1. Create container image that's has Jenkins installed using dockerfile

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

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

4. Job1 : Pull the Github repo automatically when some developers push repo to Github.

5. Job2 : By looking at the code or program file, Jenkins should automatically start the respective language interpreter install image container to deploy code (eg. If code is of PHP, then Jenkins should start the container that has PHP already installed).

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

7. Job4 : if app is not working , then send email to developer with error messages.

8. Create One extra job job5 for monitor : If container where app is running. fails due to any reason then this job should automatically start the container again.

Solution

Step1:

a) We will create the environment of Jenkins in Docker file.

Activities Text Editor

Open Dockerfile /pro2 Save

```
FROM centos
RUN yum install sudo -y
RUN yum install wget -y
RUN yum install net-tools -y
RUN wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
RUN rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
RUN yum install java-11-openjdk.x86_64 -y && yum install jenkins -y
RUN yum install php -y
RUN yum install httpd -y
RUN yum install git -y && yum install python36 -y
RUN yum install /sbin/service -y
RUN echo -e "jenkins ALL=(ALL)  NOPASSWD:ALL">> /etc/sudoers
CMD sudo /etc/rc.d/init.d/jenkins start && /bin/bash
EXPOSE 8080
```

→FROM is used for the image that we want to use in our environment (It acts like docker pull)

→RUN is used for executing the command that the features required for project while building the new image

→CMD here is used to start Jenkins and keep the container live even after executing in this project

→Expose is used for patting, docker is isolated so we must expose, that the client should connect.

Step2: Building the image

docker build -t jenk:v5 /pro2/

→/pro2/ is the directory where Dockerfile is created.

```
File  Edit  View  Search  Terminal  Tabs  Help
@09f2378b5047:/        root@localhost:~/Desktop        root@localhost:/pro2
[root@localhost pro2]# gedit Dockerfile
[root@localhost pro2]# docker build -t jenk:v5 .
Sending build context to Docker daemon  7.168kB
Step 1/14 : FROM centos
 ---> 470671670cac
Step 2/14 : RUN yum install sudo -y
 ---> Using cache
 ---> 9da87f43c2a6
Step 3/14 : RUN yum install wget -y
 ---> Using cache
 ---> 1f9293ca4713
Step 4/14 : RUN yum install net-tools -y
 ---> Using cache
 ---> 05ba90c3ecb7
Step 5/14 : RUN wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.
repo
 ---> Using cache
 ---> d852bc3612ab
Step 6/14 : RUN rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
 ---> Using cache
 ---> 9e45a0ce1f54
Step 7/14 : RUN yum install java-11-openjdk.x86_64 -y && yum install jenkins -y
 ---> Running in 4f87485713d6
Jenkins-stable                                   13 kB/s |  17 kB     00:01
Dependencies resolved.
================================================================================
 Package                    Arch   Version                       Repo       Size
================================================================================
Installing:
 java-11-openjdk            x86_64 1:11.0.7.10-1.el8_1           AppStream 247 k
Installing dependencies:
 abattis-cantarell-fonts    noarch 0.0.25-4.el8                  AppStream 155 k
 adwaita-cursor-theme       noarch 3.28.0-2.el8                  AppStream 647 k
 adwaita-icon-theme         noarch 3.28.0-2.el8                  AppStream  11 M
 alsa-lib                   x86_64 1.1.9-4.el8                   AppStream 429 k
 at-spi2-atk                x86_64 2.26.2-1.el8                  AppStream  89 k
 at-spi2-core               x86_64 2.28.0-1.el8                  AppStream 169 k
 atk                        x86_64 2.28.1-1.el8                  AppStream 272 k
 cairo                      x86_64 1.15.12-3.el8                 AppStream 721 k
```

