

**DevOps Automation done with GitHub, Jenkins and Kubernetes**

[**Git**](https://git-scm.com/)  *is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.*

[***Jenkins***](https://www.jenkins.io/)*is a free and open source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat.*

[***Kubernetes***](https://kubernetes.io/)*(commonly stylized as****k8s****) is an open-source container-orchestration system for automating computer application deployment, scaling, and management. It was originally designed by Google and is now maintained by the Cloud Native Computing Foundation. It aims to provide a “platform for automating deployment, scaling, and operations of application containers across clusters of hosts”.*

*I have complete a DevOps Task in which I have deployed the Website on Apache Web server or codes written in PHP on PHP server using Jenkins over the Kubernetes container management platform. For deploying these kind of server I have used the Docker images and also create my own Docker image that will provide complete environment in single OS.*

*In the Docker image which I have created already configured the admin and installed some plugins that will need in performing the task. You get all resources for this task by visiting the links provided below.*

***Steps to be followed for the task:***

***Step 1:****Create container image that’s has Jenkins installed using Docker file Or You can use the Jenkins Server on RHEL 8/7*

***Step 2:****When we launch this image, it should automatically starts Jenkins service in the container.*

***Step 3:****Create a job chain of job1, job2, job3 and job4 using build pipeline plugin in Jenkins*

***Step 4:****Job1 : Pull the GitHub repository automatically when some developers push repository to GitHub.*

***Step 5:****Contains multiple job steps*

*Job2 :*

***1.****By looking at the code or program file, Jenkins should automatically start the respective language interpreter installed image container to deploy code on top of Kubernetes ( eg. If code is of PHP, then Jenkins should start the container that has PHP already installed )*

***2.****Expose your pod so that testing team could perform the testing on the pod*

***3****. Make the data to remain persistent ( If server collects some data like logs, other user information )*

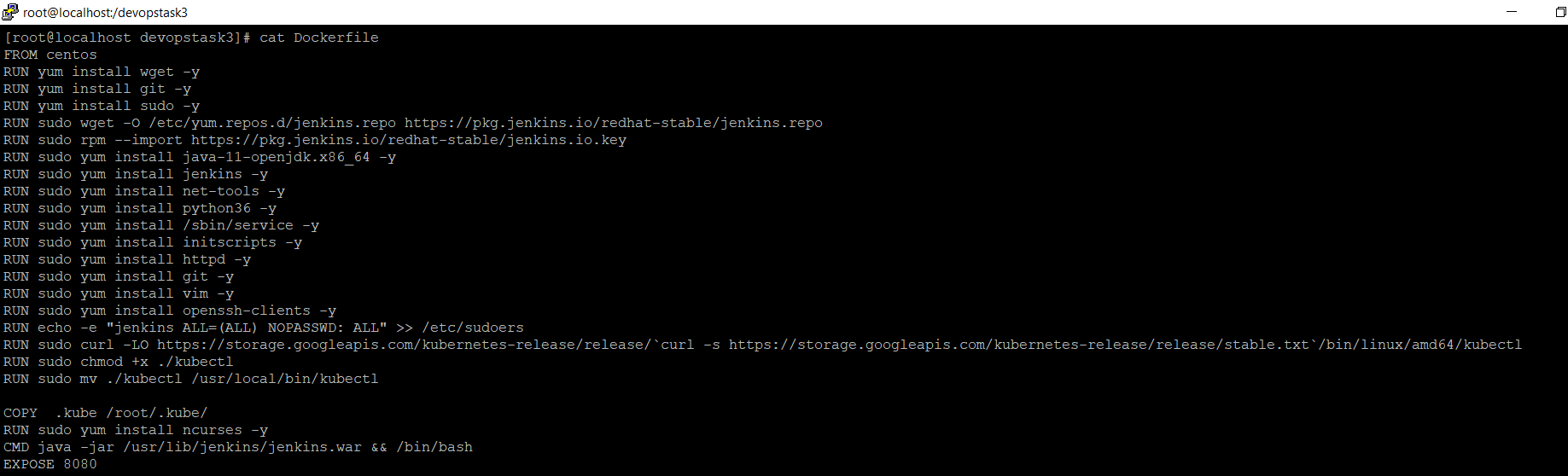
***Step 6:****Job3 : Test your app if it is working or not.*

***Step 7:****Job4 : if app is not working , then send email to developer with error messages and redeploy the application after code is being edited by the developer.*

***Enough theory done. Now its time to action!!!!***

*Step by step procedure and explanation:*

***Step 1:****Created one container image that’s has Jenkins and Kubernetes installed using Docker file.*



FROM centos

RUN yum install wget -y

RUN yum install git -y

RUN yum install sudo -y

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

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

RUN sudo yum install java-11-openjdk.x86\_64 -y

RUN sudo yum install jenkins -y

RUN sudo yum install net-tools -y

RUN sudo yum install python36 -y

RUN sudo yum install /sbin/service -y

RUN sudo yum install initscripts -y

RUN sudo yum install httpd -y

RUN sudo yum install git -y

RUN sudo yum install vim -y

RUN sudo yum install openssh-clients -y

RUN echo -e "jenkins ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

RUN sudo curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl

RUN sudo chmod +x ./kubectl

RUN sudo mv ./kubectl /usr/local/bin/kubectl

COPY .kube /root/.kube/

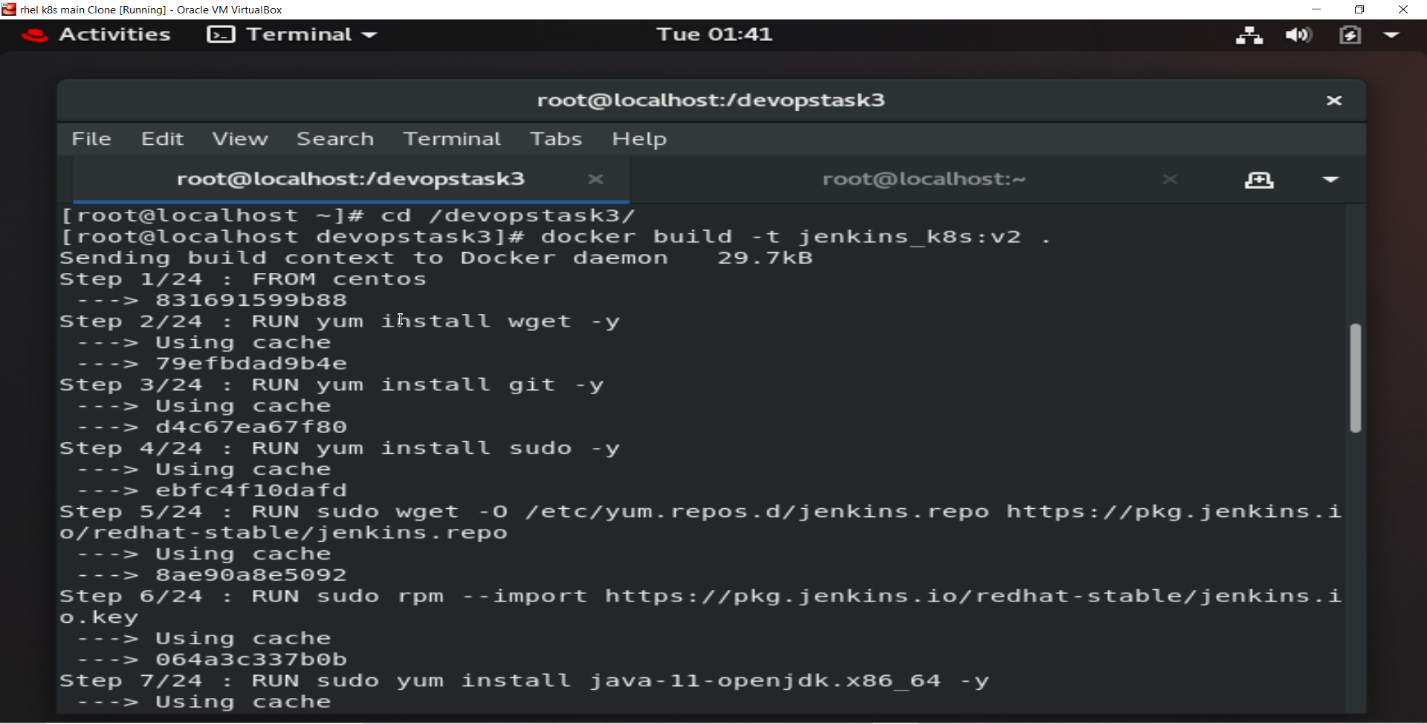
RUN sudo yum install ncurses -y

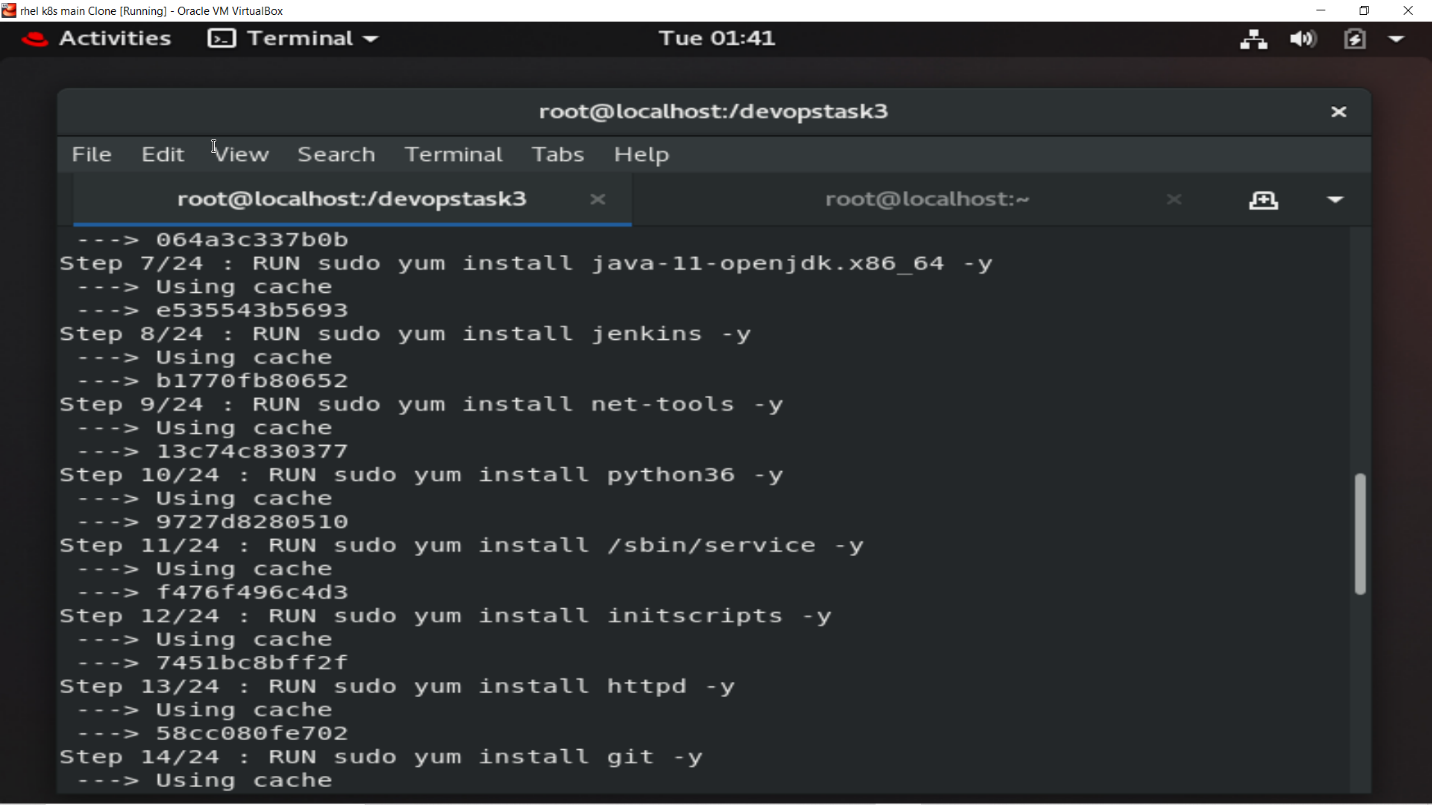
CMD java -jar /usr/lib/jenkins/jenkins.war && /bin/bash

