v.cpus = 2
  end

  config.vm.provision :docker_compose, yml: "/vagrant/docker-compose.yml", run: "always"

  config.vm.provision "shell" do |shell|
    shell.path = "mem.sh"
  end

  config.vm.network "forwarded_port", guest: 9000, host: 9000
  config.vm.network "forwarded_port", guest: 9092, host: 9092

  config.vm.network "forwarded_port", guest: 8080, host: 8080
  config.vm.network "forwarded_port", guest: 50000, host: 50000
end
```

I also added provisioning file mem.sh:

```
sysctl -w vm.max_map_count=262144
echo "vm.max_map_count=262144" >> /etc/sysctl.conf
sysctl -w fs.file-max=131072
echo "fs.file-max=131072" >> /etc/sysctl.conf
ulimit -n 65536
ulimit -u 4096

sudo apt-get update
sudo apt-get upgrade
sudo apt-get install wget unzip -y

sudo apt-get install openjdk-11-jdk -y
sudo apt-get install openjdk-11-jre -y


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

# sudo passwd postgres
# su - postgres
# createuser sonar
# psql
# ALTER USER sonar WITH ENCRYPTED password 'sonar';
# CREATE DATABASE sonarqube OWNER sonar;
# grant all privileges on DATABASE sonarqube to sonar;
# \q
# exit
```

This script installs postgresql for sonarqube. It will automatically run by vagrant because it is used in vagrant file.

There is also a docker-compose.yml that I wrote for Jenkins installation:

```
version: "2"

services:
  jenkins:
    image: 'jenkinsci/blueocean:latest'
```

```
    ports:
      - "8080:8080"
      - "50000:50000"
    volumes:
      - jenkins_home:/var/jenkins
      - /var/run/docker.sock:/var/run/docker.sock
      - /usr/bin/docker:/usr/bin/docker
    environment:
      - 'JENKINS_OPTS=--httpPort=8080 --httpsPort=8443'


volumes:
  jenkins_home:
```

3. The next step is simply run vagrant up. The vagrant will create a VM and run mem.sh, which installs postgresql. It also installs docker and docker-compose. And then it will download a Jenkins docker container from `jenkinsci/blueocean:latest`. Then Jenkins, blue ocean and postgresql are all set up. I can login to Jenkins with the password in the logs.
4. The next step is to set up Sonarqube. Firstly I will have to run the commented code in mem.sh manually. This will create a username and password for sonar to use sql.

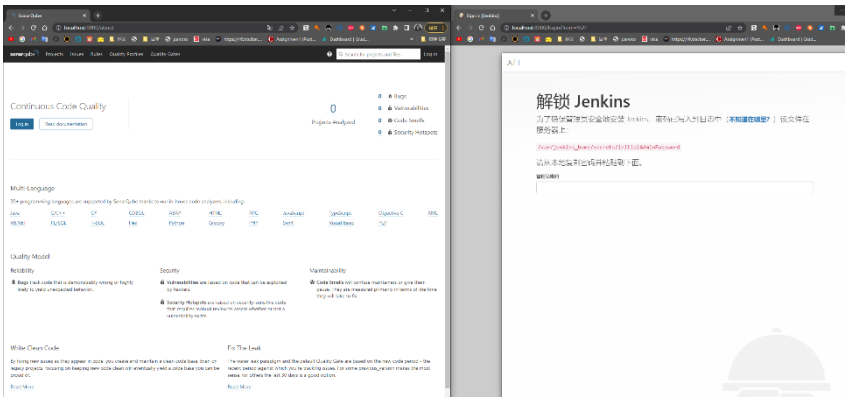
```
# sudo passwd postgres
# su - postgres
# createuser sonar
# psql
# ALTER USER sonar WITH ENCRYPTED password 'sonar';
# CREATE DATABASE sonarqube OWNER sonar;
# grant all privileges on DATABASE sonarqube to sonar;
# \q
# exit
```

Then I need to run sonar-setup.sh. which is also a script that I wrote. This will install sonarqube and set up all the config file as well.

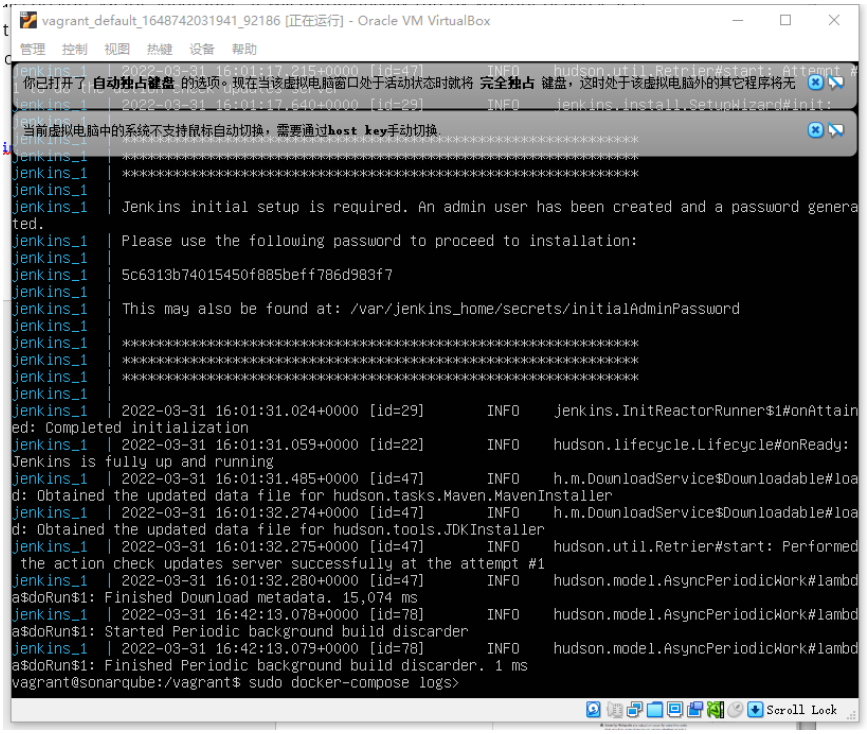
```
sudo wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-7.9.5.zip
sudo unzip sonarqube-7.9.5.zip -d /opt
sudo mv /opt/sonarqube-7.9.5 /opt/sonarqube