Step3: Starting Jenkins container

docker run -it –privileged -p 9997:8080 --name jenkos jenk:v5

```
@09f2378b5047:/
File  Edit  View  Search  Terminal  Tabs  Help
@09f2378b5047:/   root@localhost:~/Desktop   root@localhost:/pro2
[root@localhost pro2]# docker rm -f $(docker ps -a -q)
fa246f7879dc
[root@localhost pro2]# docker run -it --privileged -p 9997:8080 --name jenkos jenk:v5
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
Starting Jenkins                                           [  OK  ]
[root@a2588e8e093b /]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:cft3QSczy7moEczRSCLKZH85vMLz/md4W1orkIdIUjs root@a2588e8e093b
The key's randomart image is:
+---[RSA 3072]----+
|    o . . .      |
|   + o o.+ o     |
|    o ..*.+ . +..|
|     ...EB o ..*.|
|      +oSo=o  +. |
|       +. +o.. ..|
|        . .+o = .|
|       .  .o== o |
|        ..o+oo.  |
+----[SHA256]-----+
```

```
[root@a2588e8e093b /]# ssh-copy-id root@192.168.43.18
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '192.168.43.18 (192.168.43.18)' can't be established.
ECDSA key fingerprint is SHA256:vYGcxDZYIroX6tpDnzmbTfA9Tzxkz/MAV6rGw7R+zh0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are al
ready installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to ins
tall the new keys
root@192.168.43.18's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'root@192.168.43.18'"
and check to make sure that only the key(s) you wanted were added.

[root@a2588e8e093b /]# ssh root@192.168.43.18
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Thu May 14 07:31:21 2020
[root@localhost ~]# ls
anaconda-ks.cfg  Downloads   jenkins    my.py            project    Videos
Desktop          index.html  Music      original-ks.cfg  Public     z1
Documents        index.php   mycompose  Pictures         Templates
[root@localhost ~]# exit
logout
Connection to 192.168.43.18 closed.
```

→After Launching the container adding SSH key using command **ssh keygen -t rsa**

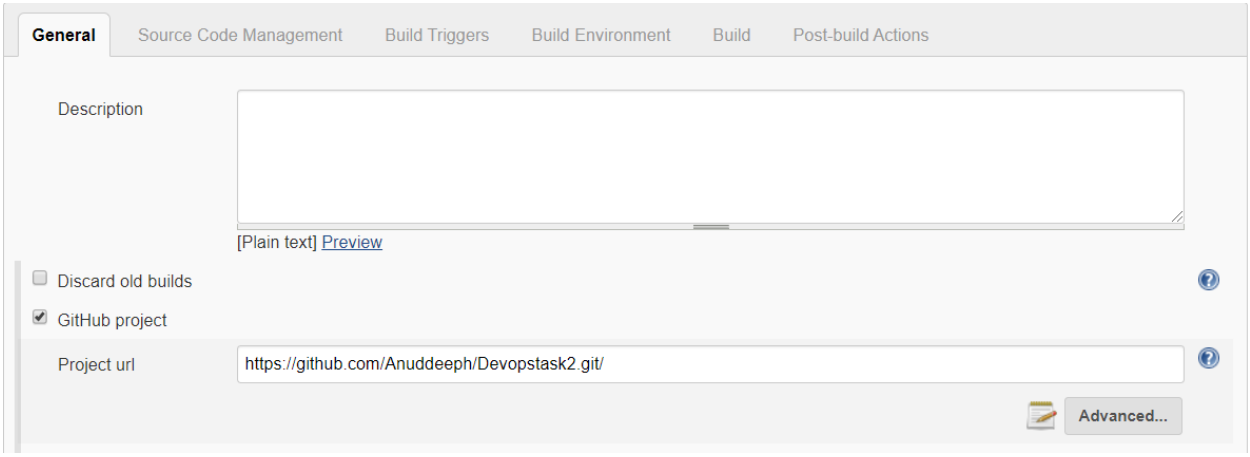
This command creates the ssh key and we have to copy it with the host i.e rhel8 using command **ssh-copy-id root@hostip [here bhostip == 192.168.43.18]**