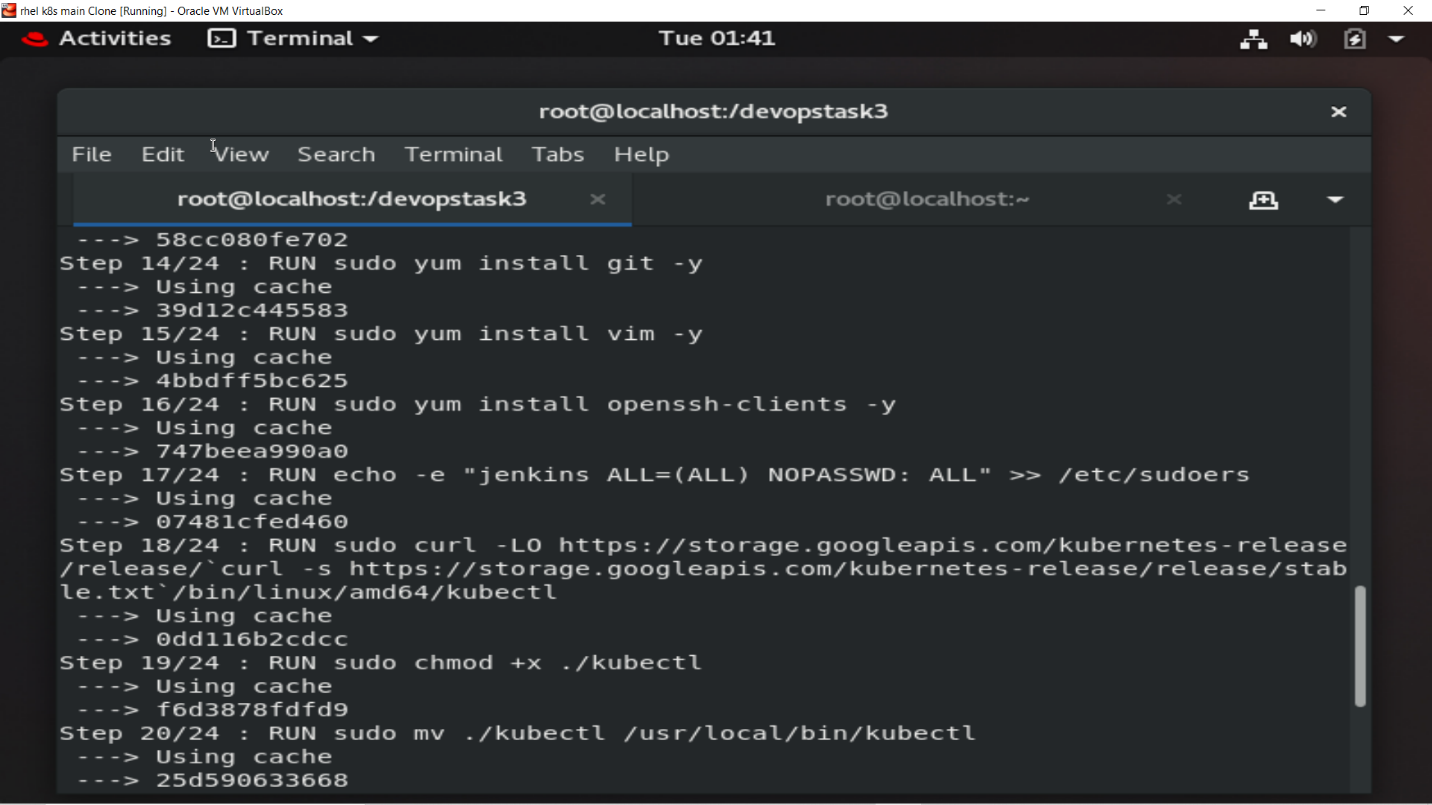
EXPOSE 8080

Command Creating for Docker image

docker build -t Jenkins\_k8s:v1 .







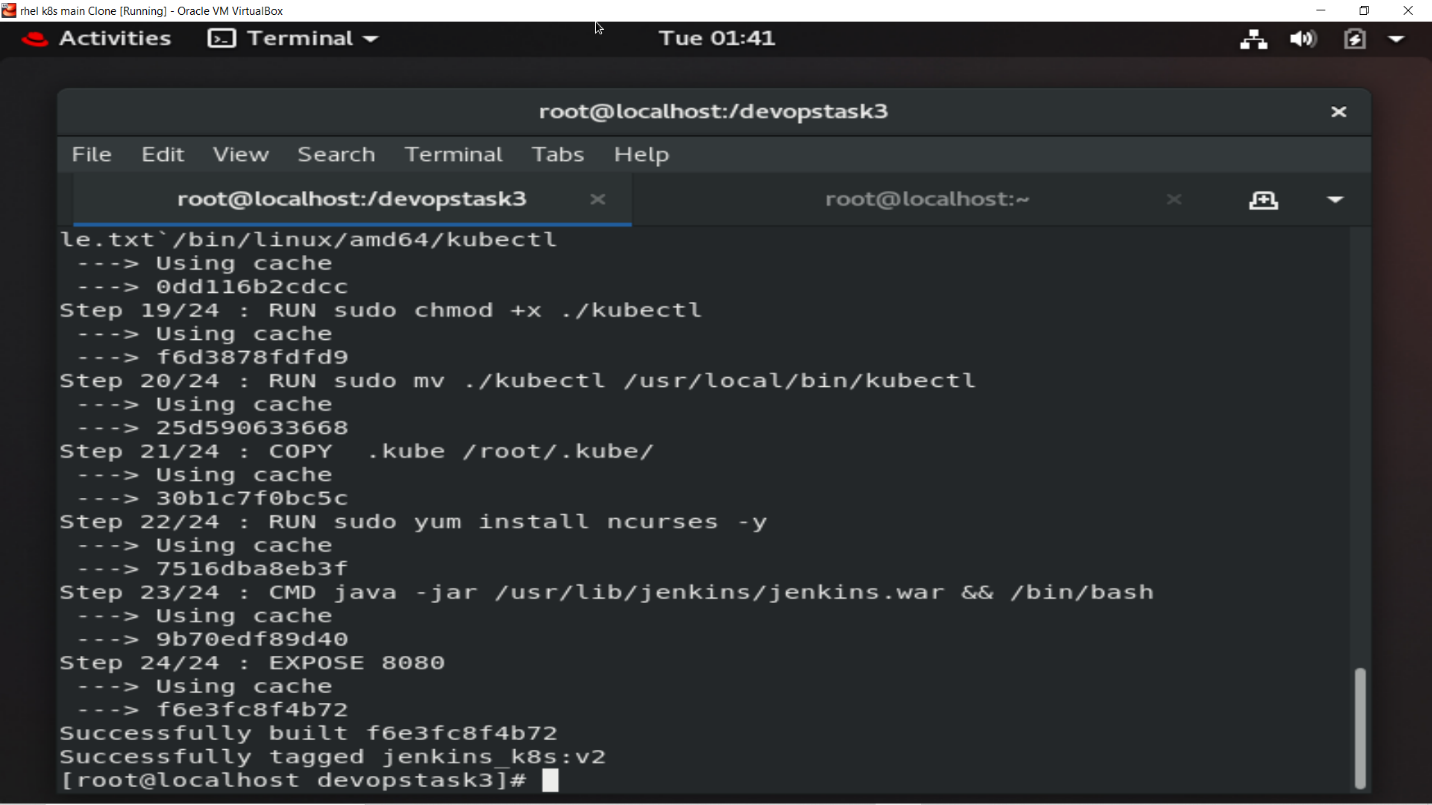
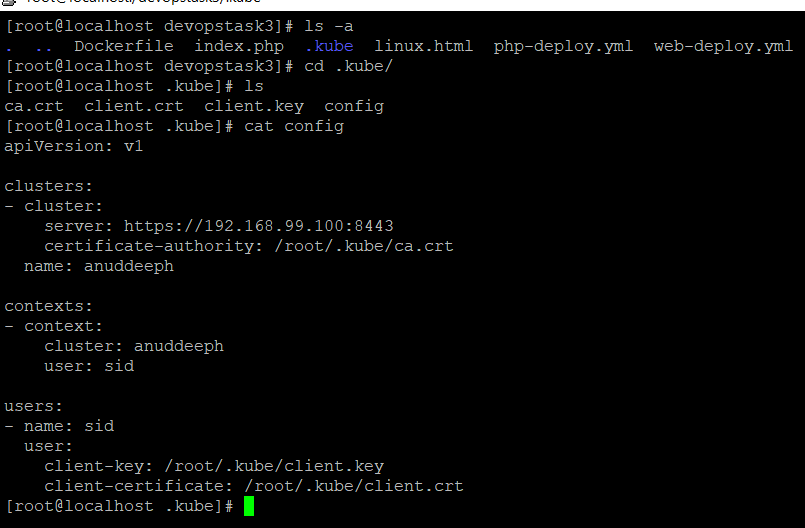


Image build successfully!



apiVersion: v1

clusters:

- cluster:

server: https://192.168.99.100:8443

certificate-authority: /root/.kube/ca.crt

name: anuddeeph

contexts:

- context:

cluster: anuddeeph

user: sid

users:

- name: sid

user:

