**DevOps Training**

**Trainer:** Hiran Ram Babu or HRB

**Date:** 23-09-2019 to 25-09-2019

**Mobile:** 8825410600

**Agenda of Session:**

* Docker
* Gitblit
* Jenkins
* Ansible
* Nagios
* Sonar

<http://mjit.in/post.php?slug=docker-tutorial>

<https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

**Day 1:**

**What is Devops?**

* Culture
* Together
* Informed
* Feedback – kick start from stage 1 if any failure
* Pipelines – building the code

**Tools used in Devops**

* Planning – JIRA
* Code – IDE
* Dashboard – Kanban
* Store – Repository
  + Distributed – GIT
  + Centralized – SVN
* Pipelines –Jenkins
* Build – Maven, Ant, MS Build
* Artifacts – Filesystem, Nexus, Azure Artifacts
* Testing – Selenium
* Releases - Ansible, Pupet, Saltstack, Chef
* Deploy
* Monitoring –Nagios

Being a devops engineer should know Pipeline and Release Deploy tools

While both **RPA** and **RBA** operate under similar concepts, the difference typically lies in how each is used as well as their complexity. ... Essentially, it can be said that **RPA** is automation for the end-user while **RBA** is more behind-the-scenes automation.

[Robotic Process Automation (RPA) vs. Run Book Automation](https://www.linkedin.com/pulse/robotic-process-automation-rpa-vs-run-book-vinil-g)

**Method 1:**

Traditional Method: Hardware: Physical

A1, A2, A3

OS

H/W

Capex – Spent all money for building home (initial setup)-CTB

Opex – Taxes paying for water bill etc (Maintenance)-RTB

Method 2: VM Workstation

1. Type 1 Hypervisor

App1 App2 App3

OS OS

Base Station -Hypervisor

OS

H/W

**Dev/Test**

1. Type 2 Hypervisor - Virtualization

App

OS OS

ESXi - Hypervisor

H/W

**PROD**

3. Containerization

Server less Architecture: e.g. Containers

FAAS – SG Lambdas

Architecture:

Vm – docker engine container

Container

Hyper V / Oracle Vbox

Windows Pro

Laptop

Linux:

Single Node

C1,C2,C3

C1,C2,C3

C1,C2,C3

Unix OS

H/W

Multi Node:

C1,C2,C3

C1,C2,C3

C1,C2,C3

VM

VM

VM

Hypervisor

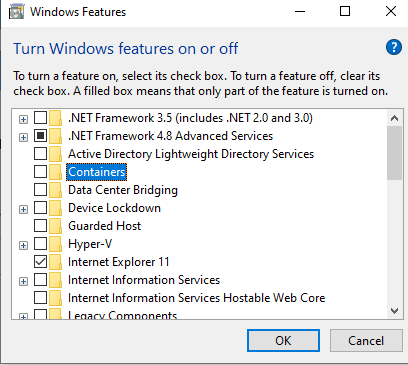
Unix OS

H/W

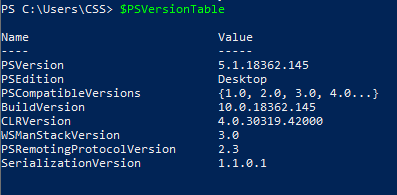
Kubernetes – Manage the container

Swarm –

**Installation Docker and Setup:**



Make sure Containers and Hyper V should be disabled for windows



If it is 3.x PS version then most of the commands will not work. Better to have 5.1

Check whether docker is installed or not



If not installed download it

<https://www.docker.com/products/docker-desktop>

**Linux Docker Toolbox:**

Docker Engine

GItBash

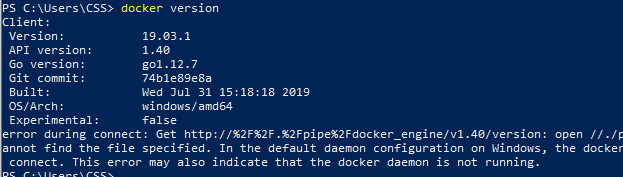
Virtual Box

**Windows Docker Toolbox:**

Git

Hyper V enabled





Code Repository for Docker:

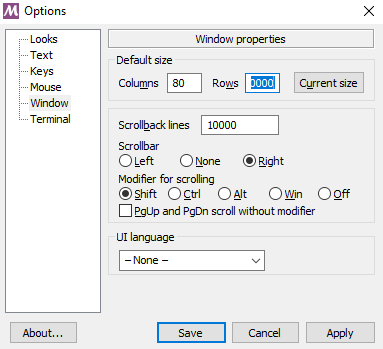
<https://hub.docker.com/signup>

Sign up and Login

Now use two terminal

1. Powershell
2. Git Bash

**Git Bash**



**Steps to follow:**

Login / Signup

Pull

Images / Image

Run – Detached, Interactive, Terminal, Port, Volume, Network

Attach

Commit

Tag / Rename

Push

Image to Container conversion should use Run command

Container to Image – Commit

Repo to Image – Pull

Image to Docker Repo – Push

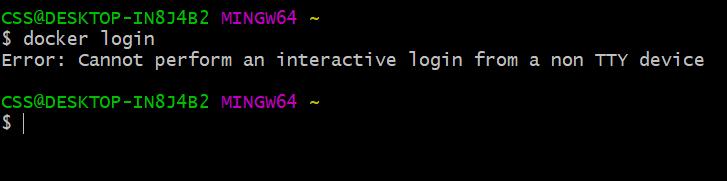
To use in local save & load command will use:

Save convert to -> .tar

Load - >

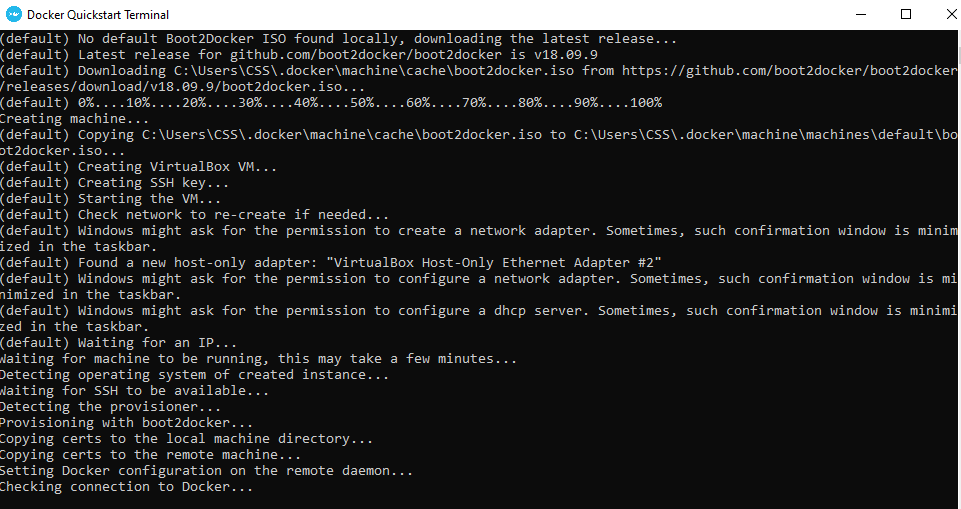
Docker Client – Type - Power Shell or Git Bash

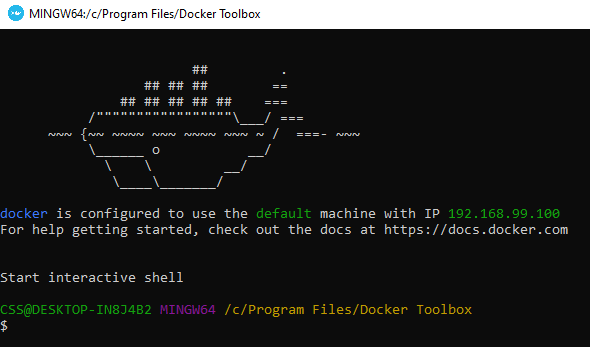
Docker Enine - Execution – Virtual Machine



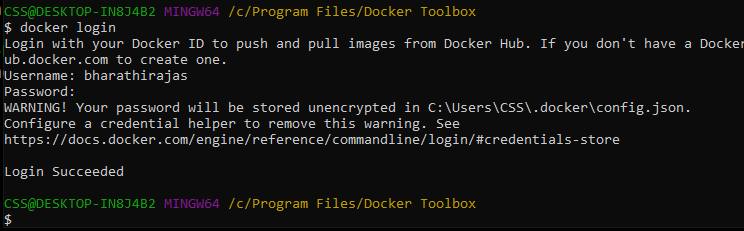
If any error come like above then use docker

Docker Quickstart Terminal

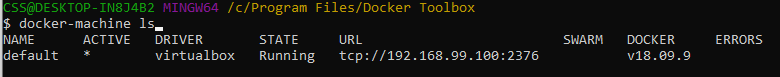




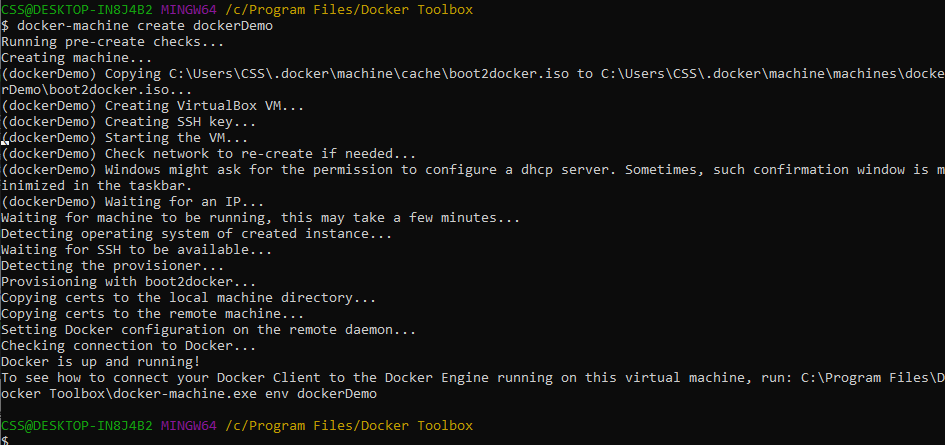
Ship image should display.



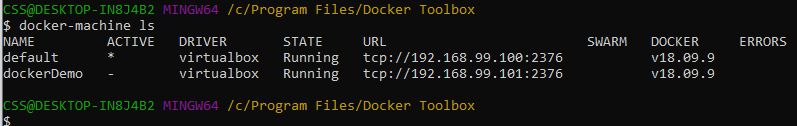
docker-machine ls – manage the VM, list the VM



Docker-machine create <name>



To verify the new docker



**Minimum space leave for Windows 10 machine:**

20GB Harddisk

1 CPU

2GB – RAM

Docker-machine create –help

**Default Installation Path of docker:**

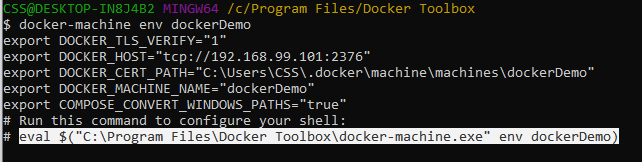
C:\Users\CSS\.docker\machine\machines\default\config.json

To see all configuration stuff either in config.json or below command

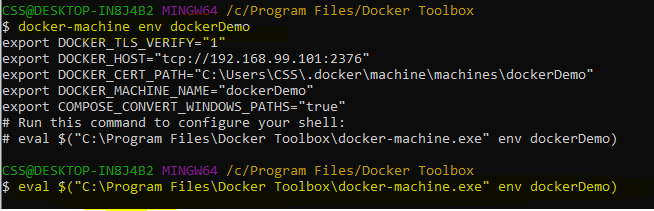
Docker-machine inspect <dockerName>

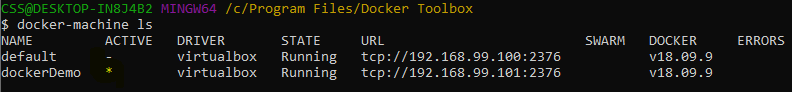
Make newly created docker as **active**:

Docker-machine env <dockerName>



Copy the highlighted one and paste it and enter





Provision is a command will recreate all container. Don’t use this in production.