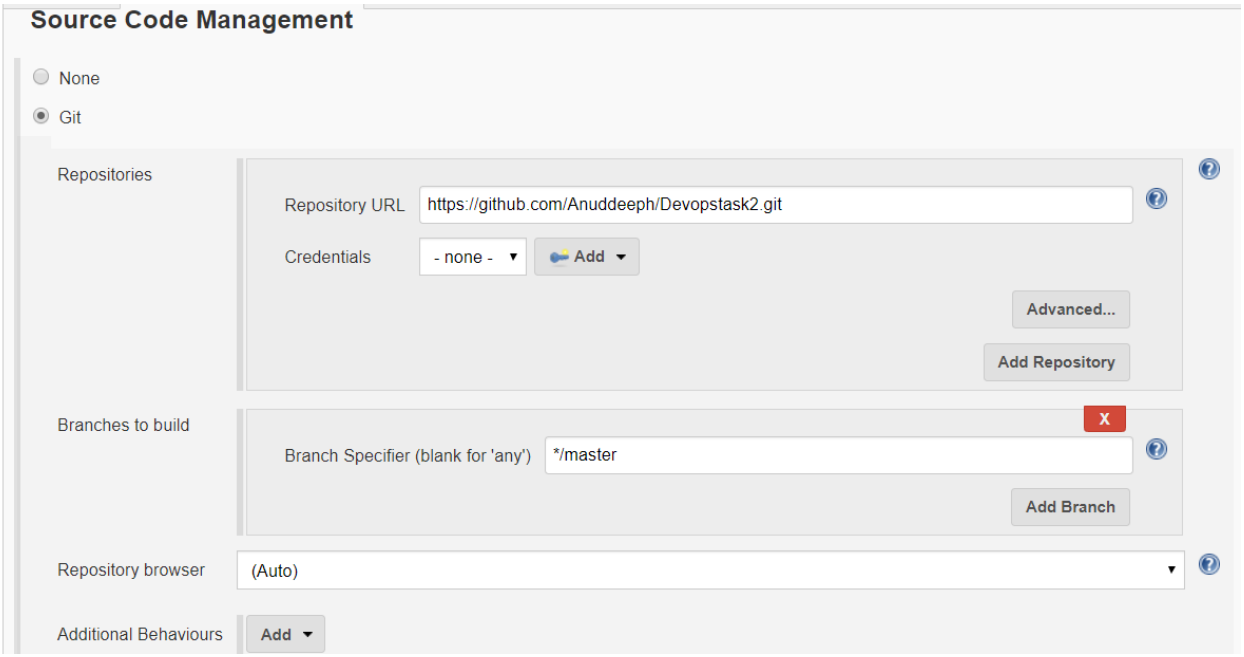
Step 4: Configuring Jenkins

Configuration is nothing but adding required plugins to use in the project like GitHub, Build Pipeline, Delivery Pipeline.

Step 5: Creating Jobs that we required in this project (i.e Automation)

a)Pulling the GitHub repo (job1)





General | Source Code Management | **Build Triggers** | Build Environment | Build | Post-build Actions

Build Triggers

- [ ] Trigger builds remotely (e.g., from scripts)
- [ ] Build after other projects are built
- [ ] Build periodically
- [x] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Build Environment

- [ ] Create Delivery Pipeline version
- [ ] Execute shell script on remote host using ssh

Build

Execute shell

Command

```
sudo rm -rf /Task2
sudo mkdir /Task2
sudo cp -vrf * /Task2
#sudo scp /Task2/* root@192.168.43.18:/var/lib/docker/volumes/DevOpsAL2/_data/
```

See the list of available environment variables

Console Output

```
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/job1
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Anuddeeph/Devopstask2.git
 > git init /var/lib/jenkins/workspace/job1 # timeout=10
Fetching upstream changes from https://github.com/Anuddeeph/Devopstask2.git
 > git --version # timeout=10
 > git fetch --tags --progress -- https://github.com/Anuddeeph/Devopstask2.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Anuddeeph/Devopstask2.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Anuddeeph/Devopstask2.git # timeout=10
Fetching upstream changes from https://github.com/Anuddeeph/Devopstask2.git
 > git fetch --tags --progress -- https://github.com/Anuddeeph/Devopstask2.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 0638b544d0ad82506feacd73ab34ff6dec062fb1 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 0638b544d0ad82506feacd73ab34ff6dec062fb1 # timeout=10
Commit message: "final"
First time build. Skipping changelog.
[job1] $ /bin/sh -xe /tmp/jenkins1809019860098849567.sh
+ sudo rm -rf /Task2
+ sudo mkdir /Task2
+ sudo cp -vrf index.html lw.html lw.txt my.php /Task2
'index.html' -> '/Task2/index.html'
'lw.html' -> '/Task2/lw.html'
'lw.txt' -> '/Task2/lw.txt'
'my.php' -> '/Task2/my.php'
+ sudo scp /Task2/index.html /Task2/lw.html /Task2/lw.txt /Task2/my.php root@192.168.43.18:/var/lib/docker/volumes/DevOpsAL2/_data/
Finished: SUCCESS
```

In this job1, we are removing the existing directory, creating the new directory, and copying the files from GitHub repo. # also we can use docker volume to store data and we can mount to docker container.