sudo groupadd sonar
sudo useradd -c "user to run SonarQube" -d /opt/sonarqube -g sonar sonar
sudo chown sonar:sonar /opt/sonarqube -R

cp /opt/sonarqube/conf/sonar.properties ./sonar.properties
echo 'sonar.jdbc.username=sonar' >> ./sonar.properties
echo 'sonar.jdbc.password=sonar' >> ./sonar.properties
echo 'sonar.jdbc.url=jdbc:postgresql://localhost:5432/sonarqube' >> ./sonar.properties

cp ./sonar.properties /opt/sonarqube/conf/sonar.properties

cp /opt/sonarqube/bin/linux-x86-64/sonar.sh ./sonar.sh
sed -i '50 i RUN_AS_USER=sonar' ./sonar.sh
cp ./sonar.sh /opt/sonarqube/bin/linux-x86-64/sonar.sh

# sudo su sonar
# cd /opt/sonarqube/bin/linux-x86-64/
# ./sonar.sh start
```

Lastly, I will run the commented code in sonar-setup.sh to start-up the sonarqube

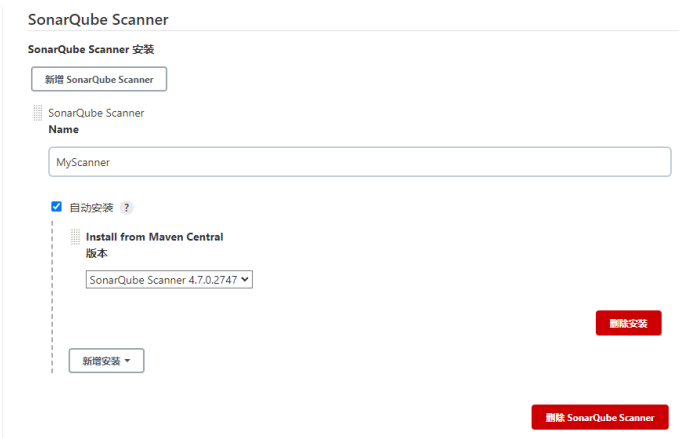
5. Then I can access Jenkins using localhost:8080. And access sonarqube using localhost:9000

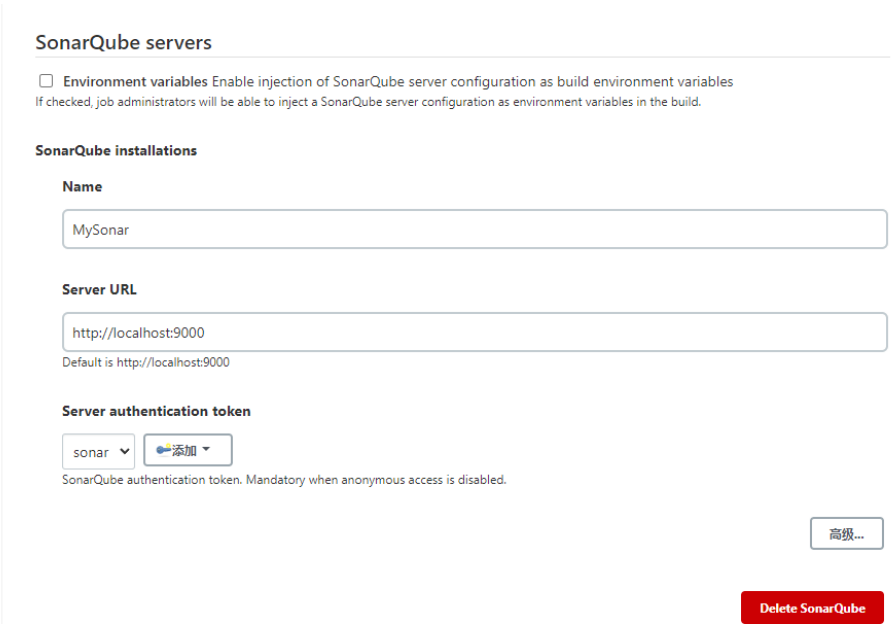