Regenerate –search

**Stop Docker Container:**

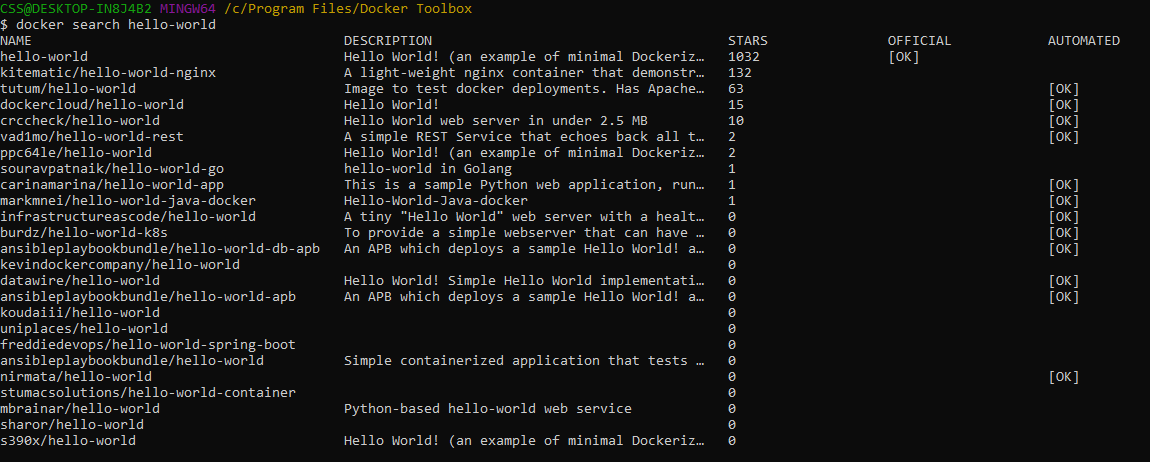
docker-machine stop <name1, name2>

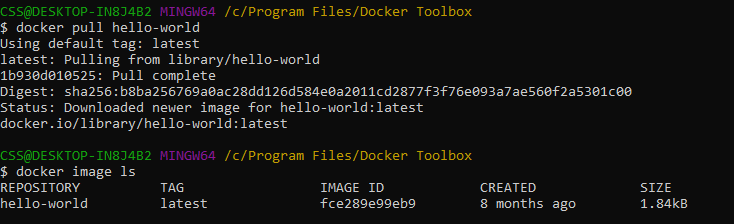
**Start Docker Container:**

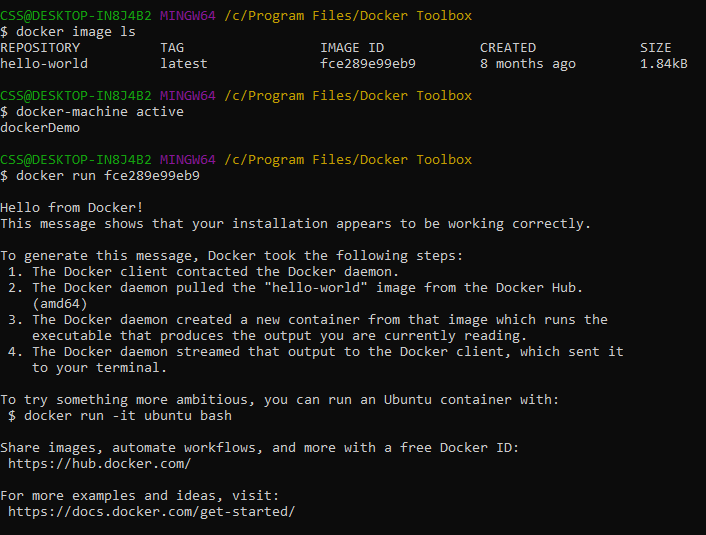
docker-machine start <name1, name2>

See list of docker image:









Docker container ls

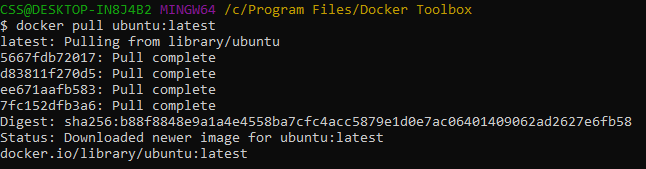
Docker ps

Will show only running container

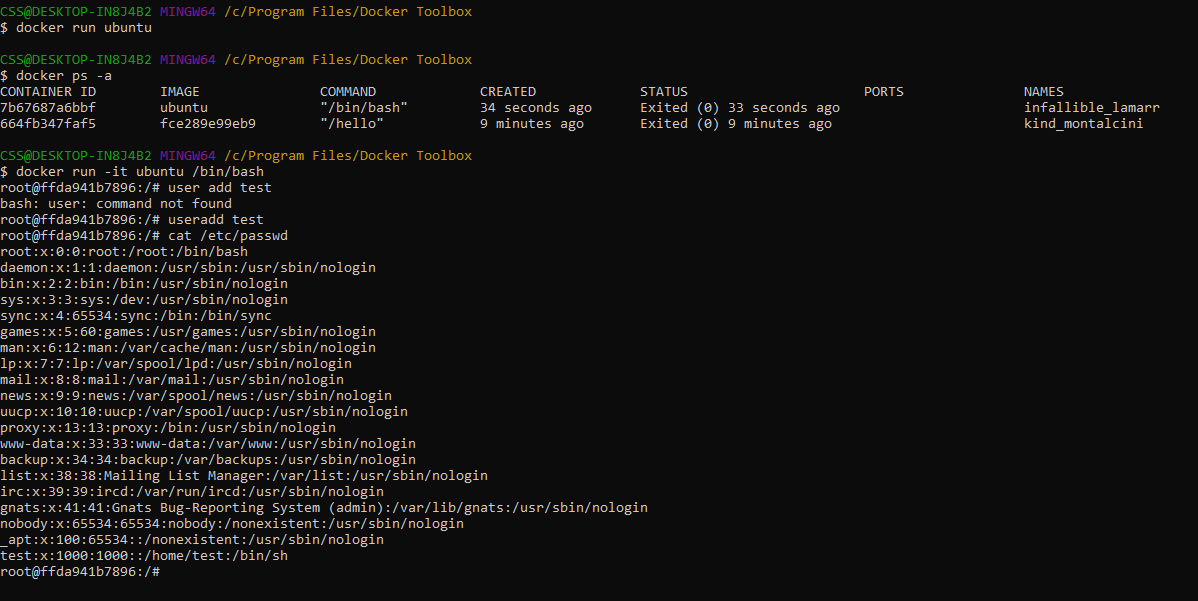
Docker ps –a

Will show all available container either active or inactive

Installing Ubuntu on docker







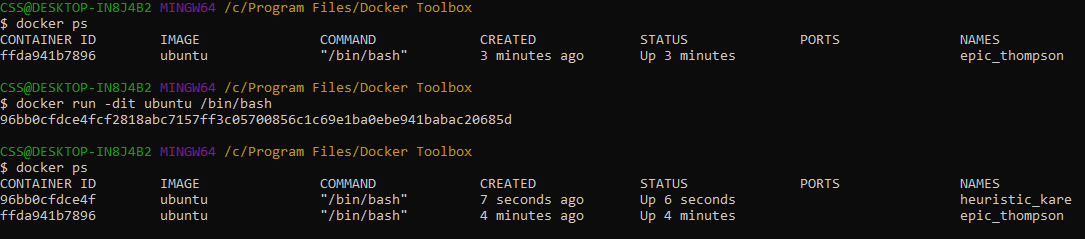
ctrl + p + q - control termination command

dont use exit - it will quit all containers

-I interactive

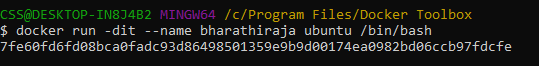
-d detached - dont want to enter in to console

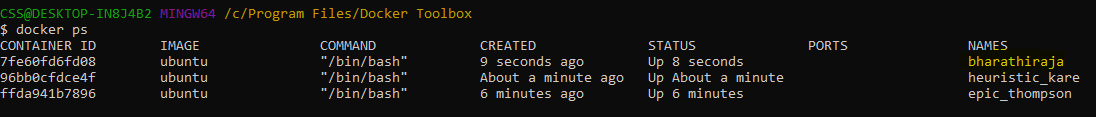
-t terminal

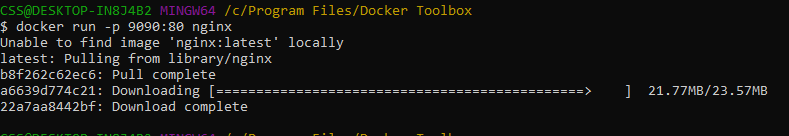


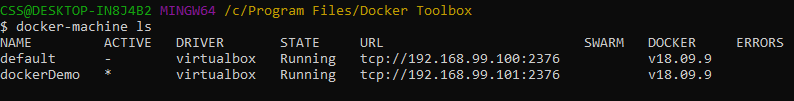
Docker <options> <commands>

docker run -dit --name bharathiraja ubuntu /bin/bash



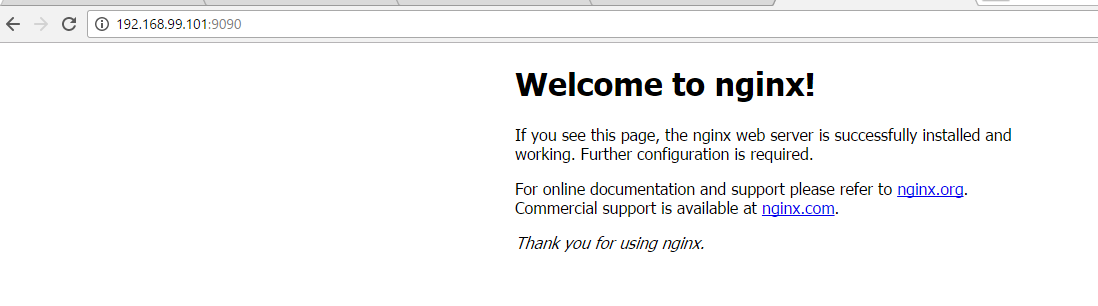






9090 - docker machine

80 – nginx



Refer - <https://hub.docker.com/_/nginx>

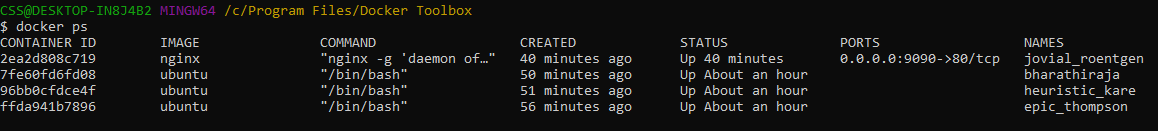
**To create new image:**

Docker compose using yaml file

Yaml – Python (indentation) + JSON

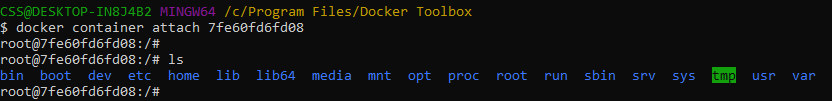
**Docker-compose.yml**

Save and Load is for loading tar file



Docker exec <id> hostname - Without connecting to docker need to know details

Want to get inside in to container use attach command

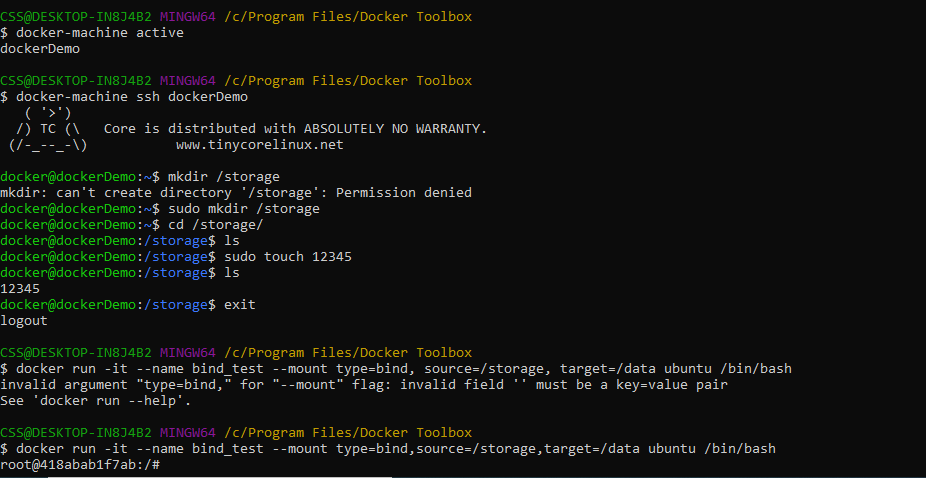


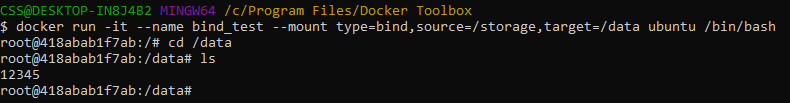
**Types of Mounts:**

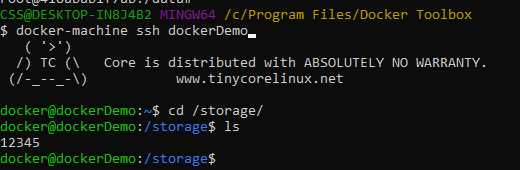
* Volume -
* Bind – connecting mobile with laptop and can see files in both devices
  + Source – VM / DM
  + Target – Container
* Temporary file system(tmpfs) – Only destination