B) Launching the container and copying the files to webserver (location /var/www/html)

General | Source Code Management | **Build Triggers** | Build Environment | Build | Post-build Actions

☑ Build after other projects are built

Projects to watch: job1,

- ◉ Trigger only if build is stable
- ○ Trigger even if the build is unstable
- ○ Trigger even if the build fails

☐ Build periodically

☐ GitHub hook trigger for GITScm polling

☐ Poll SCM

Build Environment

☐ Create Delivery Pipeline version

☐ Execute shell script on remote host using ssh

Build

Execute shell

Command

```
#cd /Task2
#file -s *
#cd ..
if sudo ssh root@192.168.43.18 docker ps -a | grep webos
then
sudo ssh root@192.168.43.18 docker rm -f webos
fi
#sudo rm -rf /Task
#sudo mkdir /Task
sudo scp /Task2/* root@192.168.43.18:/root/Task
sudo ssh root@192.168.43.18 docker run -dit -p 89:80 -v /root/Task:/var/www/html/ --name webos vimal13/a
```

Console Output

```
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/job2
[job2] $ /bin/sh -xe /tmp/jenkins9094706760695449310.sh
+ grep webos
+ sudo ssh root@192.168.43.18 docker ps -a
5ac46745caa9        vimal13/apache-webserver-php   "/usr/sbin/httpd -DF???"   About an hour ago   Up About an hour    0.0.0.0:89->80/tcp       webos
+ sudo ssh root@192.168.43.18 docker rm -f webos
webos
+ sudo scp /Task2/index.html /Task2/lw.html /Task2/lw.txt /Task2/my.php root@192.168.43.18:/root/Task
+ sudo ssh root@192.168.43.18 docker run -dit -p 89:80 -v /root/Task:/var/www/html/ --name webos vimal13/apache-webserver-php
dcc31aa956815d70c25afce557f6ca3386b0ae386c6a78455d450d2b1e5c8f43
Finished: SUCCESS
```

while running the container we can also mount the volume DevOpsAL2

C) Testing and Reporting error

Build Triggers

- [ ] Trigger builds remotely (e.g., from scripts)
- [x] Build after other projects are built

Projects to watch: job2,

- (•) Trigger only if build is stable
- ( ) Trigger even if the build is unstable
- ( ) Trigger even if the build fails

- [ ] Build periodically
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM

Build Environment

- [ ] Create Delivery Pipeline version
- [ ] Execute shell script on remote host using ssh

Build

Execute shell

Command

```
status=$(curl -o /dev/null -s -w "%{http_code}" 192.168.43.18:89/my.php)
if [[ $status == 200 ]] ; then exit 0 ; else exit 1 ; fi
```