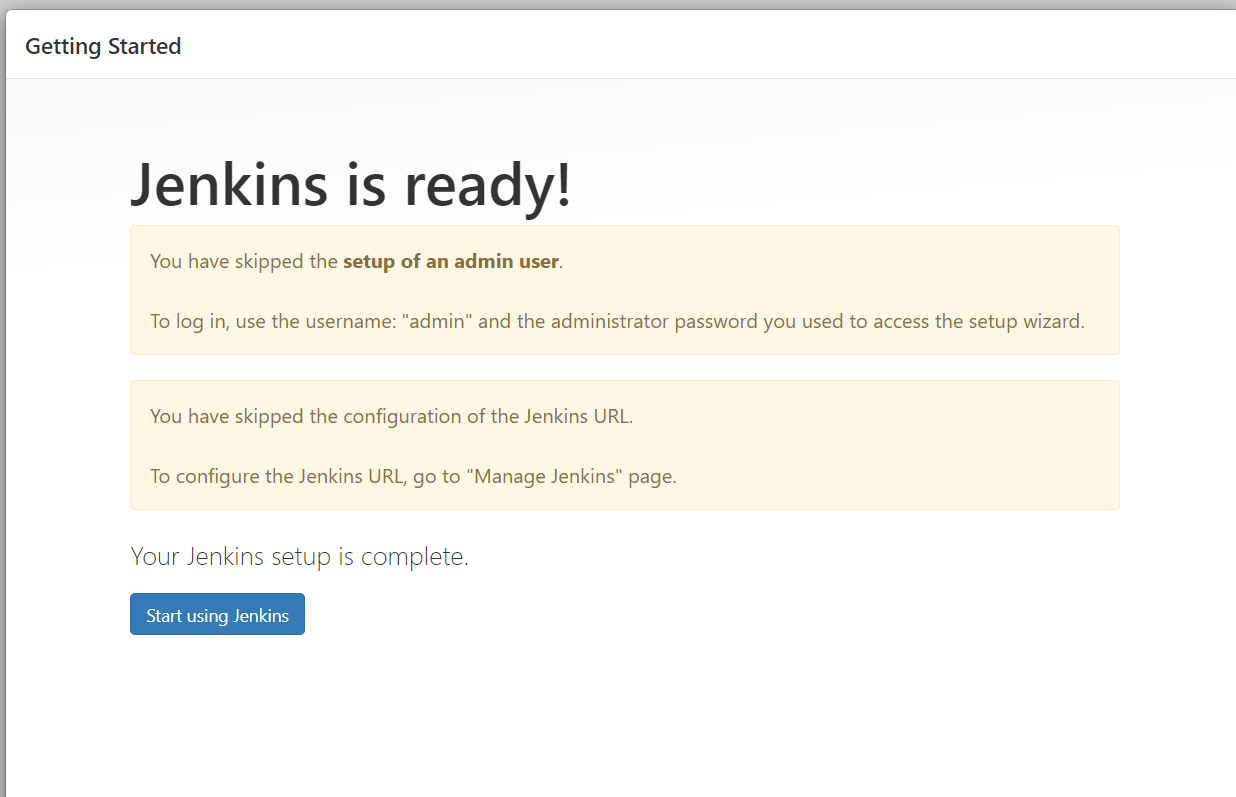
client-key: /root/.kube/client.key

client-certificate: /root/.kube/client.crt

### Step 2: When we launch this image, it should automatically starts Jenkins service in the container.

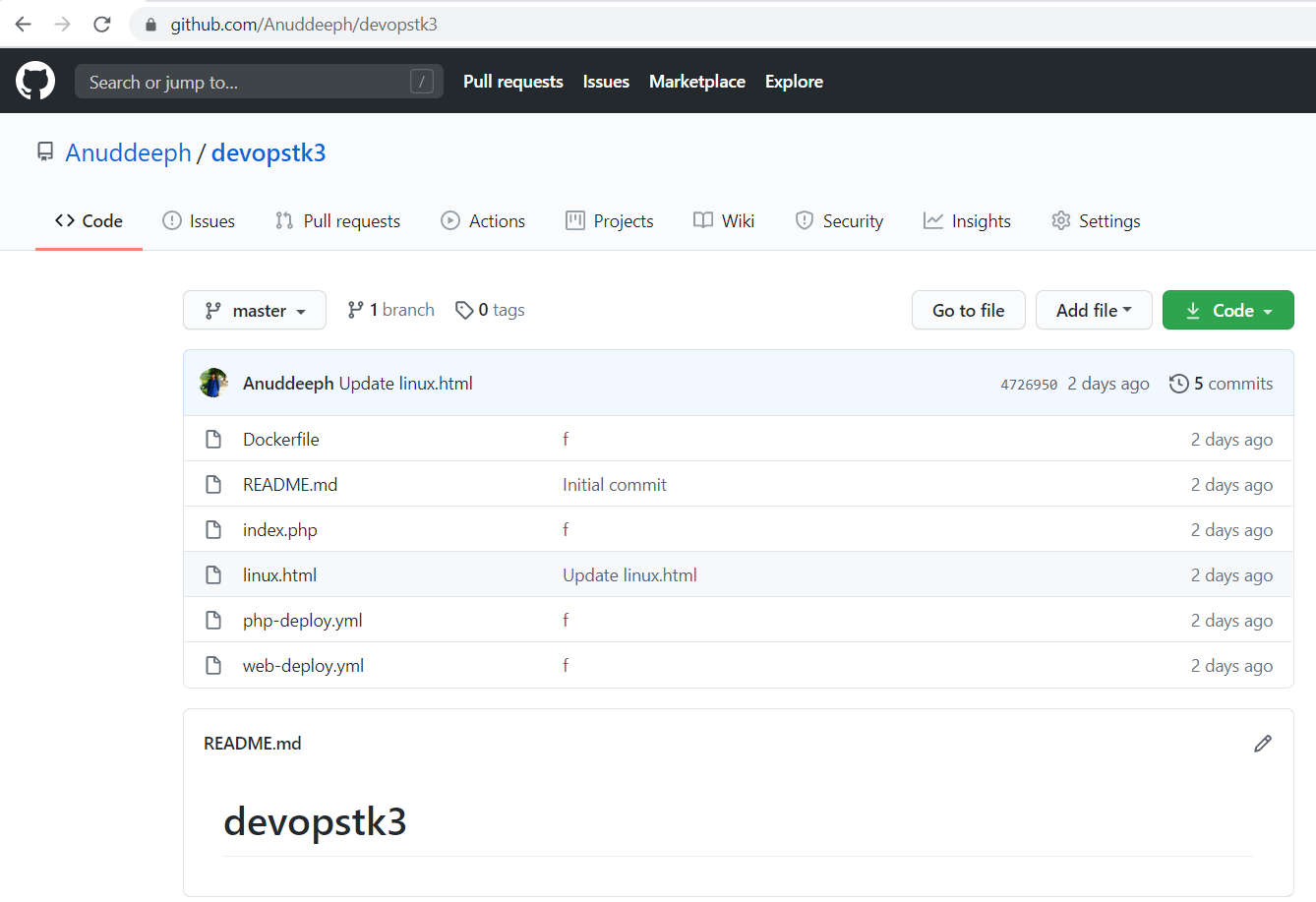
### Now run the docker container and expose it

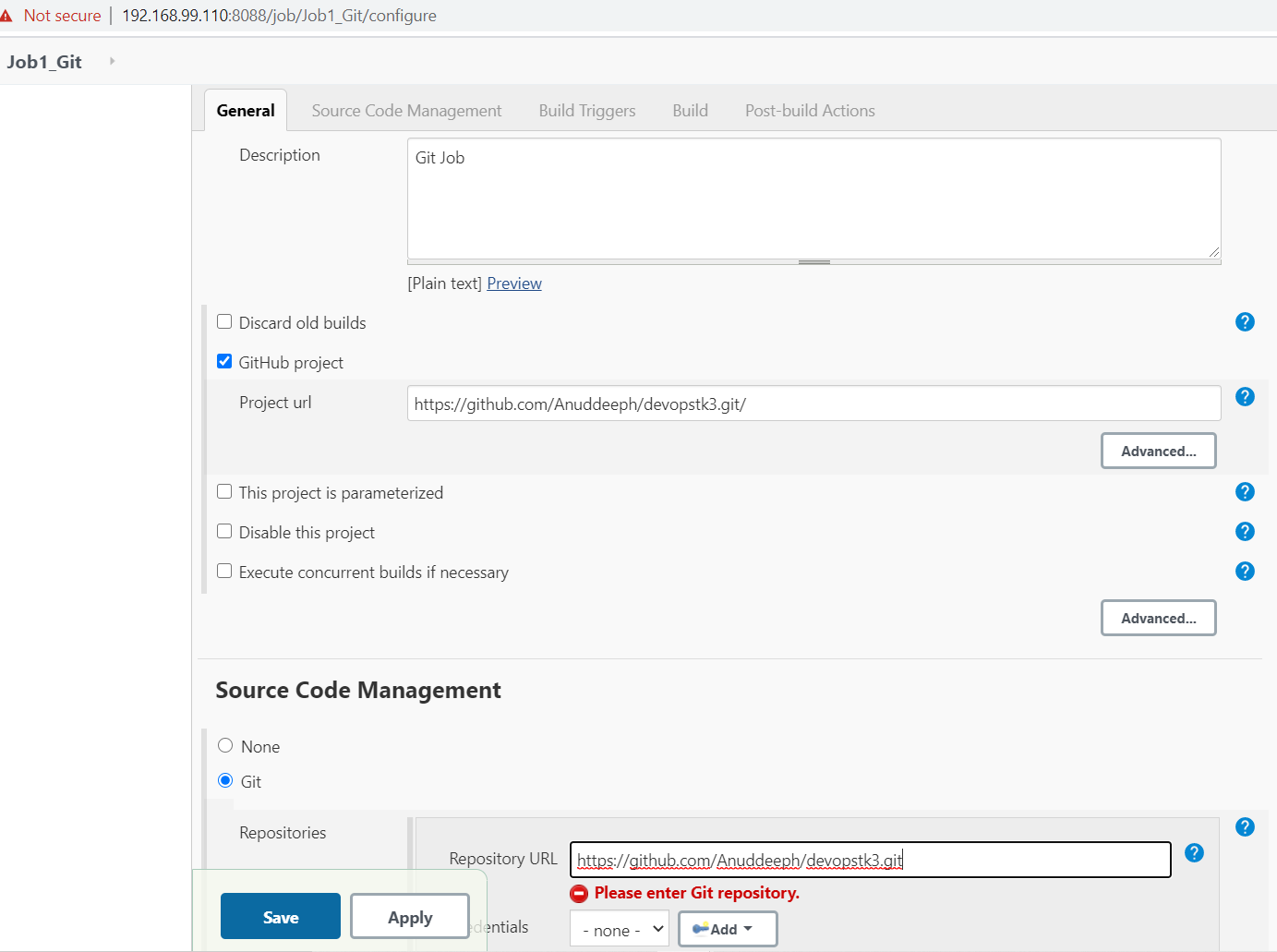
docker run -dit -p 8085:8080 --name jenk anuddeeph/jenkins\_k8s:v1