By default docker swarm uses Bind

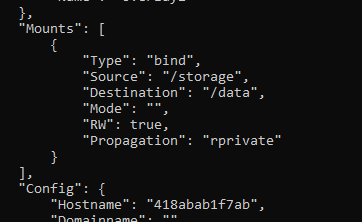
**Create space for the container**



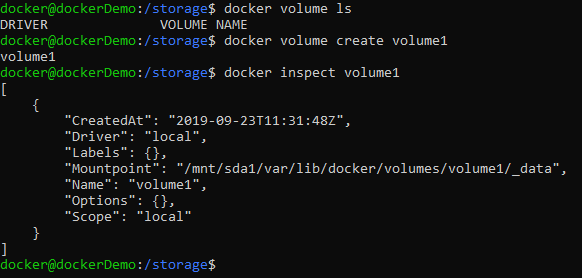
Ctrl P Q

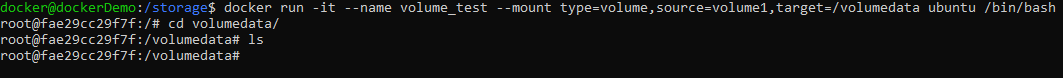




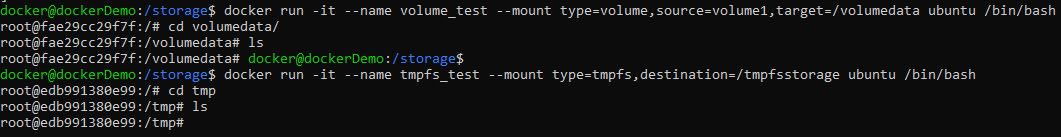


**Volume:**

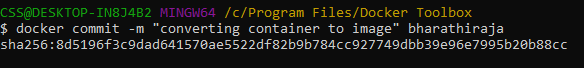


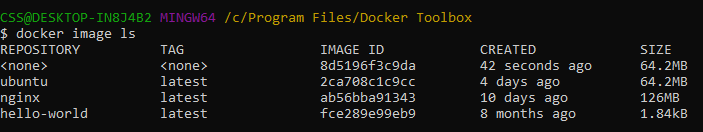


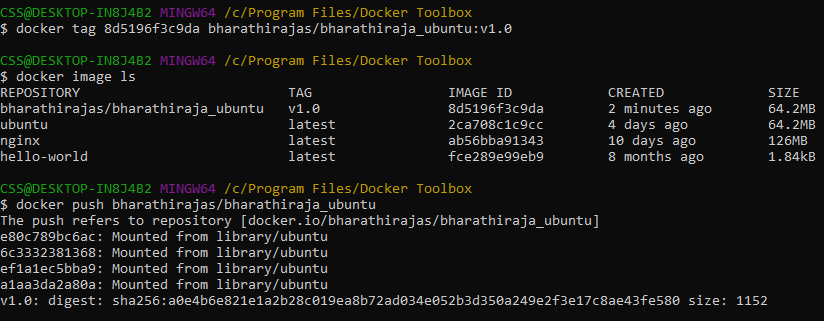
**Tmpfs:**











**Network:**

Every network will have DHCP

Types:

Bridge – Same n/w like the

Host only – only within the host

NAT (Network Address Trans)- IP to IP config

Physical NIC

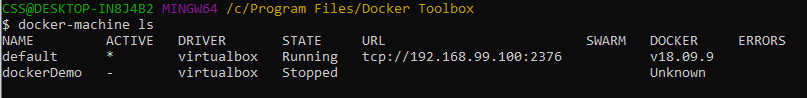
Virtual NIC

Docker network connect <network name> <container name>

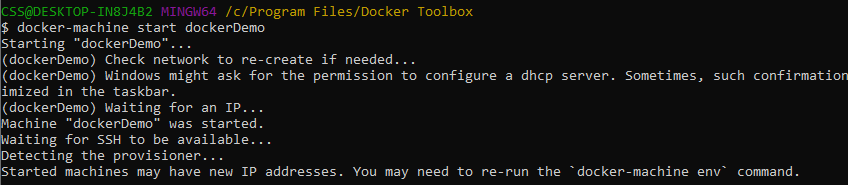
[info@mjit.in](mailto:info@mjit.in)

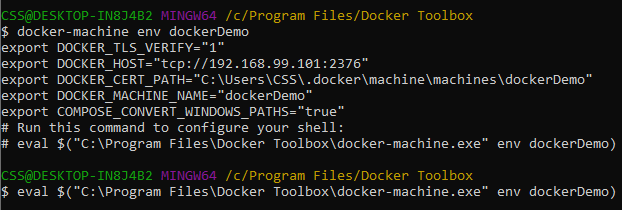
**Day 2: Agenda**

* Network
* Gitblit
* Jenkins
* Ansible



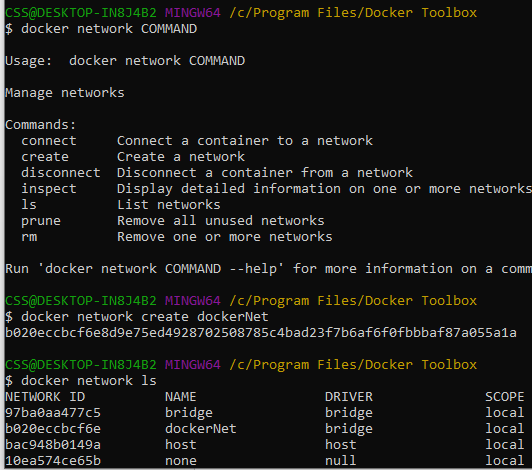
Active is default container need to change to mine



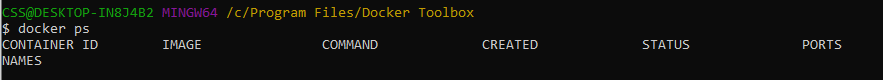




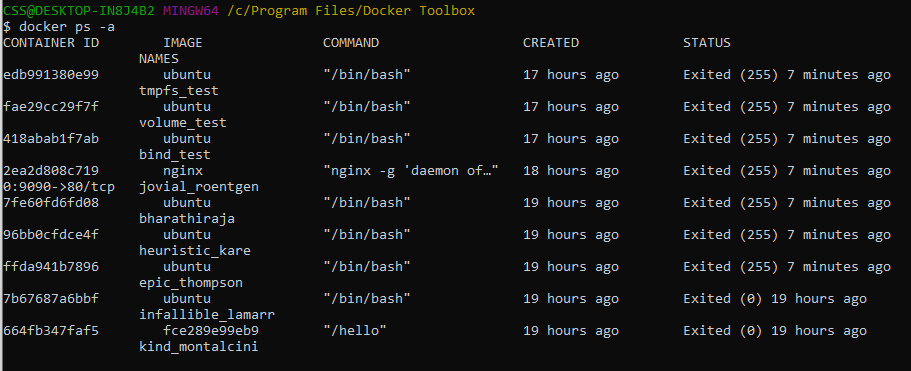
**Creating network:**

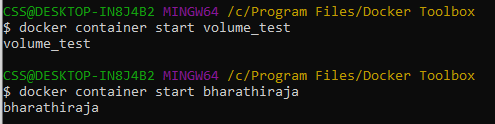


Bridge, Default – default containers – doubt

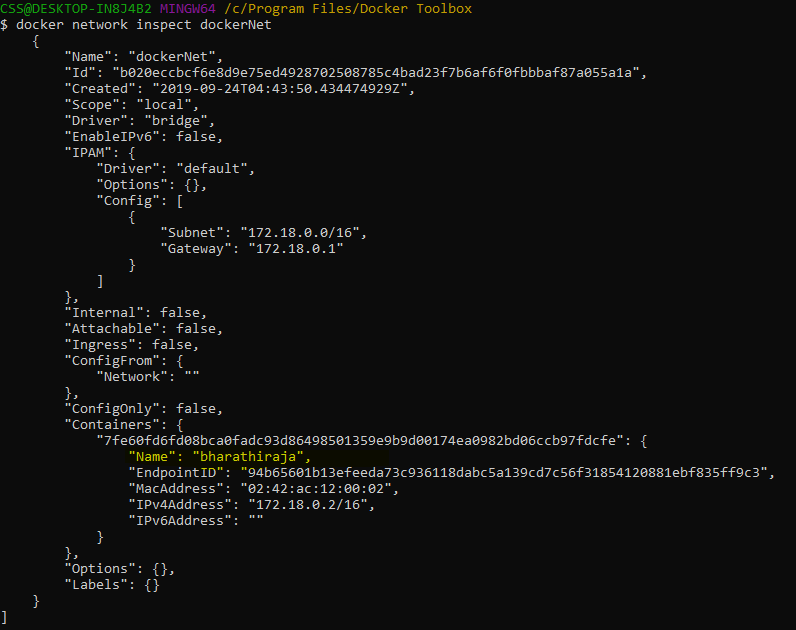


No docker available, to connecting the network docker should be available







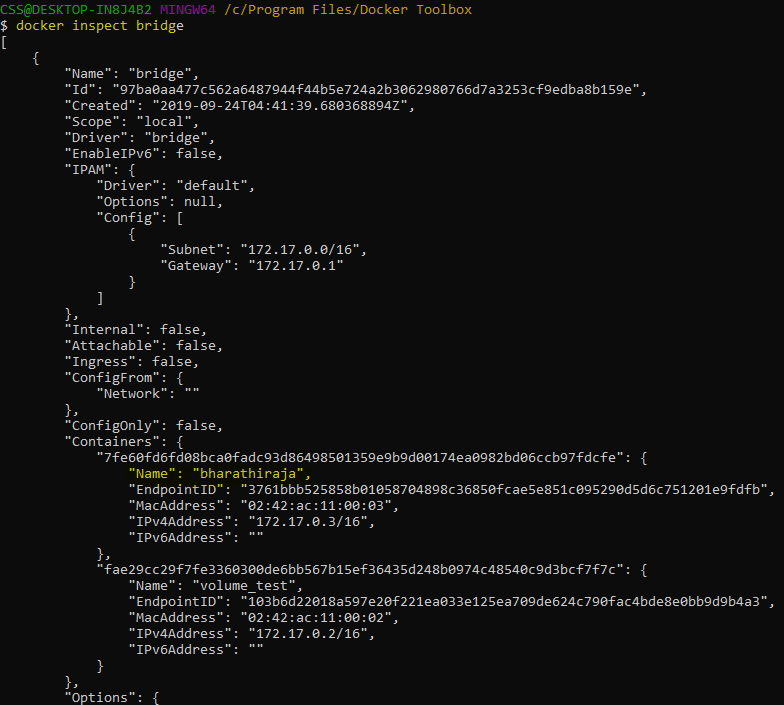


I can connect many network to the container

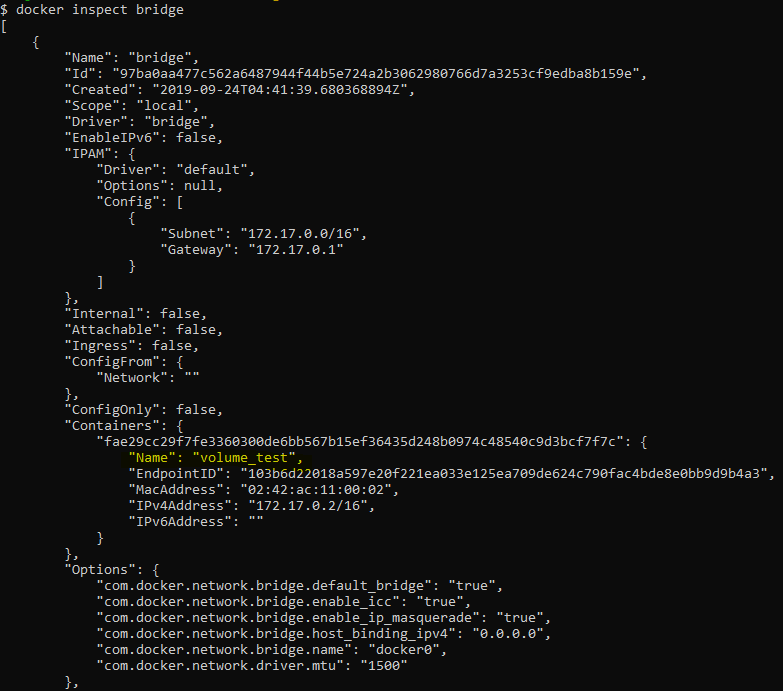
**Remove the network from container:**

Docker network disconnect bridge bharathiraja

Bridge is default container we don’t want to run our container in default. So disconnect it and make use our new created container