Post-build Actions

E-mail Notification

Recipients: Anudeep.nalla1@gmail.com

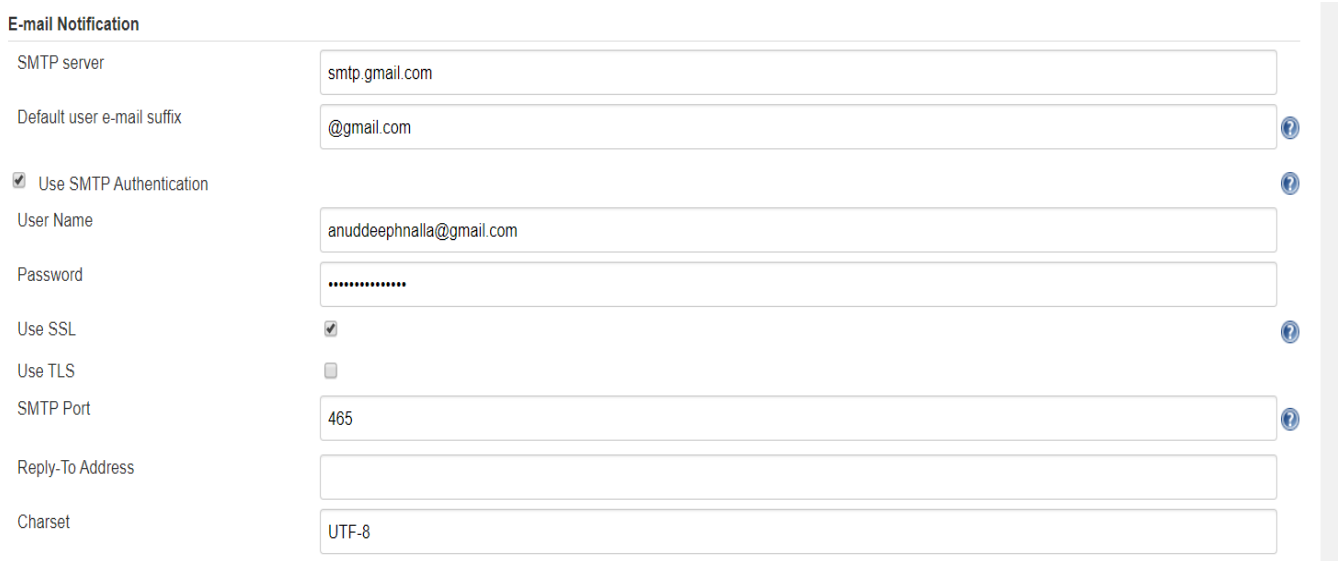
Whitespace-separated list of recipient addresses. May reference build parameters like $PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

- [x] Send e-mail for every unstable build
- [ ] Send separate e-mails to individuals who broke the build

Console Output

```
Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/job3
[job3] $ /bin/sh -xe /tmp/jenkins6083380063223169535.sh
++ curl -o /dev/null -s -w '%{http_code}' 192.168.43.18:89/my.php
+ status=200
+ [[ 200 == 200 ]]
+ exit 0
Finished: SUCCESS
```

- ➔ In this Testing, we are storing the status code of php page and dumping the output to null, status code of success is 200, success code for error in code is 500. And we are sending mail if there any unstable build occurs. Exit 0 is to execute the success and exit 1 is to execute the failure
- ➔ Go to manage Jenkins → configure system