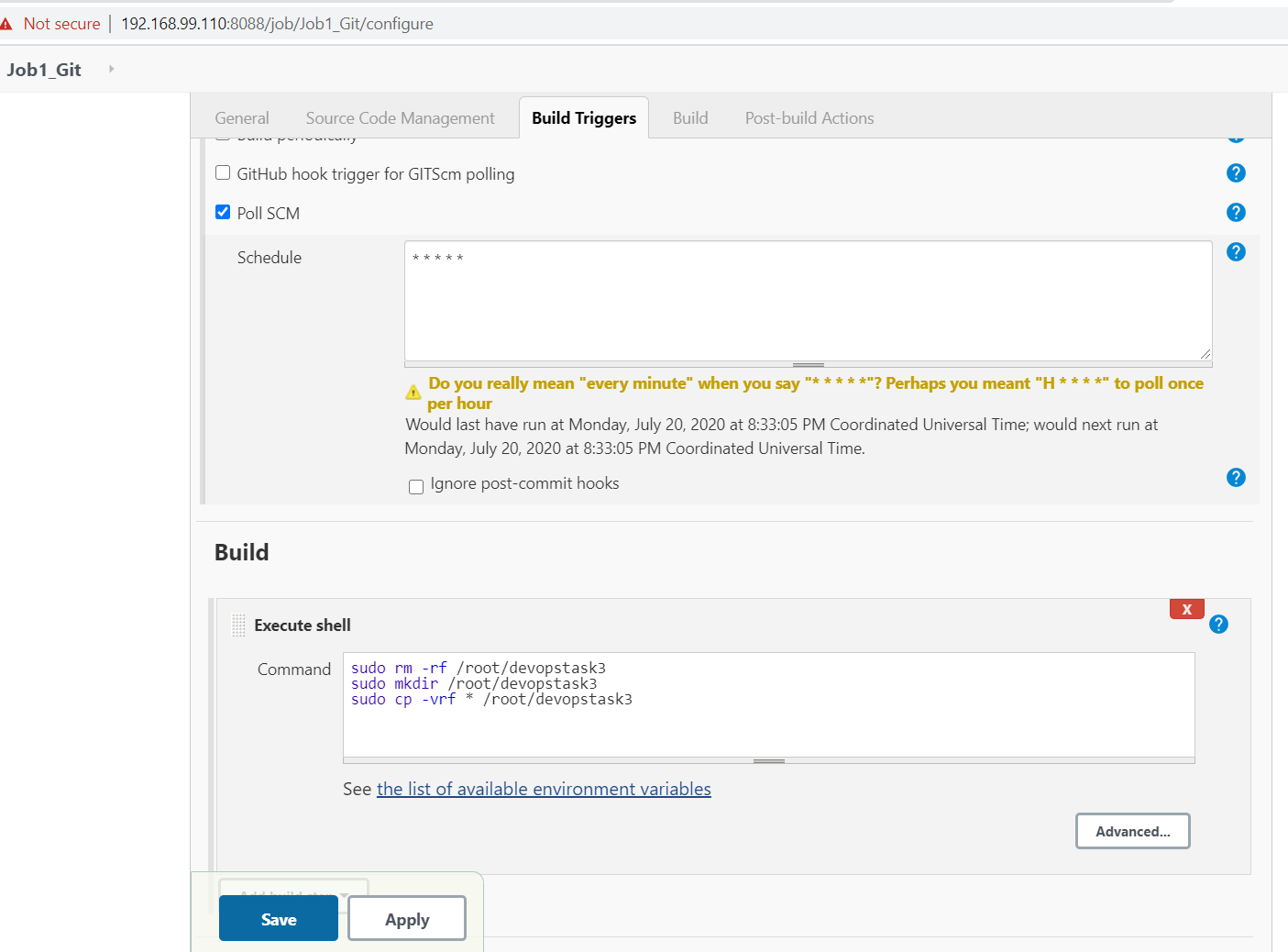


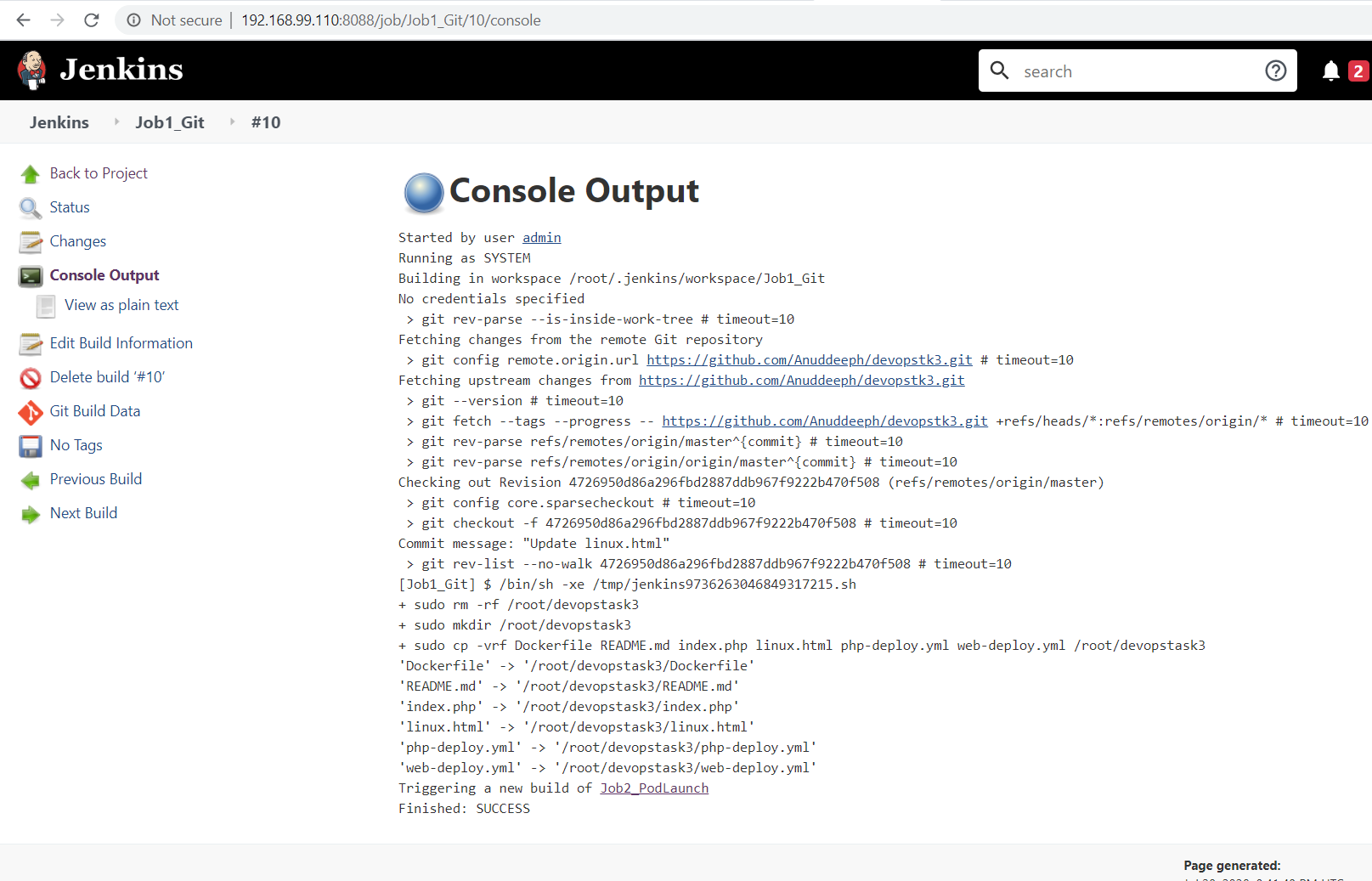
**Step 3:** This is a job chain of job1, job2, job3 and job4 using build pipeline plugin in Jenkins.