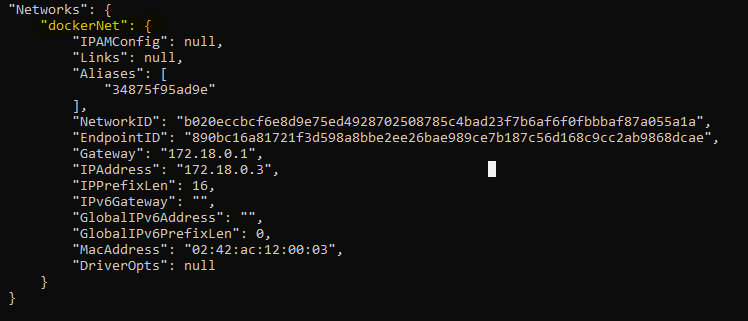




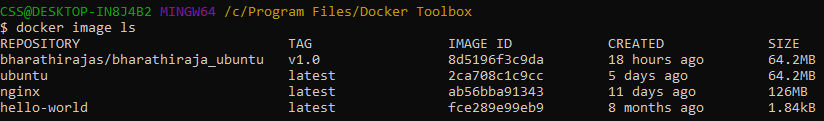
Now make sure bridge container doesn’t have our own container and do run command



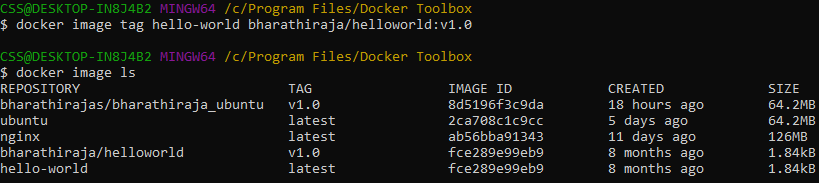




winpty docker login

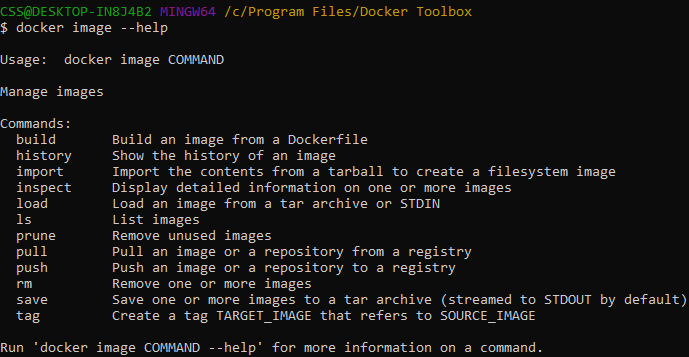


Going to push code in to repo

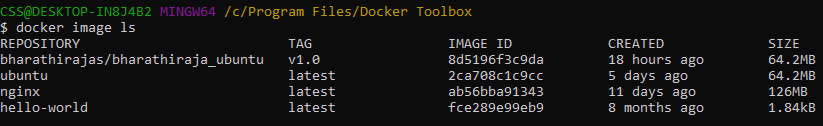


I have wrongly with the tag command, my repo name is bharathirajas not bharathiraja

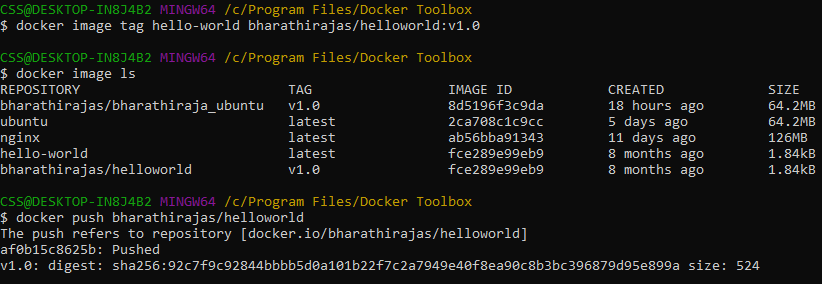
So I wanted to remove from this image

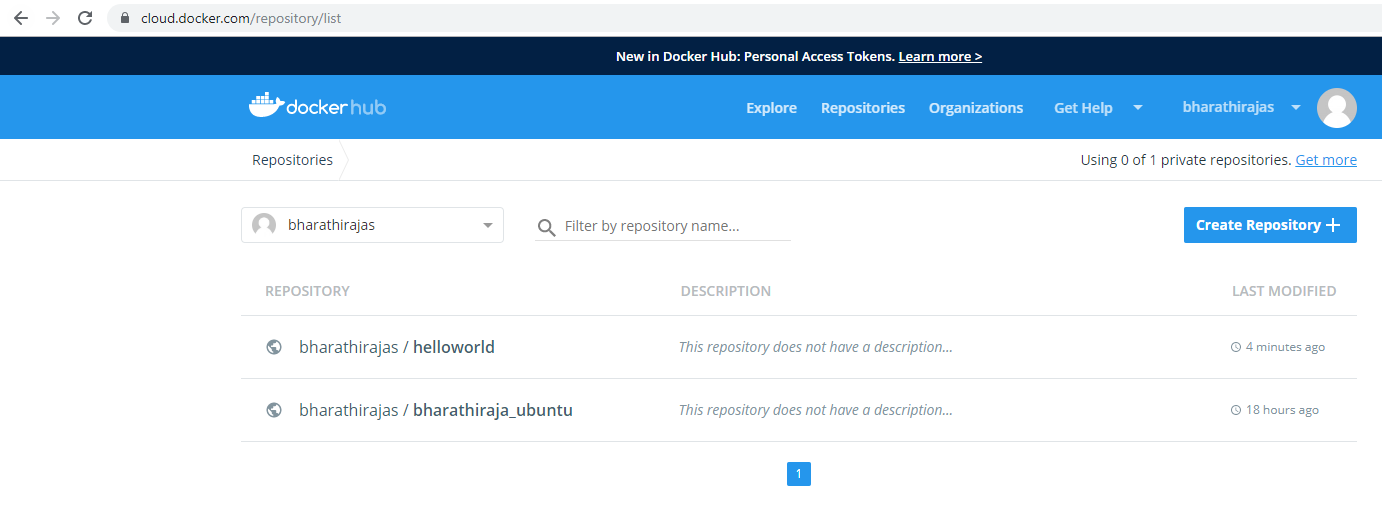




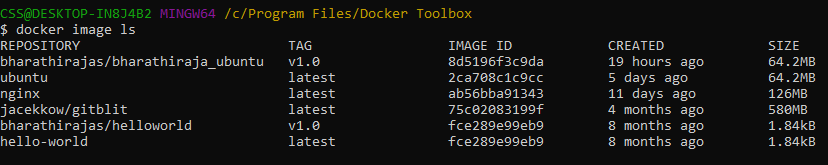


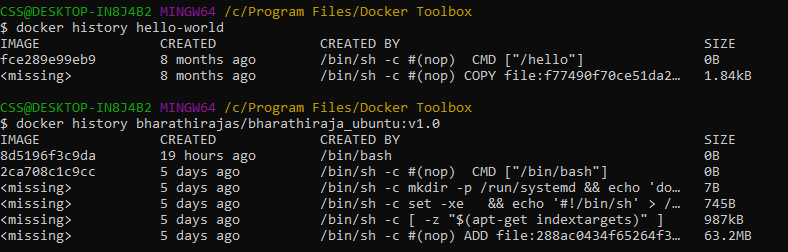
Now am doing with correct repo





To see docker image history:



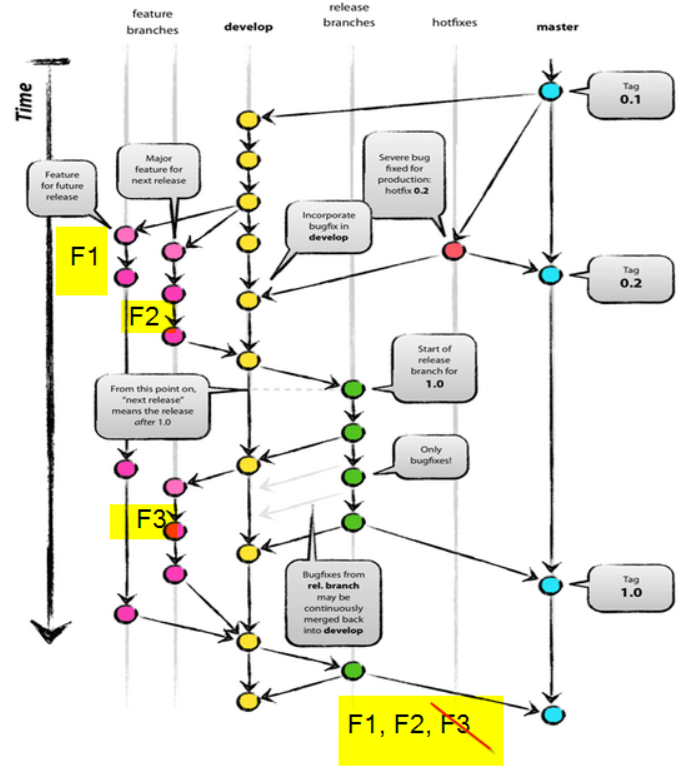


**Gitblit:**

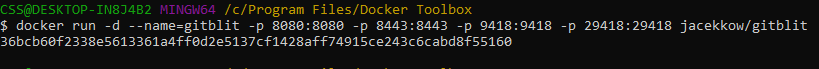
<http://gitblit.com/>

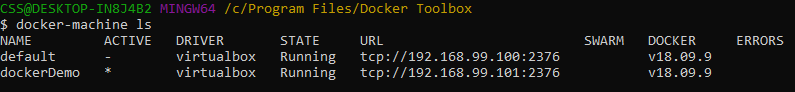
<https://hub.docker.com/r/jacekkow/gitblit>

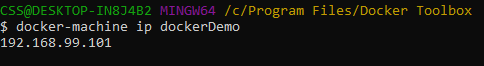
<https://nvie.com/posts/a-successful-git-branching-model/> - refer this for explanation