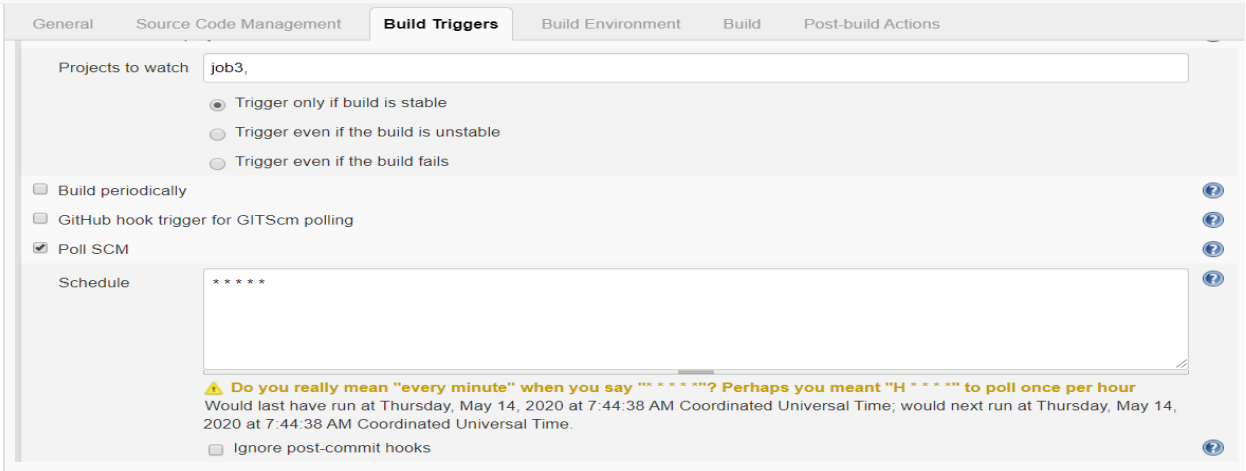


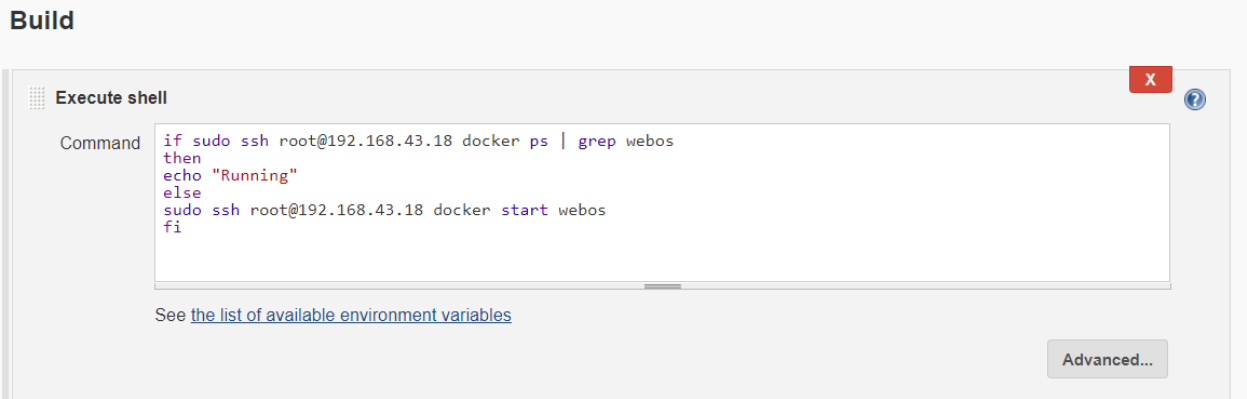
If then also you face error, then goto your jenkos container

```
vim /etc/sysconfig/jenkins
#change
JENKINS_JAVA_OPTIONS="-Djava.awt.headless=true -
Dmail.smtp.starttls.enable=true -Dmail.smtp.ssl.protocols=TLSv1.2"
```

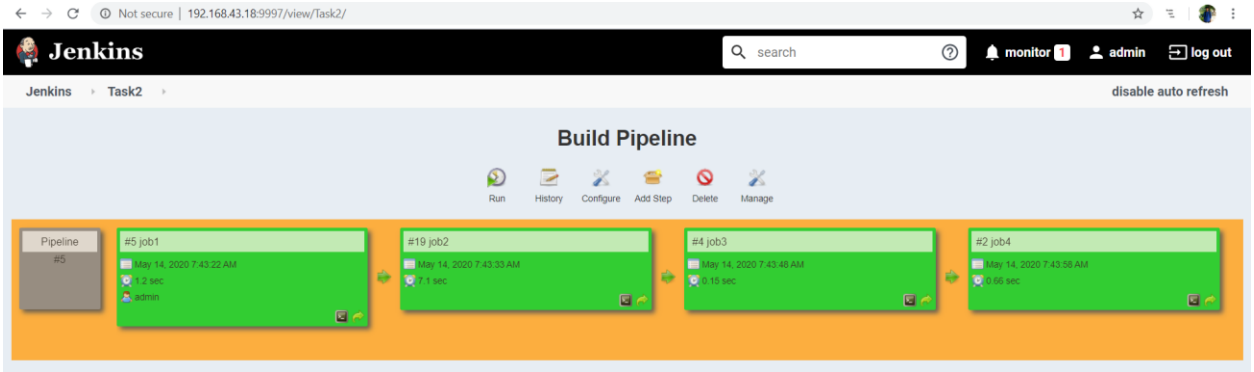
Then go to google setting → security → on less secure app.

D) Monitor the container (job 4)





This will keep checking the web container after (provided) period and if container is down the it will start the container.



This is how the Final pipeline looks:

Github repo URL: https://github.com/Anuddeeph/Devopstask2.git

t took me more than 20+ hrs. to make this project. I faced many errors while making this project. But every error was a great learning step. I learnt a lot from this project. I am very thankful to Mr. Vimal Daga sir ,who is teaching me such a great content and I am very much thankful to the volunteers and my group members also who encouraged us to solve the error by our own.

If you have any query or suggestion , Please let me know

Thank You