6. Then I login into Jenkins and installed sonarrunner. I login into Jenkins by running sudo docker-compose logs



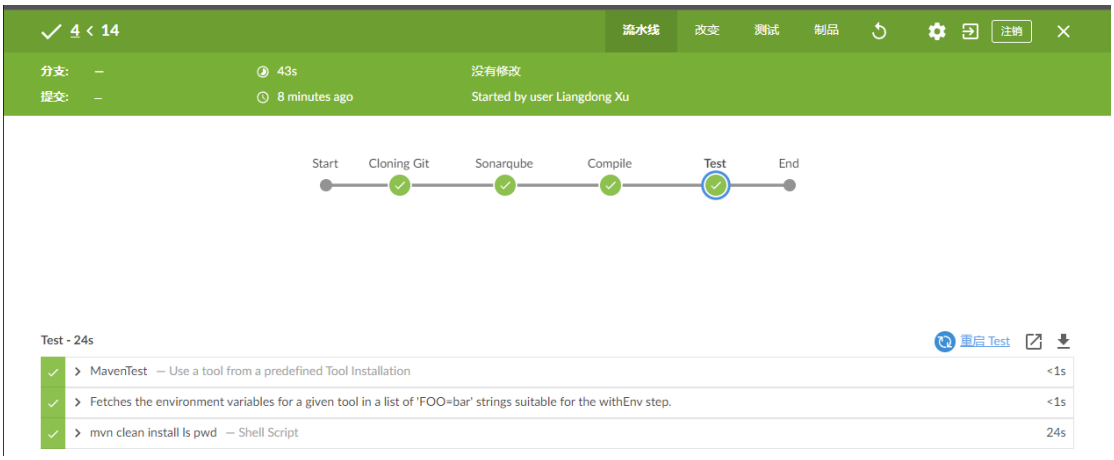
The password is in the logs.



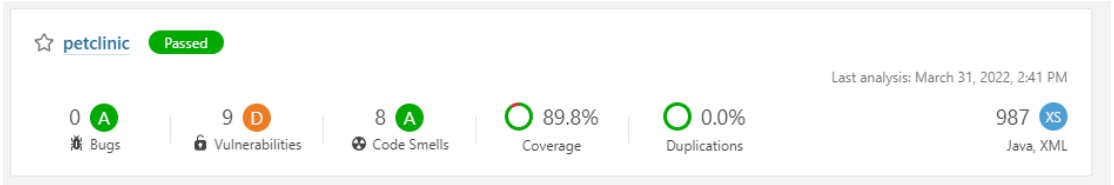


I also did some basic setup connecting sonarqube with Jenkins. It includes generating a token and use it in Jenkins.

7. Time to build the project.
   I forked the pet-clinic repo and added jenkins file. I then added a pipeline in jenkins and used pipeline script to add 4 steps of building the file. Then I built it and visualized it using blue ocean.



   Sonarqube analysis is also a step in the pipeline and here are the results. I simply used sh 'mvn sonar:sonar' in the pipeline file.



   Lastly, Because 8080 port is already in use. I used docker cp 06c03af0e8e2:/var/jenkins_home/workspace/4/target/spring-clinic.jar /vagrant/1.jar

to copy the built jar into my local machine. And then I ran it using java -jar -Dserver.port=7999 ./1.jar

,

Finally, I opened localhost:7999 on chrome and I finally see this page.