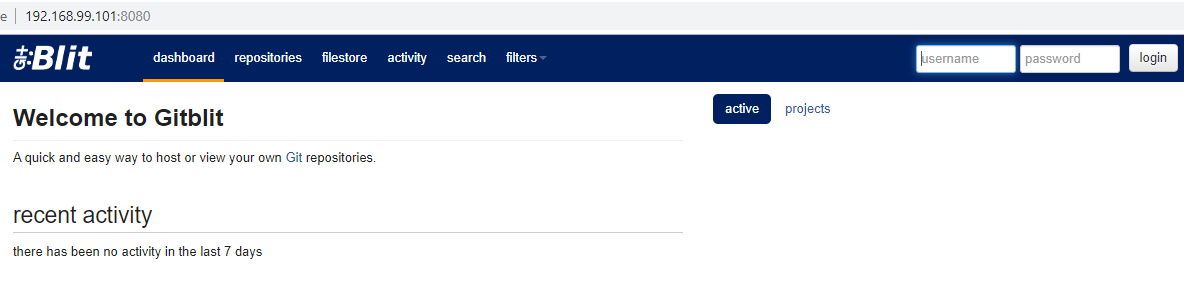


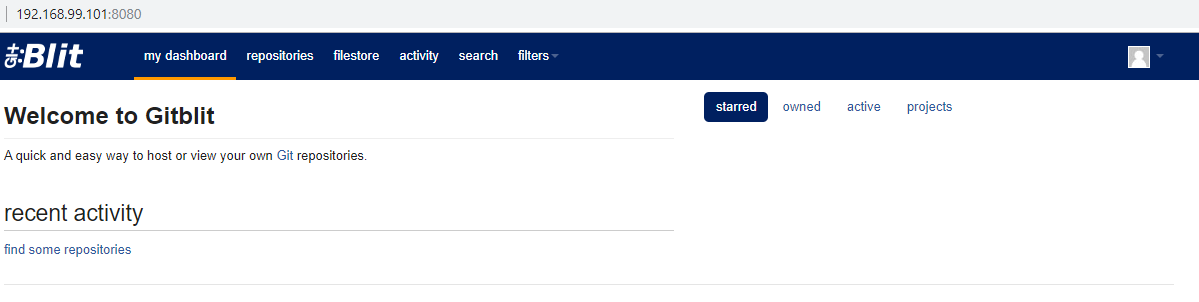




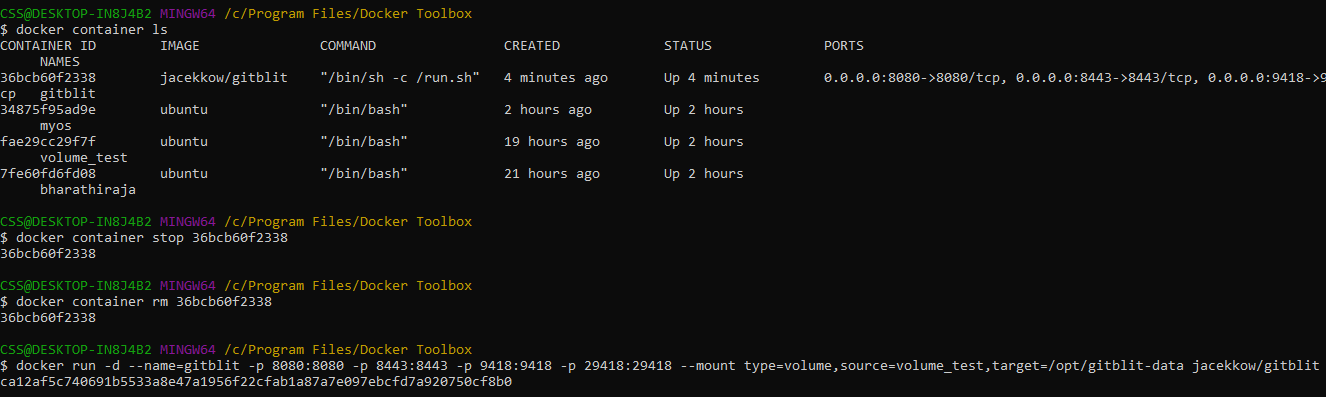
Username: admin

Password: admin

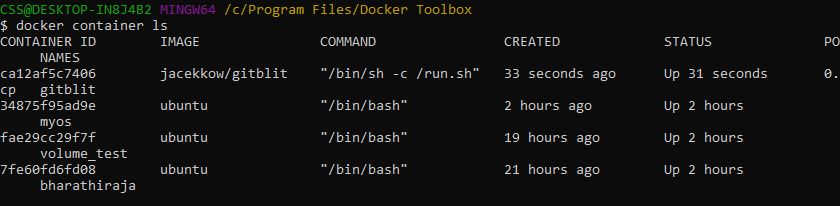


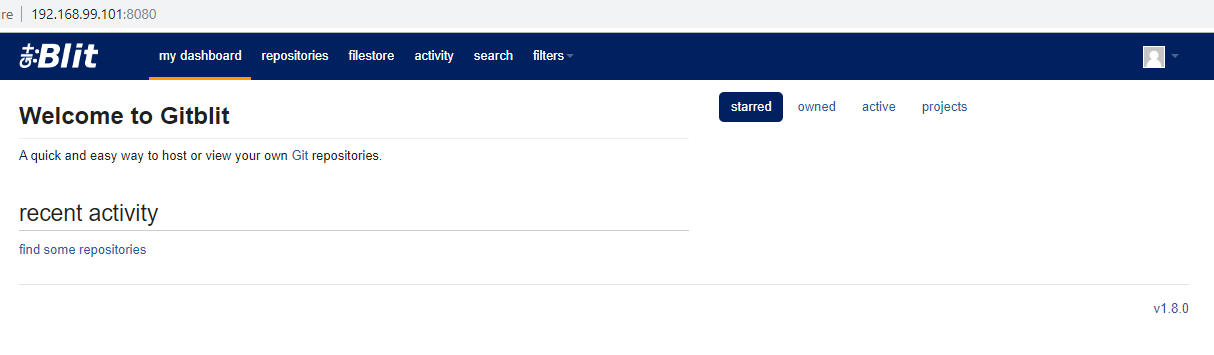


If don’t want to vanish all the data in gitblit use volume command

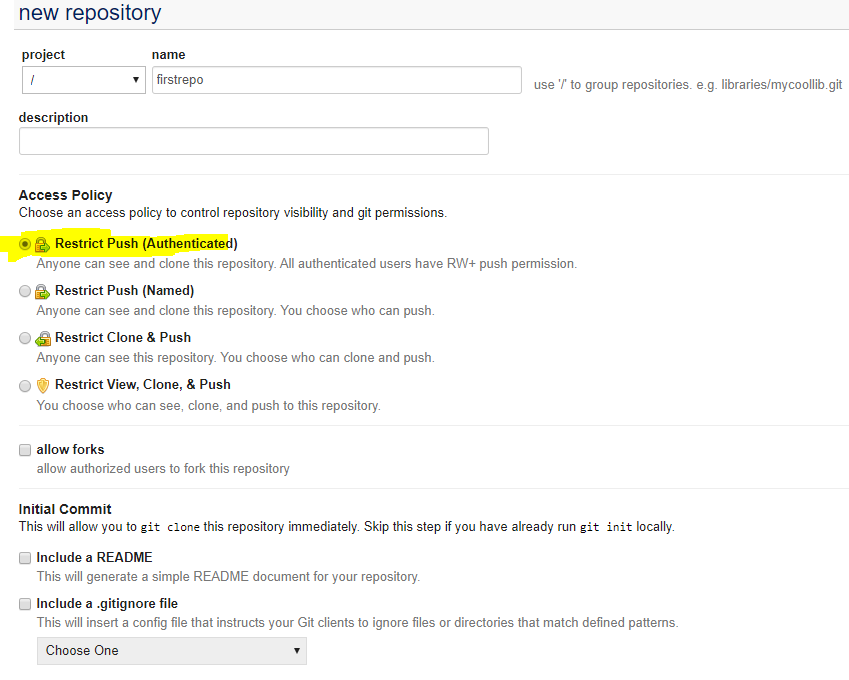


$ docker run -d --name=gitblit -p 8080:8080 -p 8443:8443 -p 9418:9418 -p 29418:29418 --mount type=volume,source=volume\_test,target=/opt/gitblit-data jacekkow/gitblit





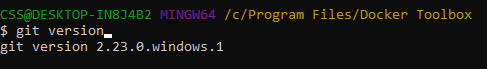
**Create Repository:**

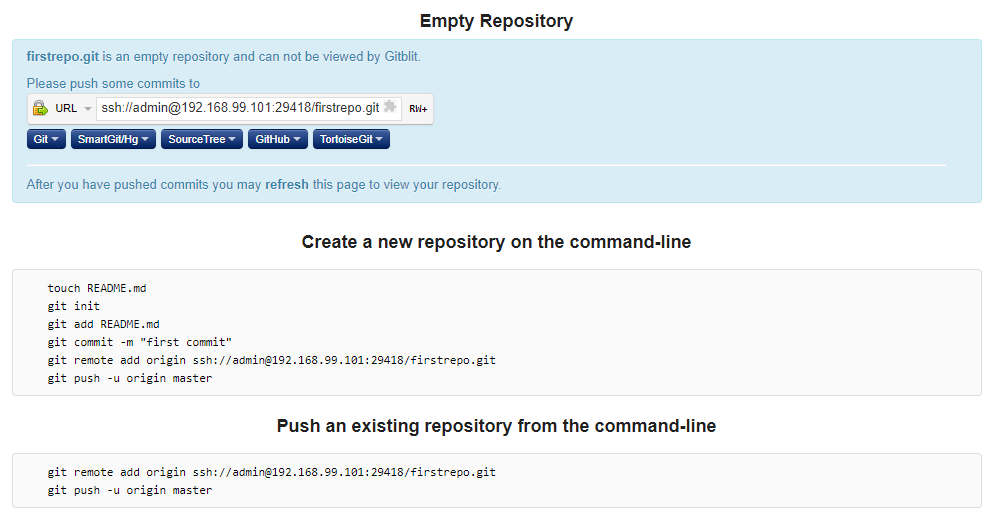


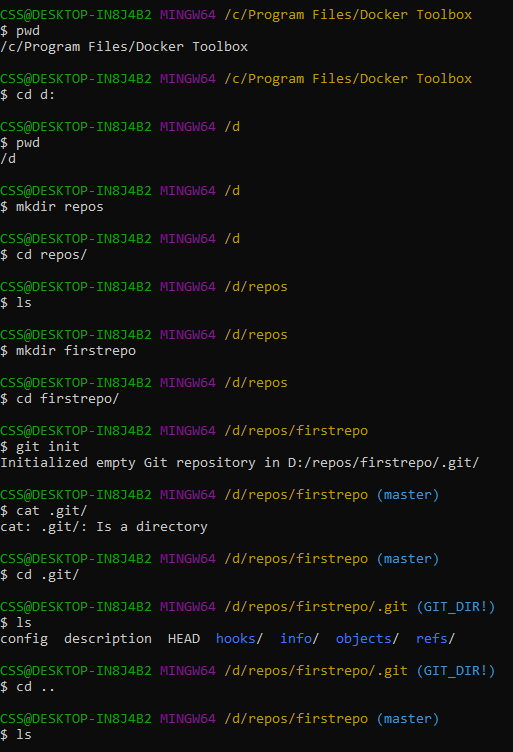
**Existing Git to GitFlow**

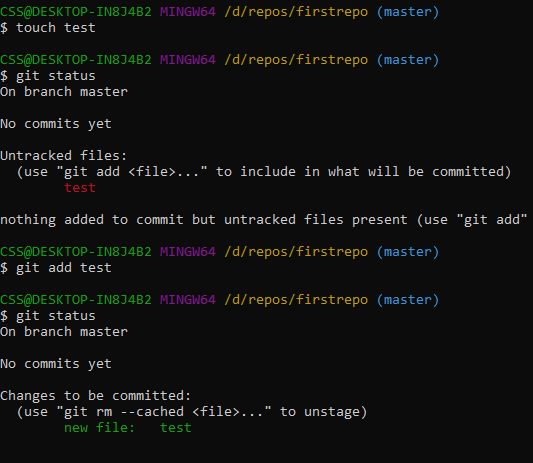
<https://github.com/nvie/gitflow/wiki/FAQ>