**Step 4:**  Job1 : Developer push codes to the GitHub repository





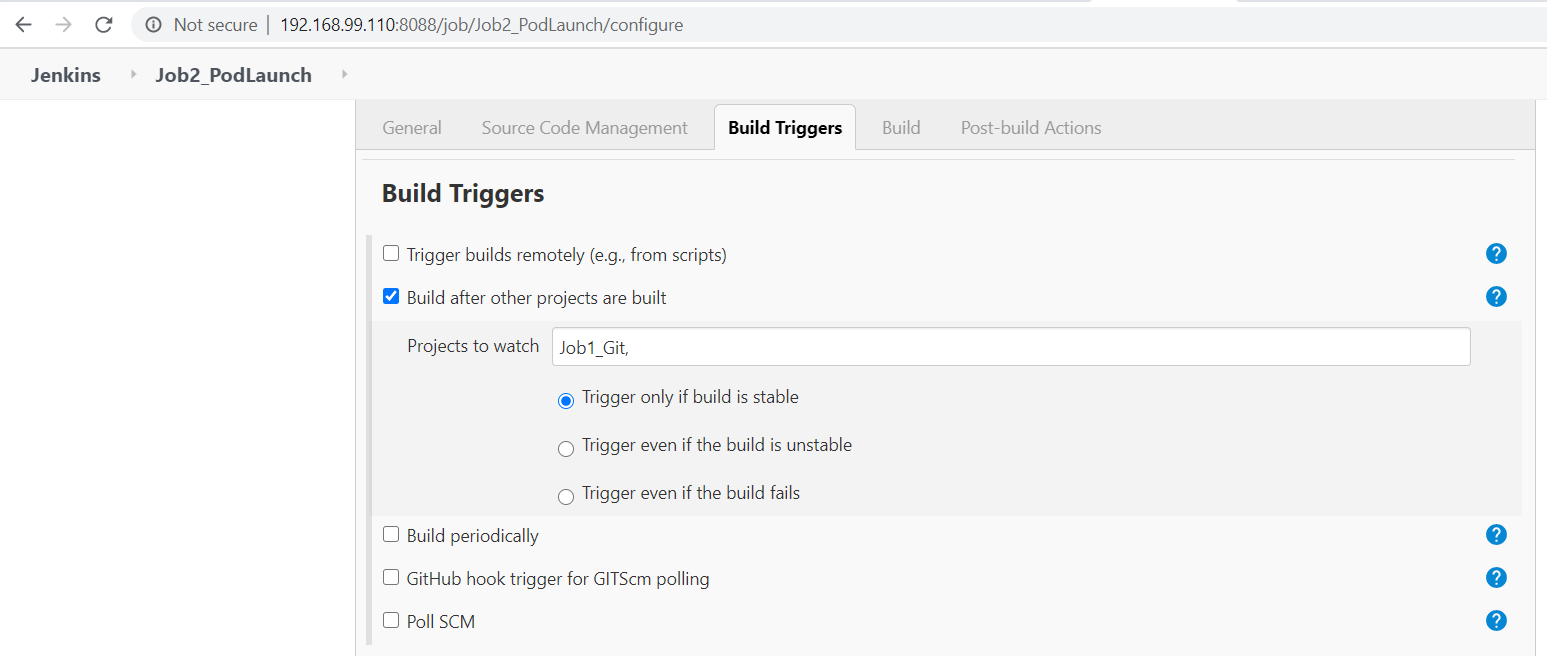


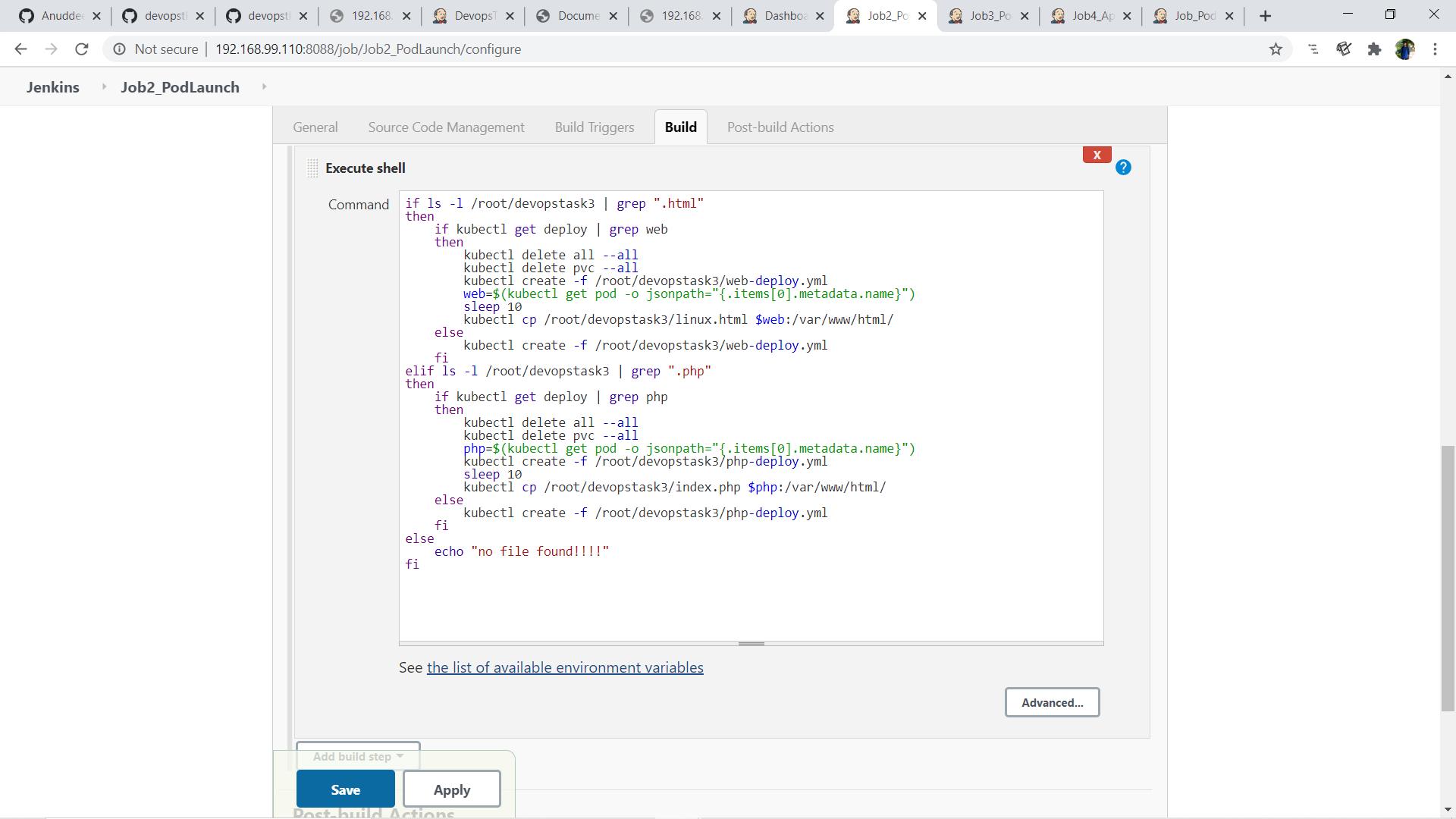


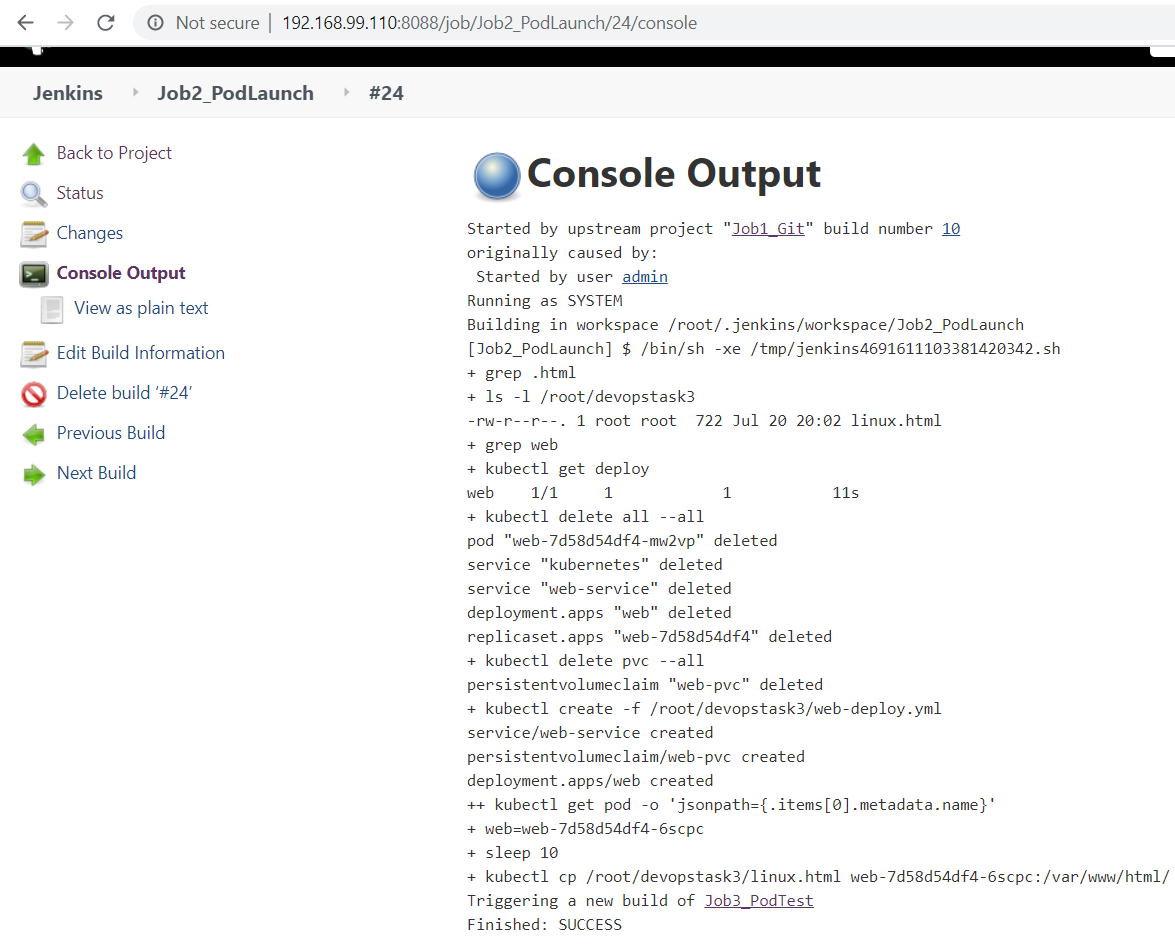
Output for the successful pull of GitHub repository