head – start of the git, details about the repo



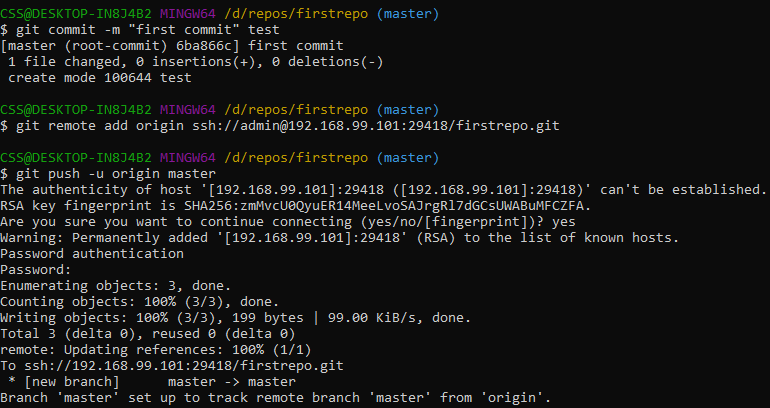












git config –system –edit

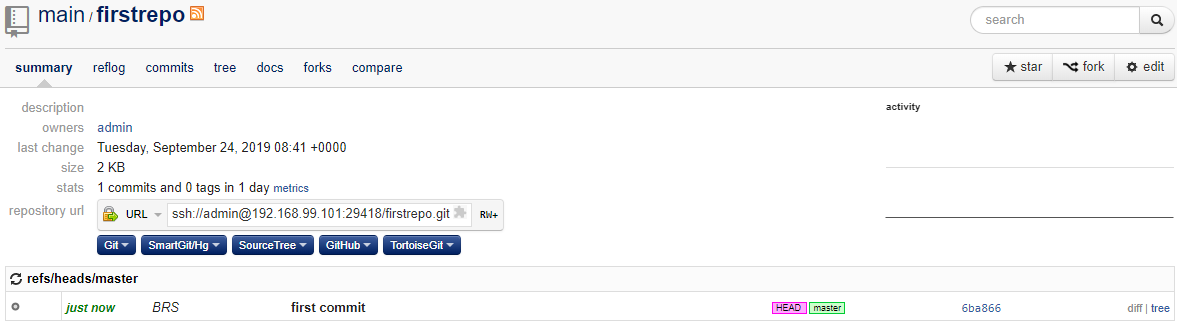
git config –local - -edit

git config –worktree –edit

--system – any body can login

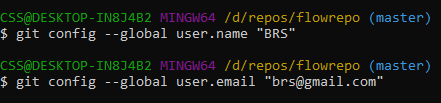
--local – only for this git

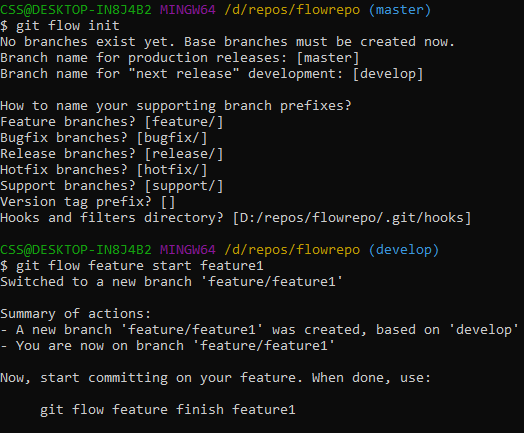
--worktree - branch

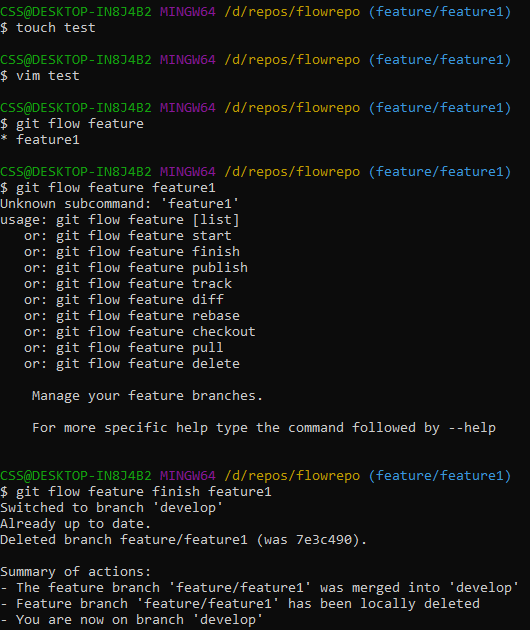


<http://mjit.in/post.php?slug=basic-git-commands>





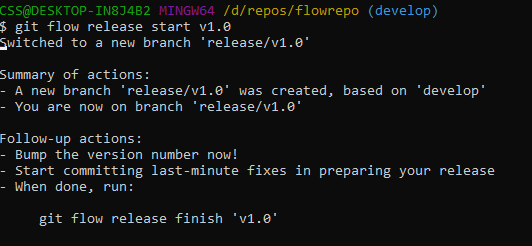


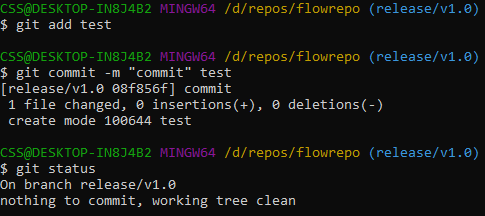


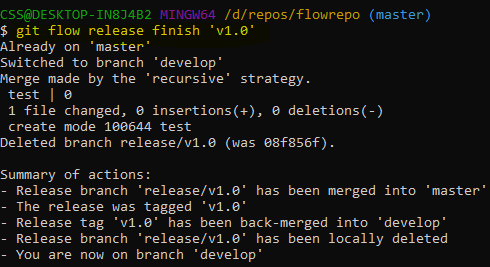
merging feature1 to develop

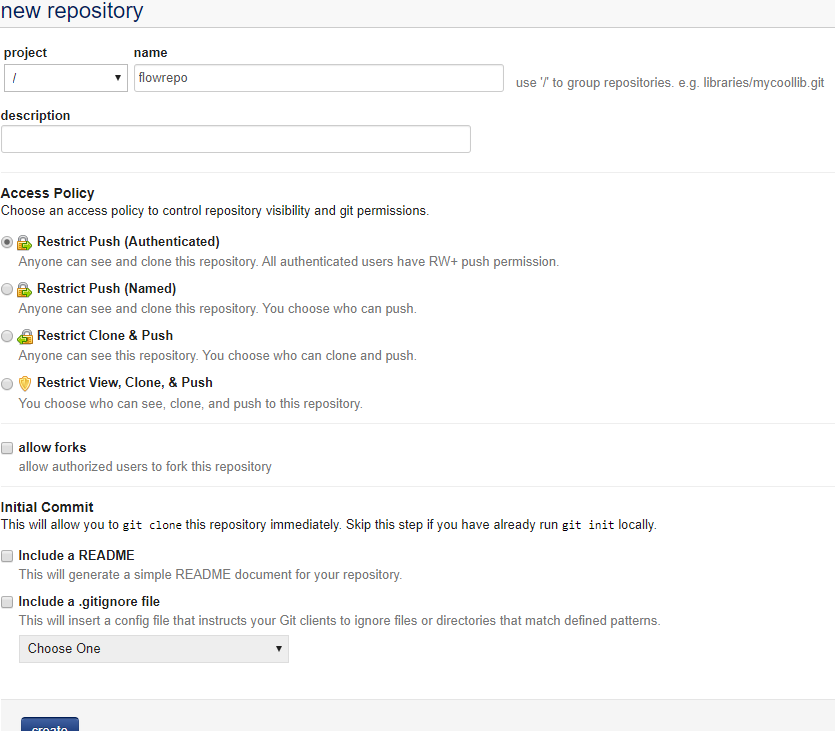
feature1 branch will get deleted from remote repo

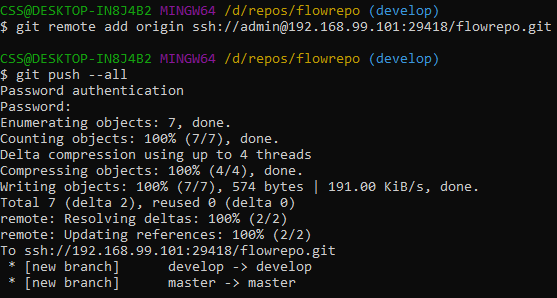
git flow publish – other fellow also can work in the same file when we do publish



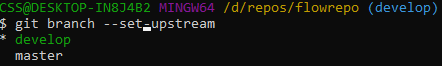


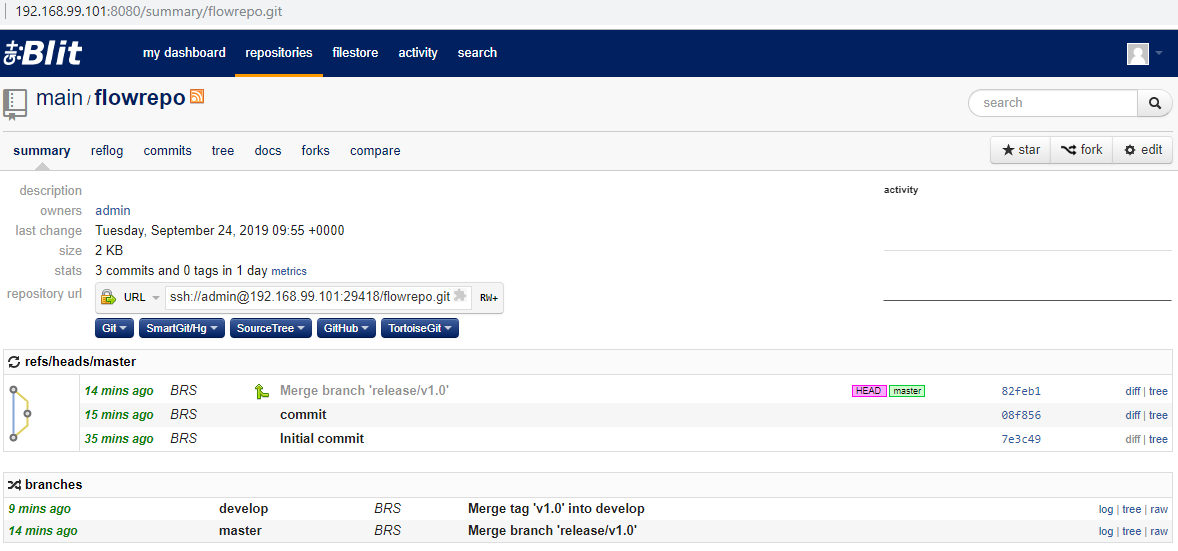






If it is require do it the below command





Docker Captain website:

<https://collabnix.com/>

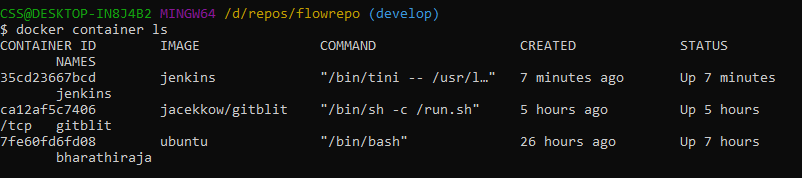
**Pipeline Tool – Jenkins – CI Tool**

* Build - project
* Plugin – external connection –

Ways to install Jenkins:

1. Using docker – will go ahead with this approach
2. Using Binaries
3. Using war

<https://hub.docker.com/_/jenkins>



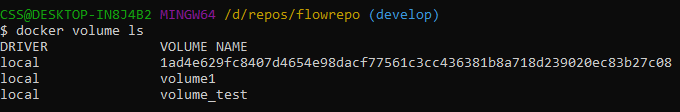


Keep only highlighted container, rest stop it and then use prune command



docker container stop fae29cc29f7f

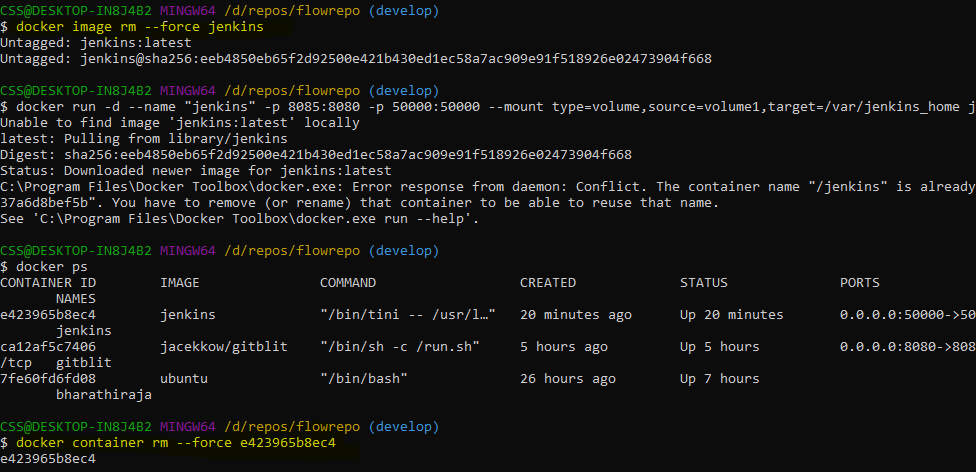
docker container prune



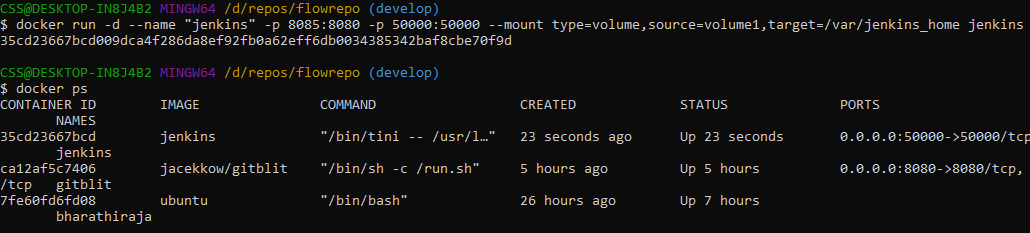
I have made wrong here, used 9090 port which is already nginx using so I have to remove it

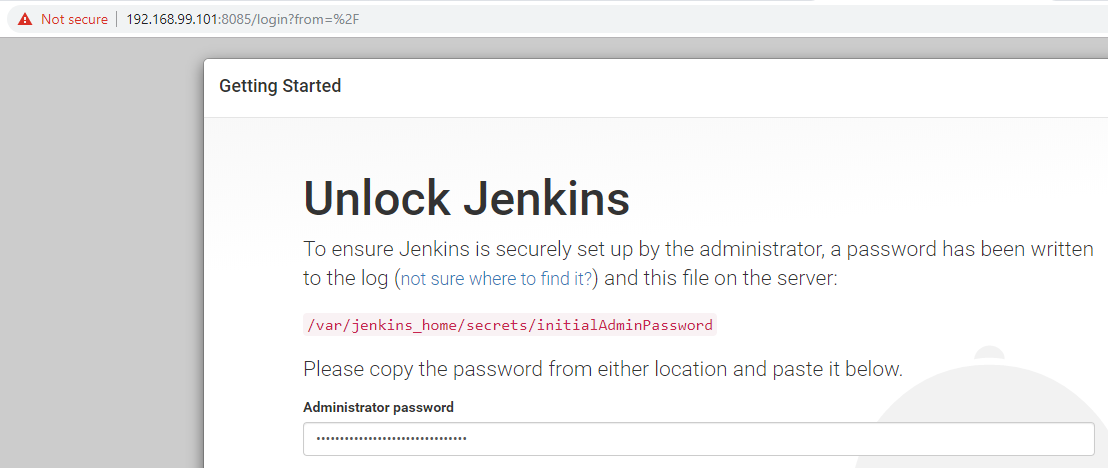


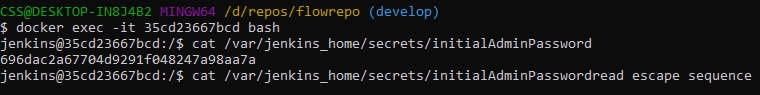
Shouldn’t remove docker image, should remove container



docker run -d --name "jenkins" -p 8085:8080 -p 50000:50000 --mount type=volume,source=volume1,target=/var/jenkins\_home Jenkins

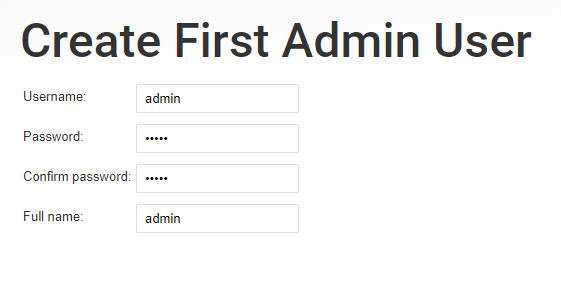


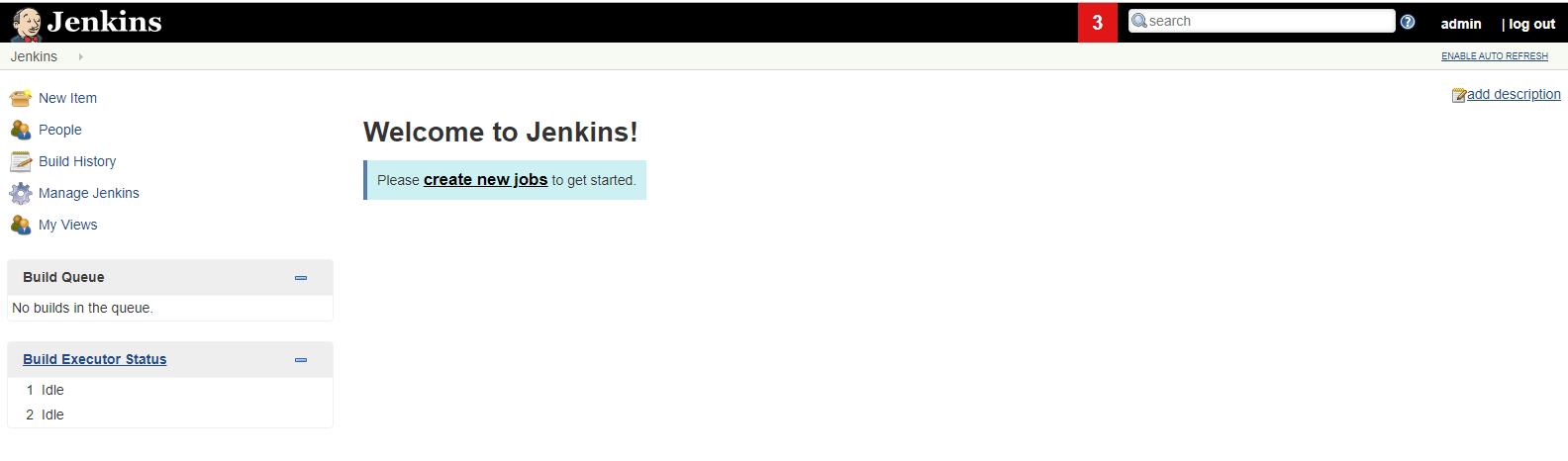




696dac2a67704d9291f048247a98aa7a







**CD Tool – Ansible**

* IT Automation
* Config management tools
* Simple CMD tools

**1. Puppet** – DSL – Robust – Cannot crash easily

**2. Chef** - DSL & Ruby – it is similar like puppet

**3. Saltstack** - Python & YML + Jinja

**4. Ansible** - Python & YML + Jinja (recovered the agent issue)

First 3 are Agent based

Fourth 4 – Agentless

Agent will capture the details of all target nodes and send it to server.

**Ansible: CD Tool -**Infra management

1. Dynamic Inventory – for cloud based
2. Smart inventory – patching only the server which are not patched based on particular time

Automate the infrastructure or commands, creating docker using yml file. We don’t want swarm and kuberneets

Ansible gallery tool – arrangements / structure of files maintains

Task

Play

Play book

**Installing ansible:**

docker pull ansible/centos7-ansible

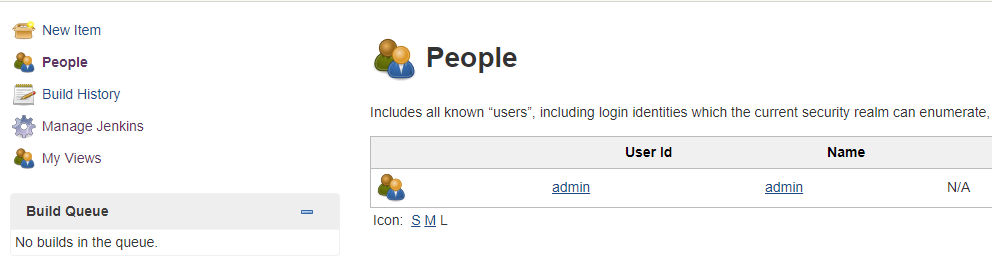


<https://hub.docker.com/r/ansible/centos7-ansible>

**Day 3**

**Jenkins Continuation:**

Adding new user we need to install user management plugins



$ docker-machine ls

$ docker-machine start dockerDemo

$ docker-machine env dockerDemo

$ eval $("C:\Program Files\Docker Toolbox\docker-machine.exe" env dockerDemo)