* Start Minikube using minikube start

**Step 5:**Job2 : Jenkins launch pod automatically by looking at the program file on top of Kubernetes.

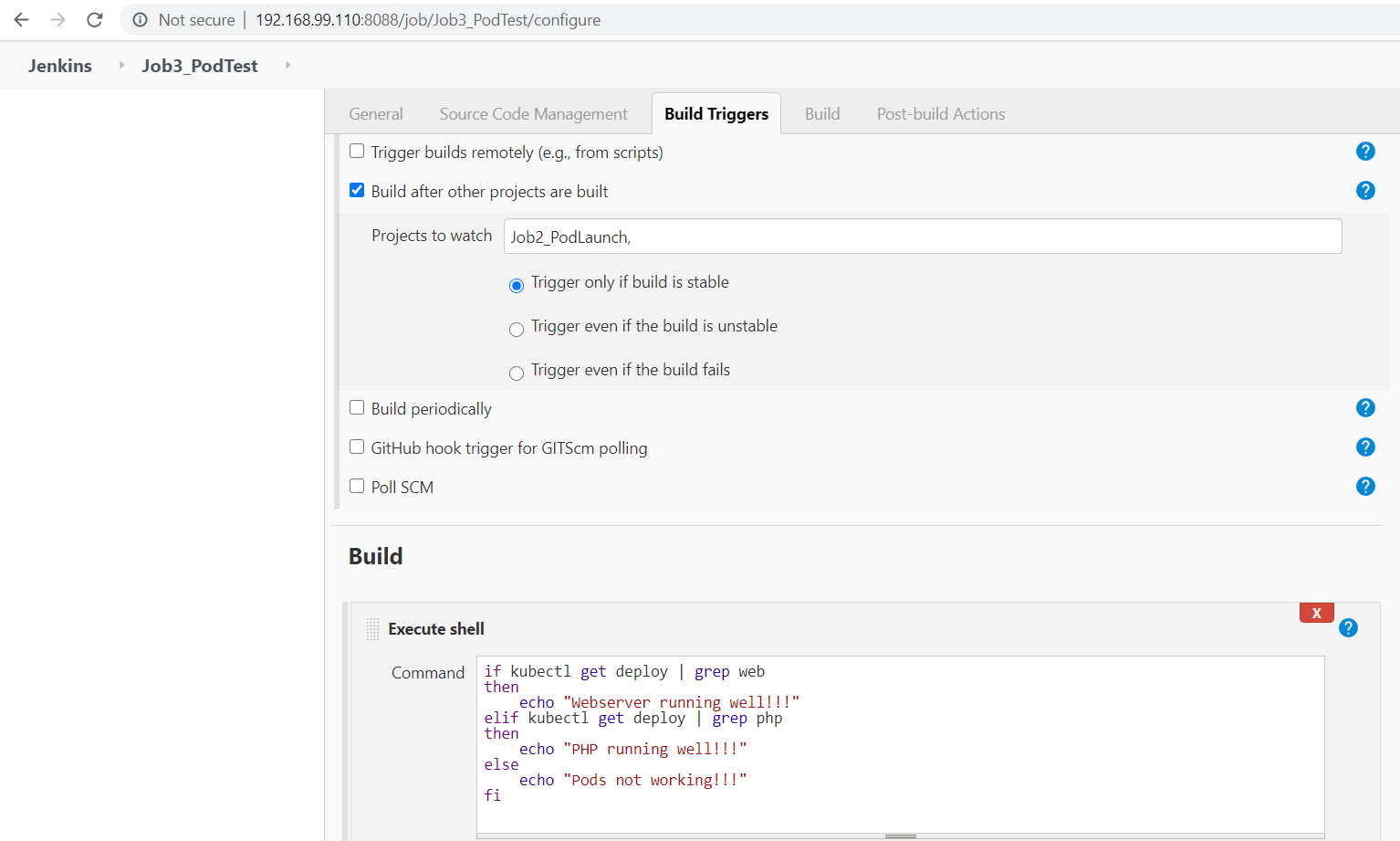


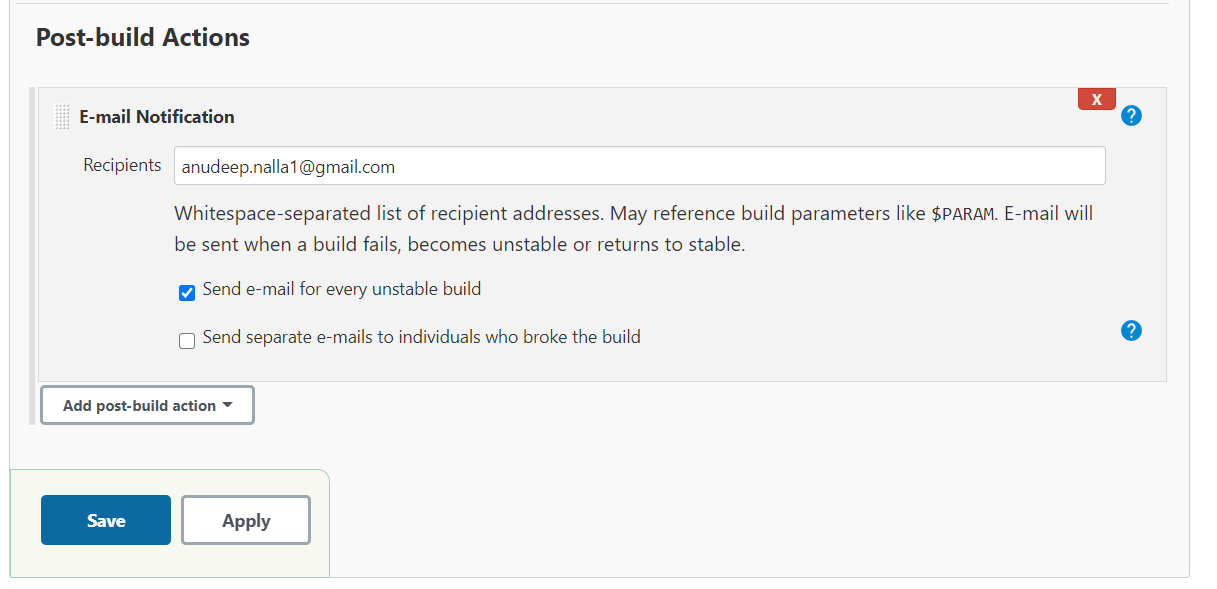


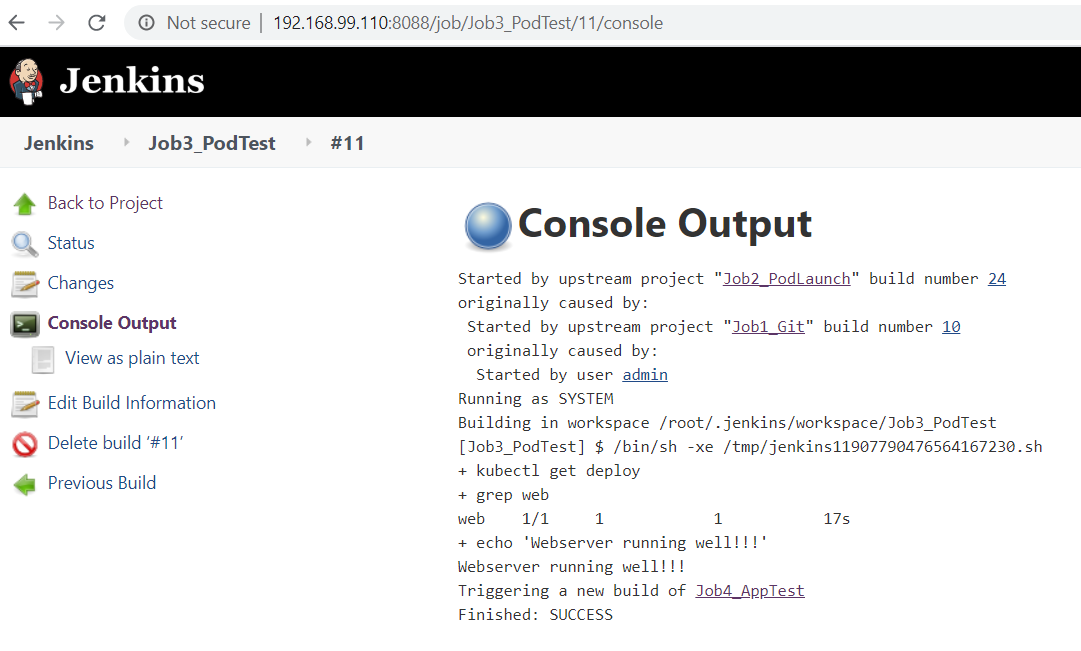


Successfully pod launch as per coding files. Here the pod having the persistent storage and exposed for outside world.

**Step 6:** Job3 : Testing whether pods working or not.

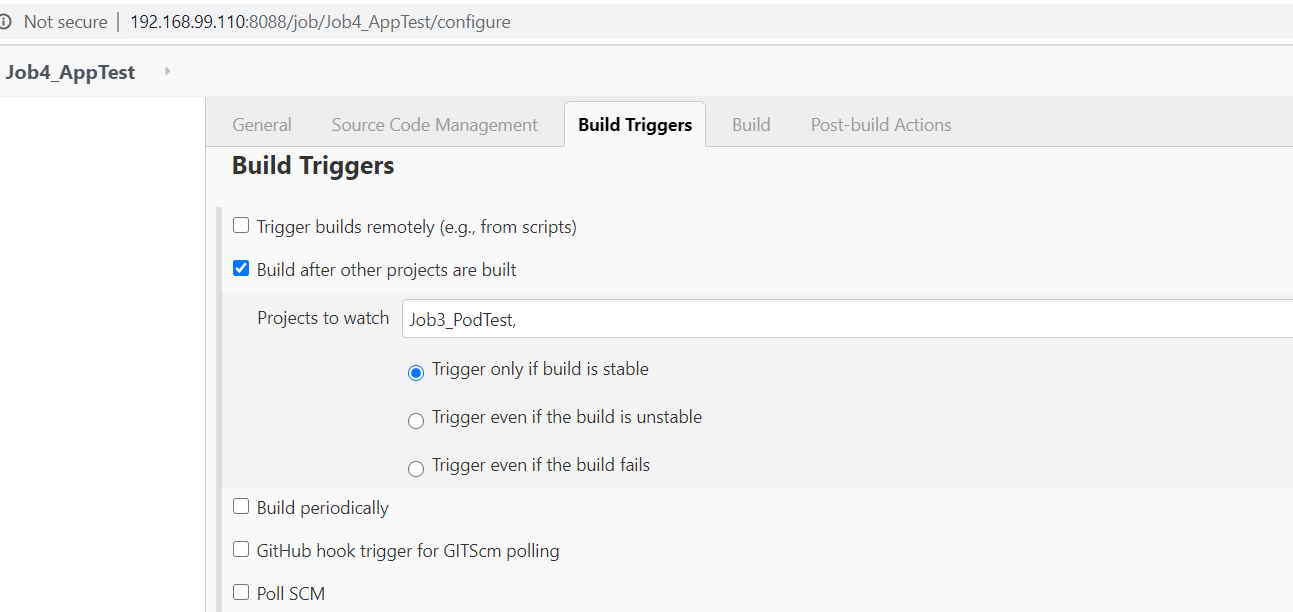


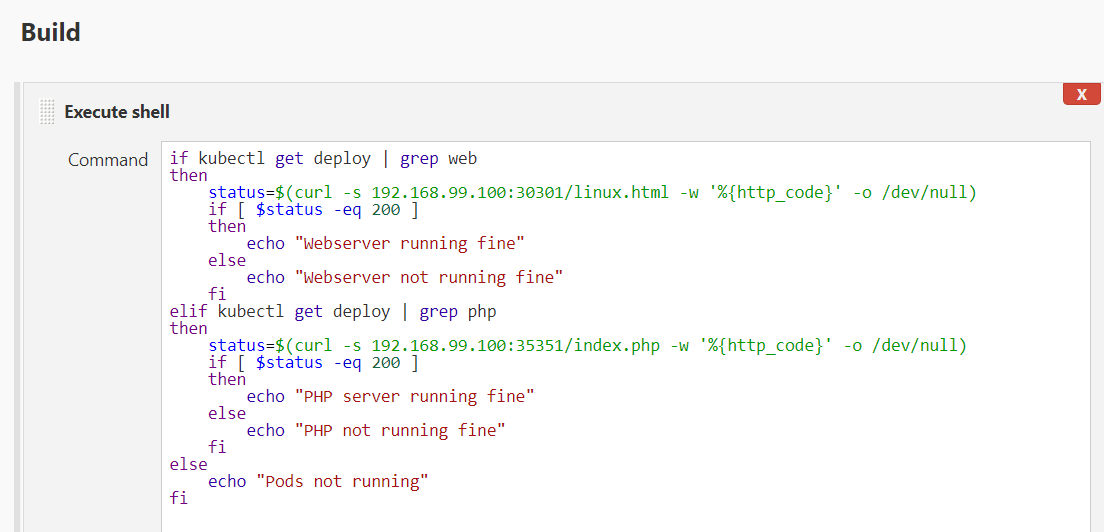


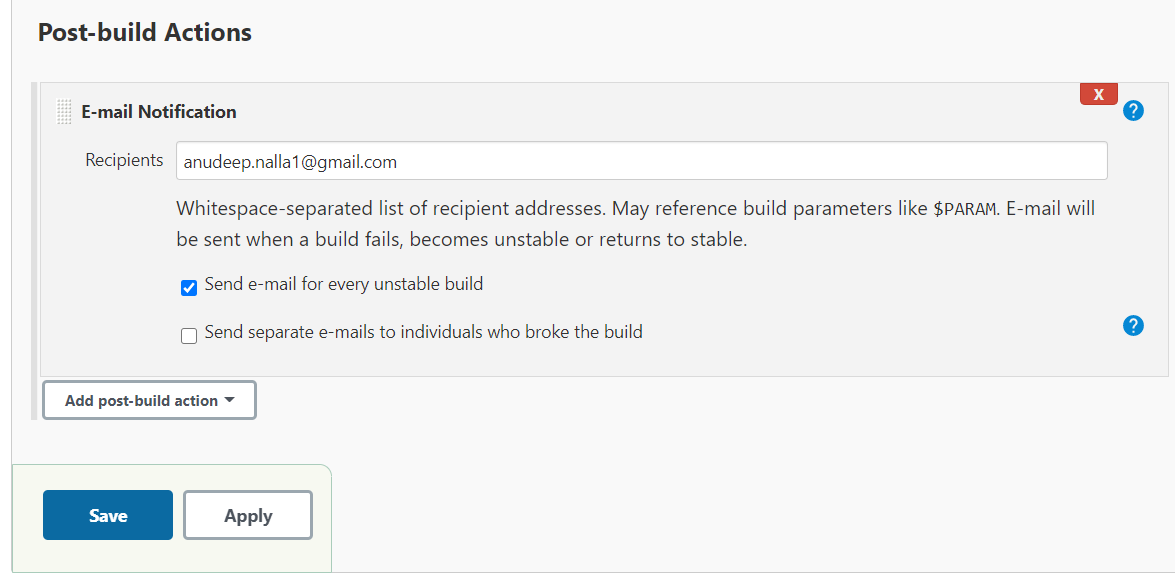


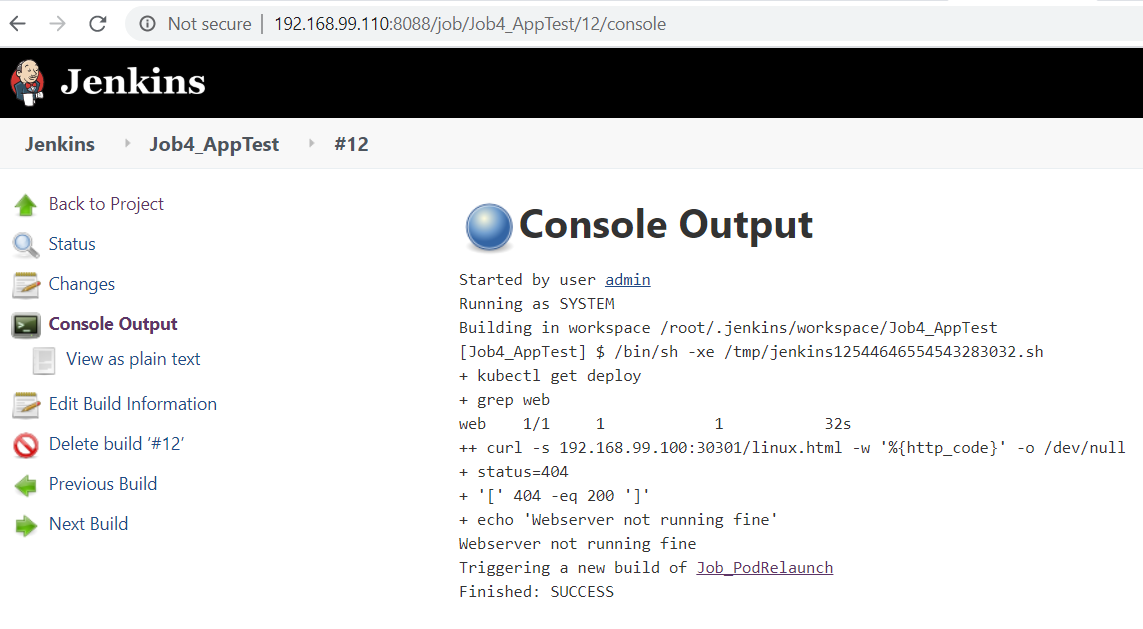
Output for the testing of pods

**Step 7:**Job4: for testing the web server or app running on the pod



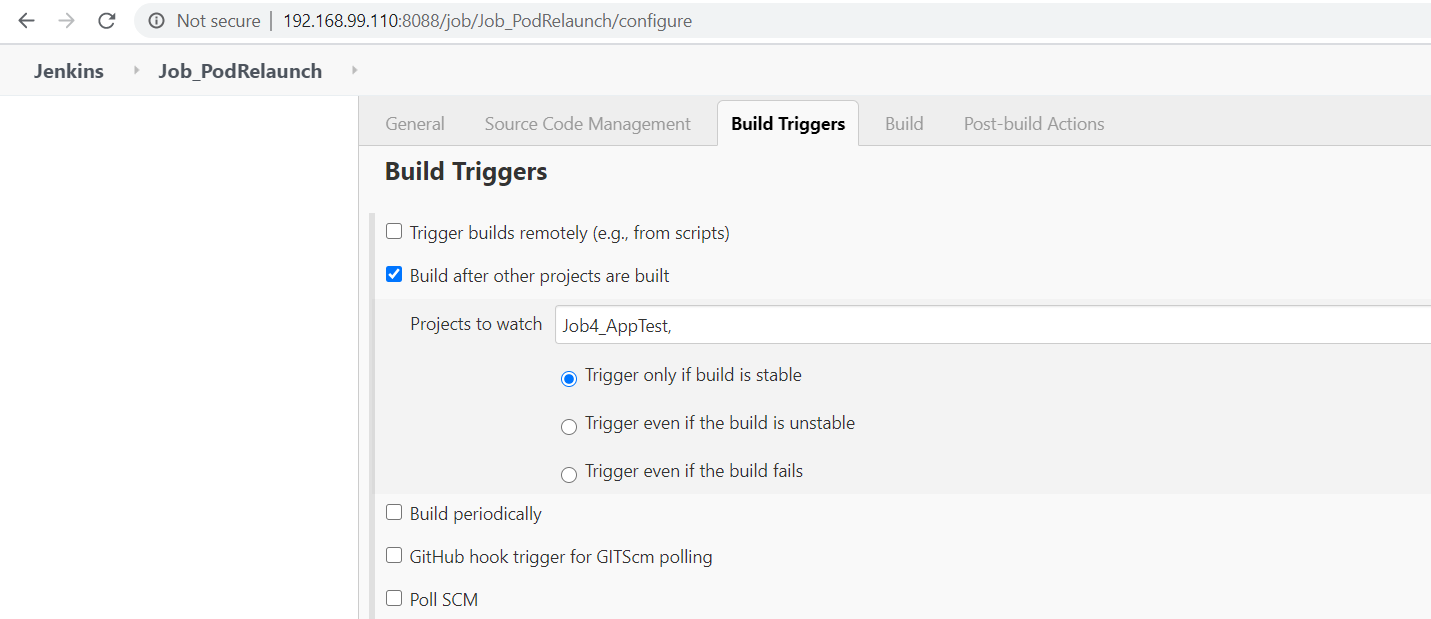


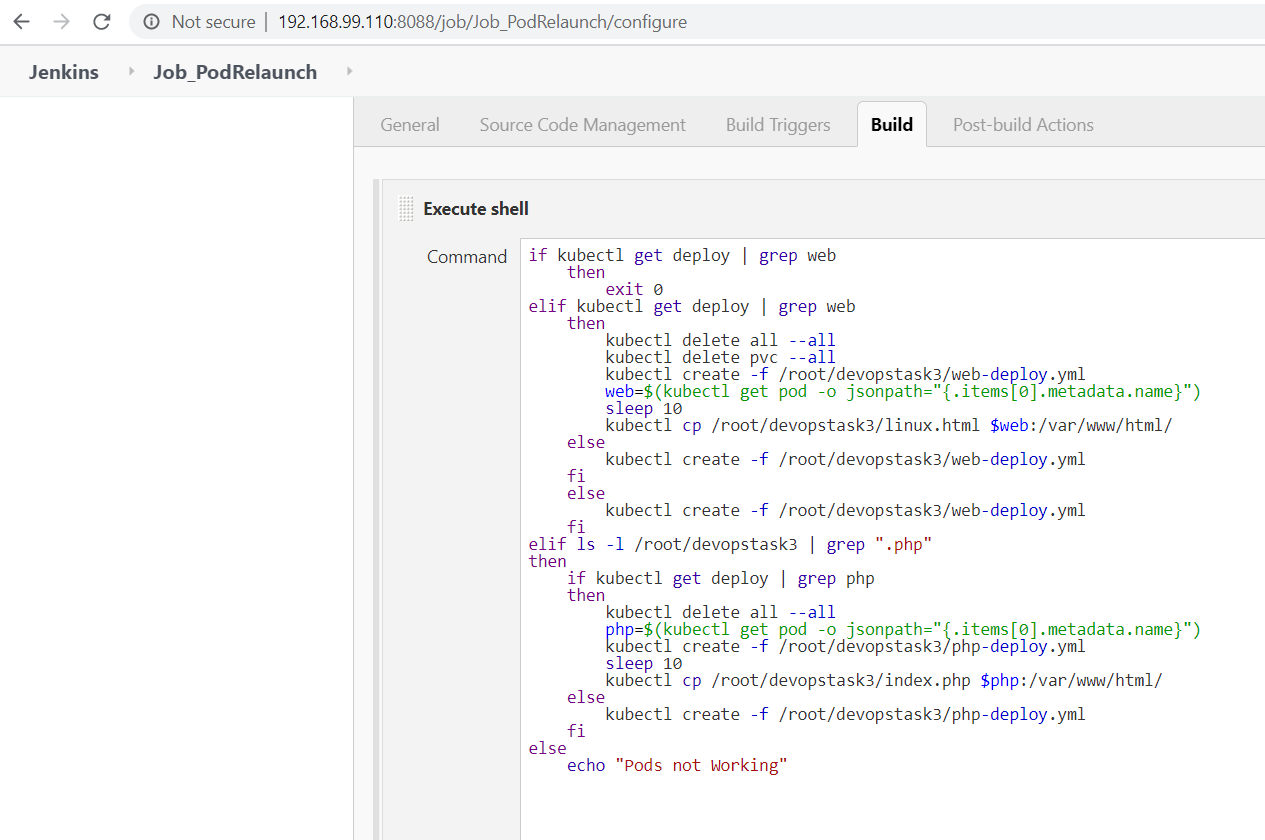


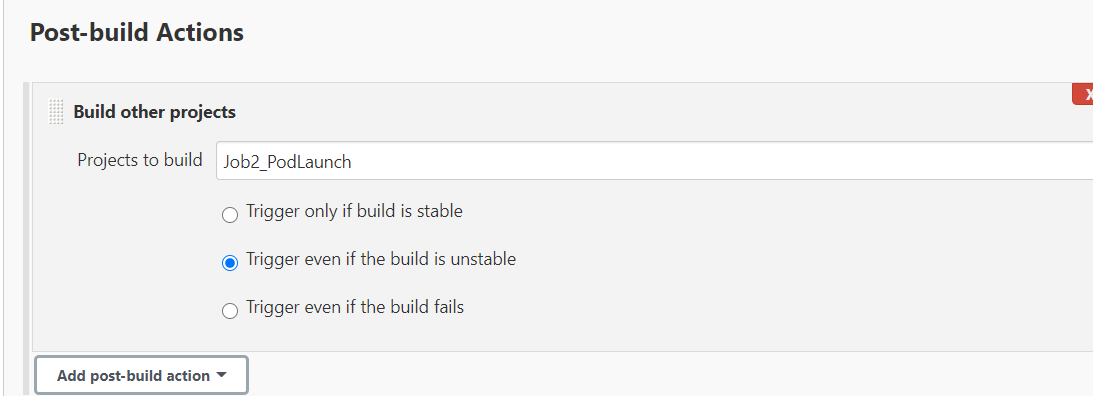


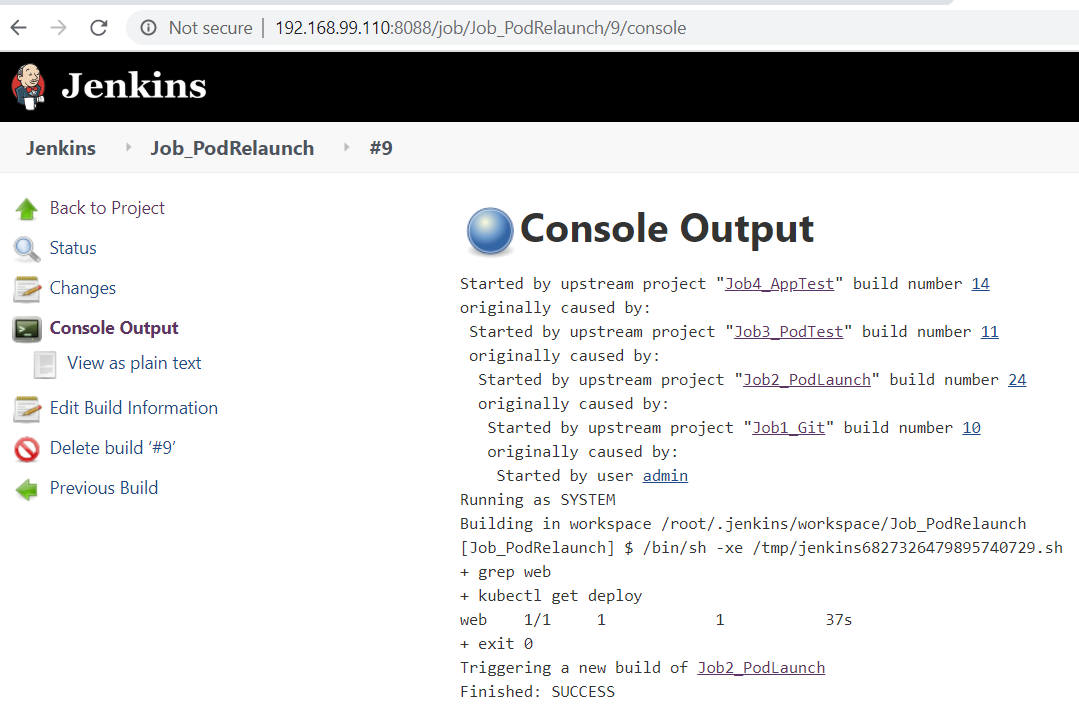
Output for the website testing

Job5: for container relaunch if any error occur.

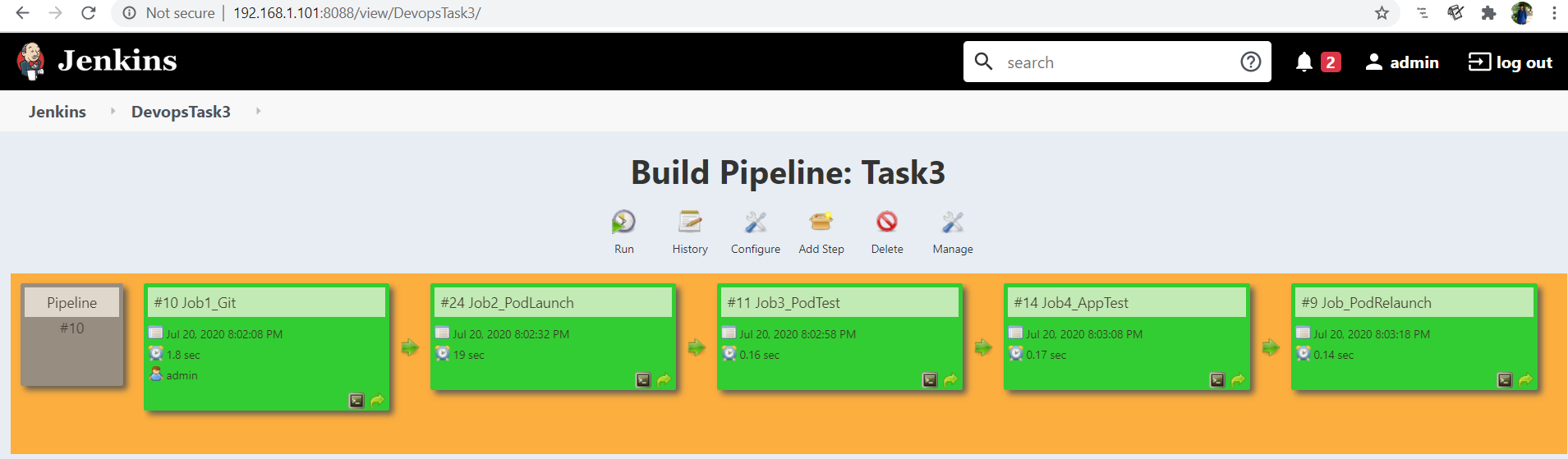




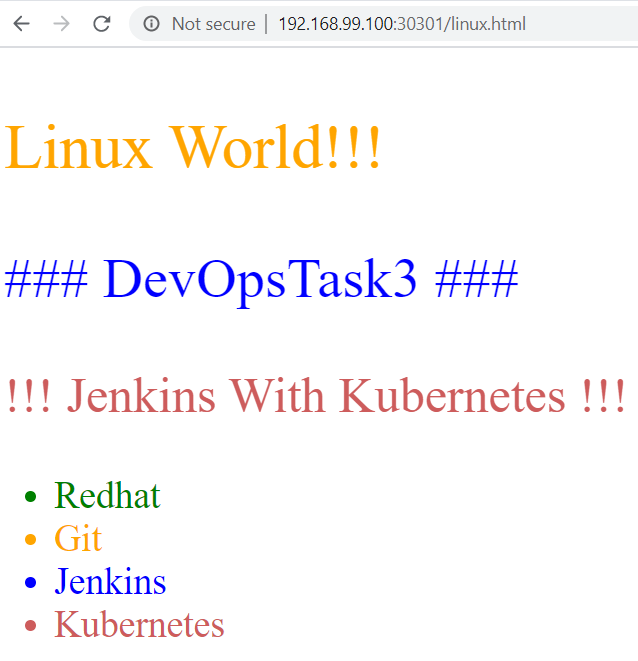




Output for pod relaunching



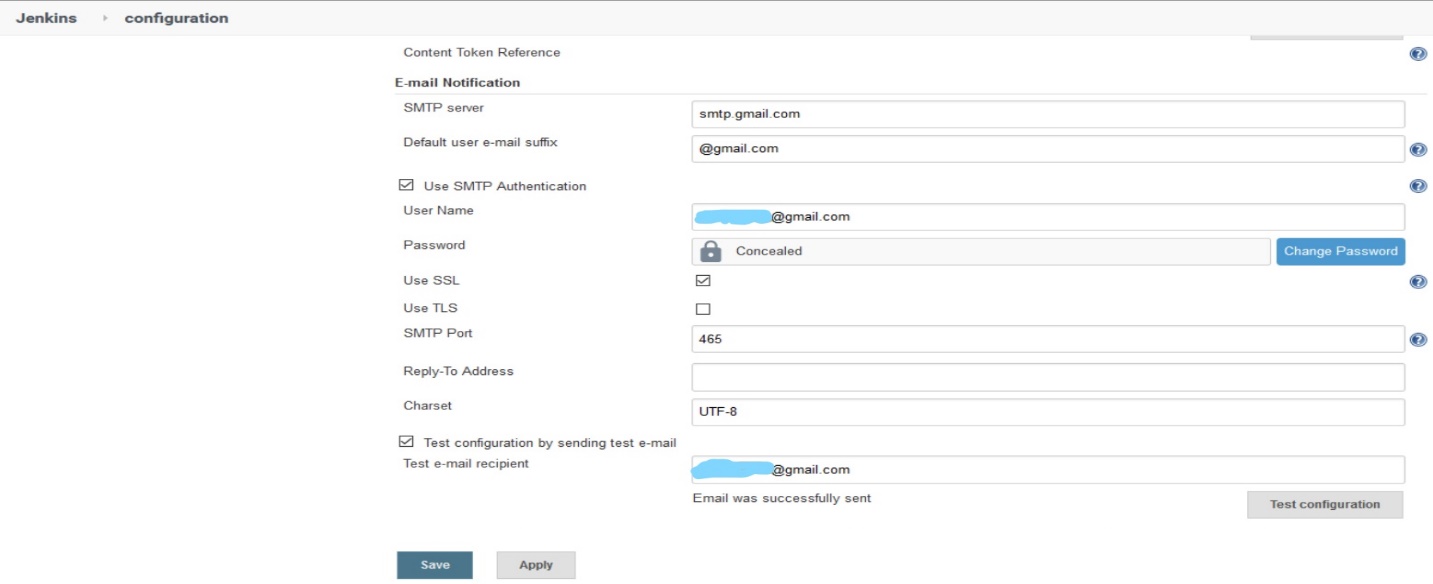
this is output obtained by **Build Pipeline**after the successful completion of all jobs

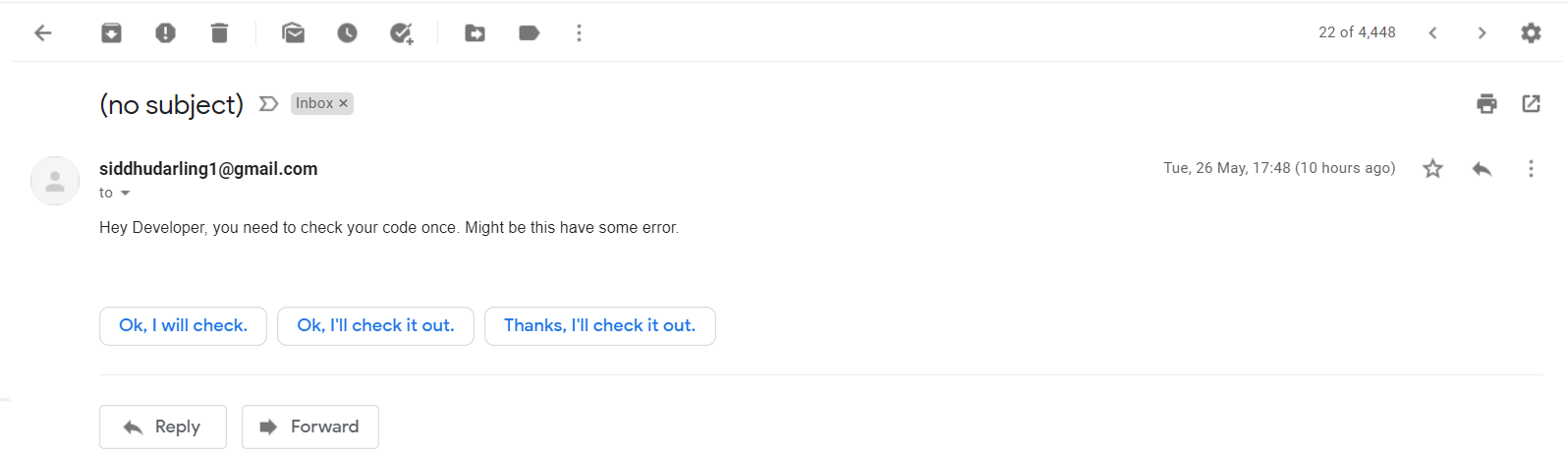


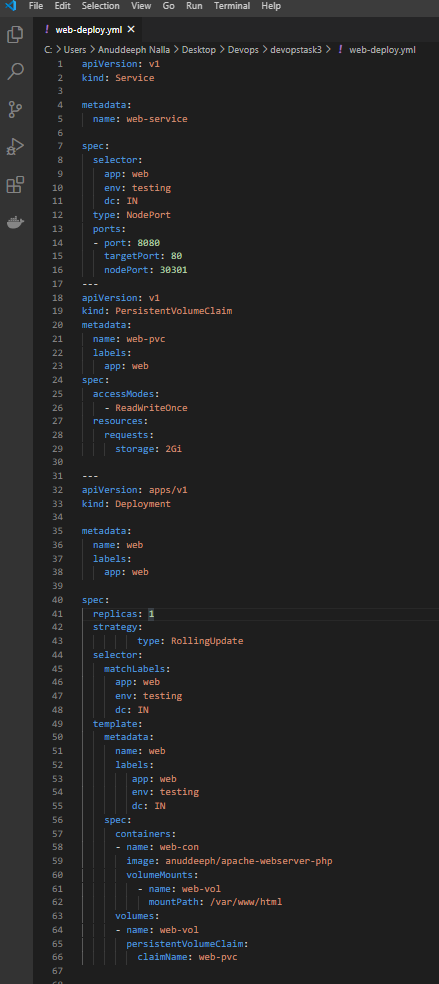
this is the output found when we visit to the url obtained when we run these code snippet

minikube service web-service --url

http://192.168.99.100:30301/index.html









this code launches the web server, it has persistent storage for coding file and exposed at port 30301 for outside world.

I am highly greatful to my mentor " [**Mr. Vimal Daga**](https://www.linkedin.com/in/vimaldaga/)**"**who teaches me these good technologies in right way and as a result this is completed successfully.

**For code files and Docker file visit my GitHub profile:**

<https://github.com/Anuddeeph/devopstk3.git>

<https://hub.docker.com/r/anuddeeph/jenkins_k8s>

https://www.linkedin.com/pulse/devops-automation-done-github-jenkins-kubernetes-anudeep-nalla