$ docker-machine ls

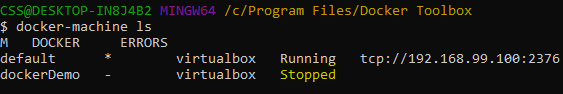
$ docker image ls –a

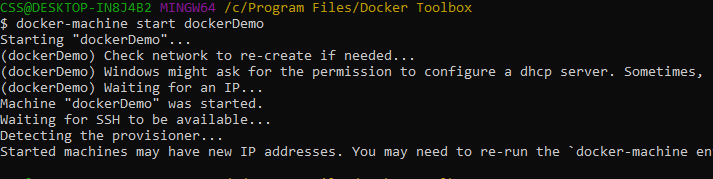
$ docker ps

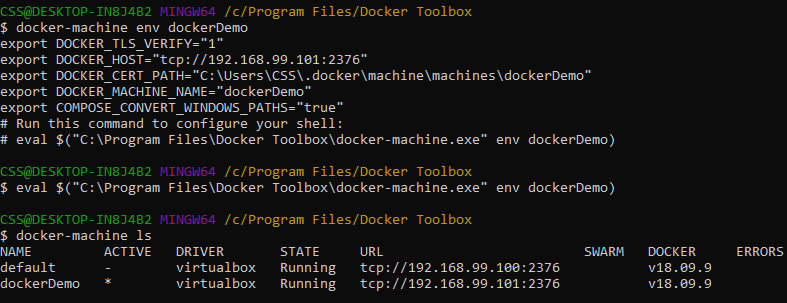
$ docker container ls –a

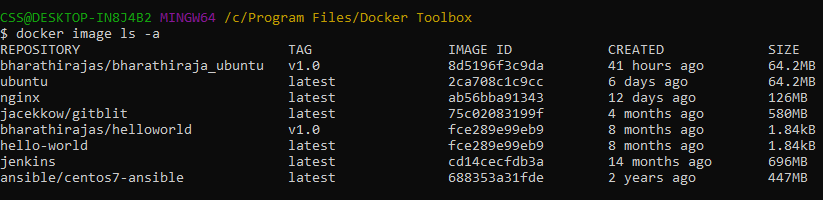
$ docker container start 35cd23667bcd

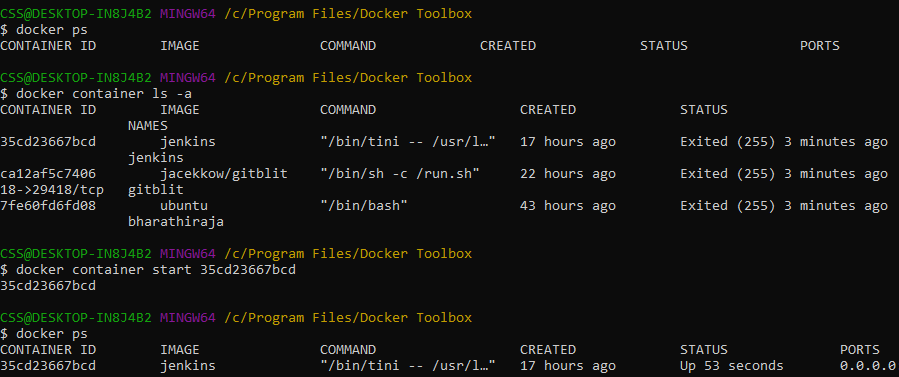
$ docker ps

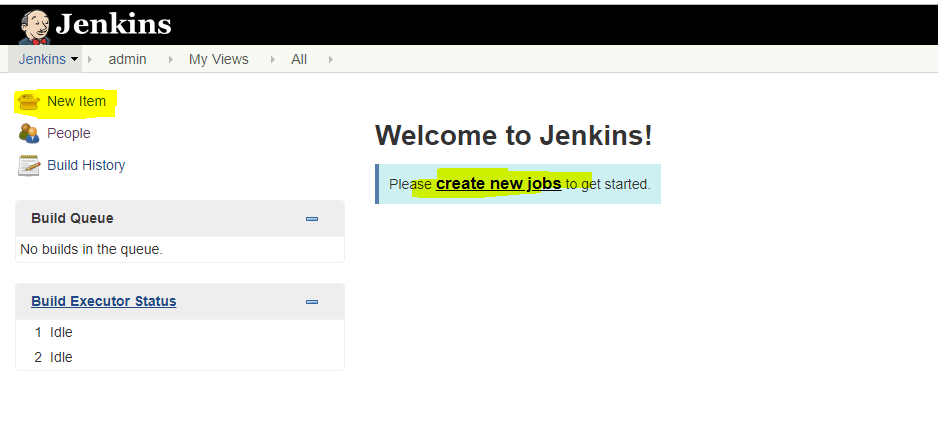


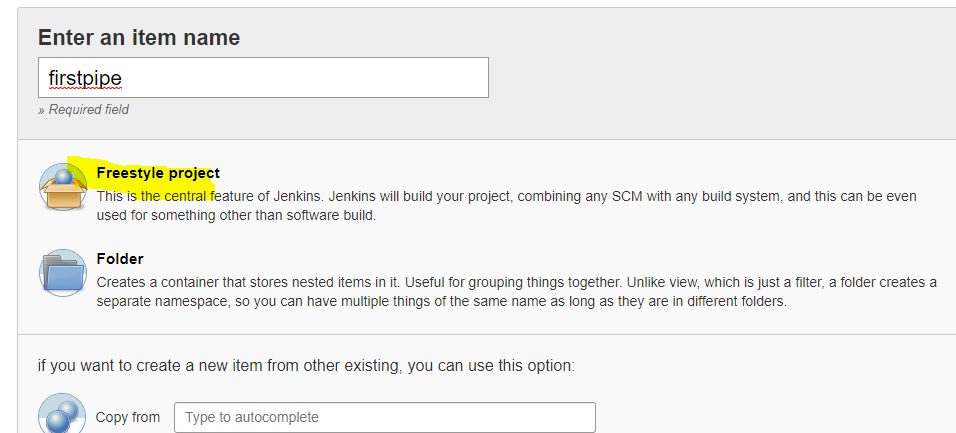


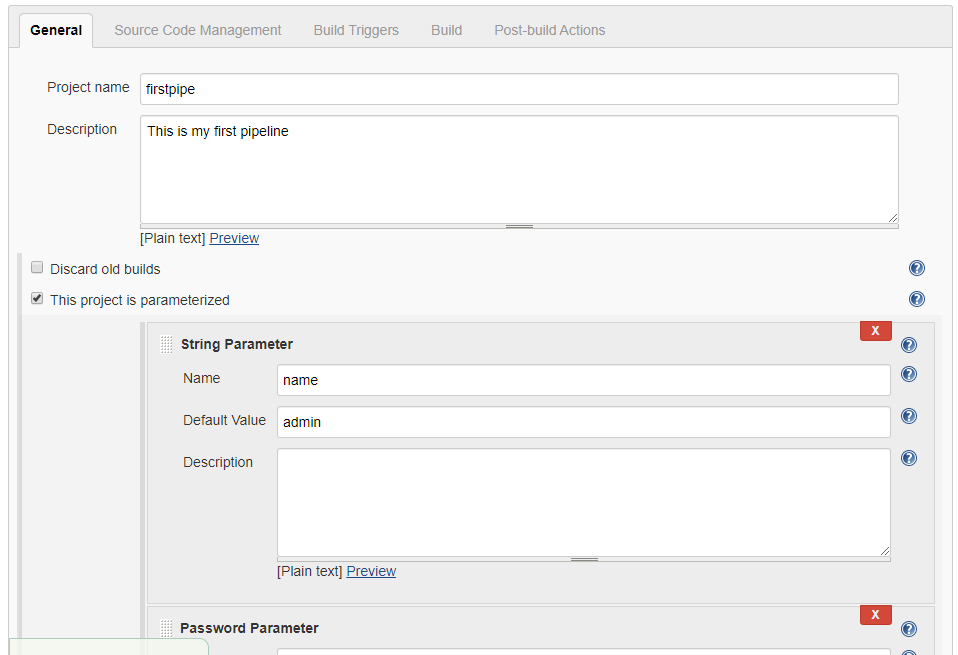


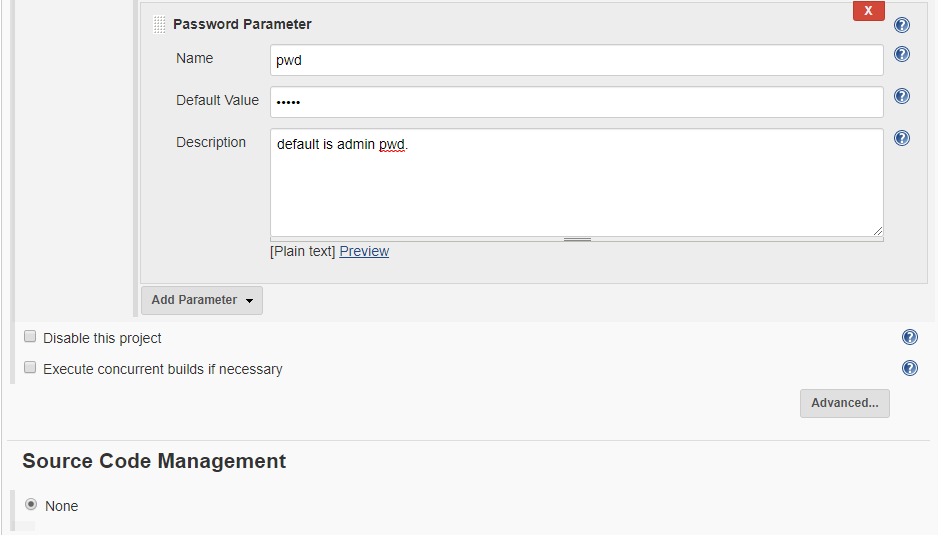


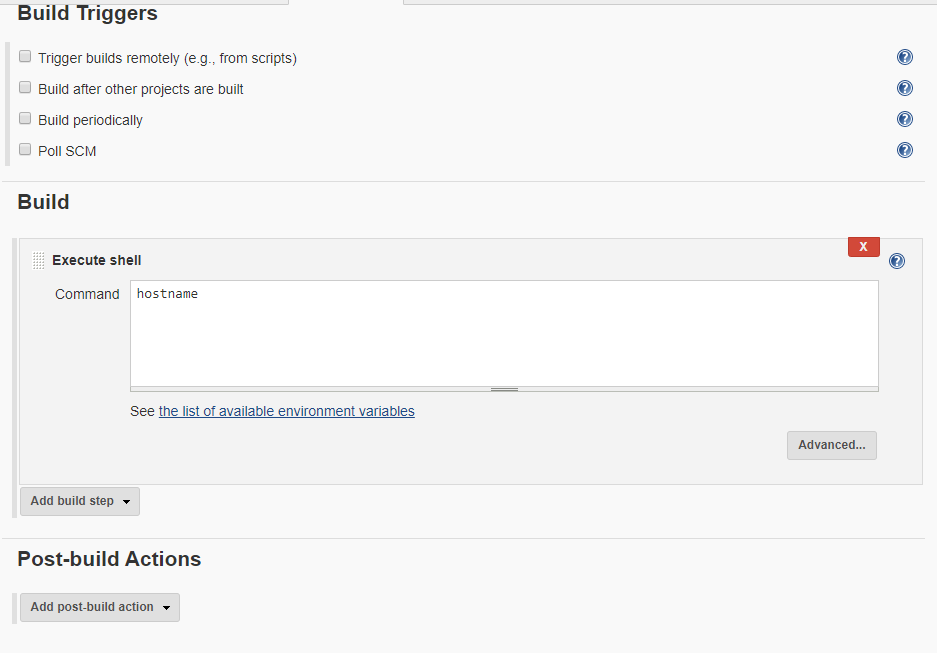


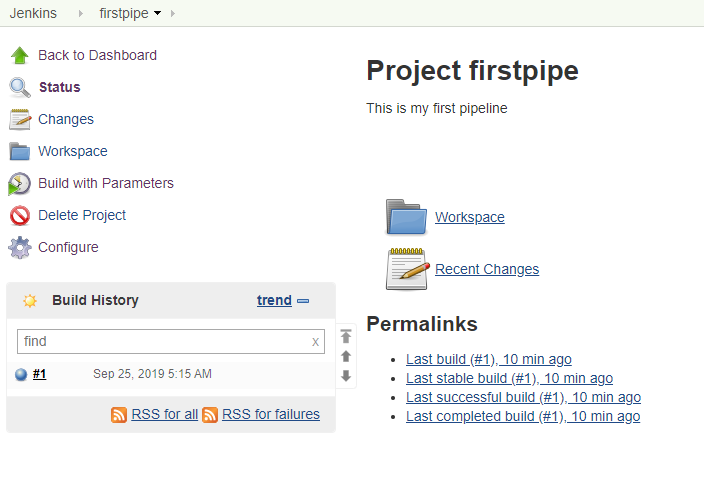












**Steps to Installing other version of Jenkins**

$ docker pull jenkins/Jenkins

$ docker container prune

$ docker run -d --name "jenkins2" -p 8086:8080 -p 50001:50000 --mount type=volume,source=volume\_jenkins,target=/var/jenkins\_home Jenkins

$ docker ps

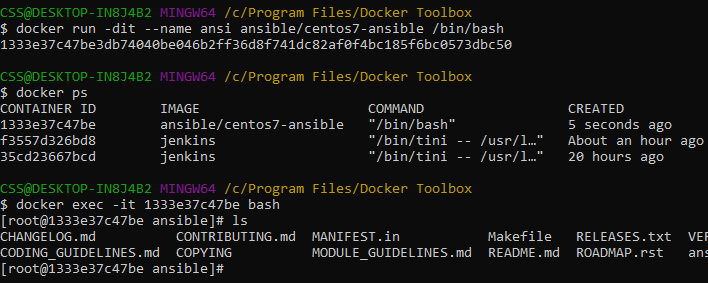
$ docker exec -it f3557d326bd8 bash

**Ansible Continuation:**

$ docker pull ansible/centos7-ansible

$ docker run -dit --name ansi ansible/centos7-ansible /bin/bash

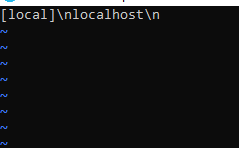
Or docker attach ansi



Provisioning capabilities

Control machine – In ansible point of view, everything is control machine

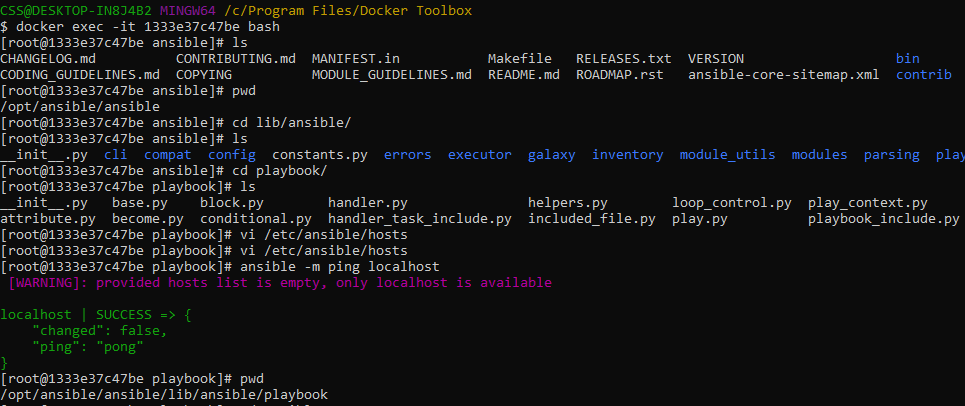
Inventory – store the list of servers – cat /etc/ansible/host

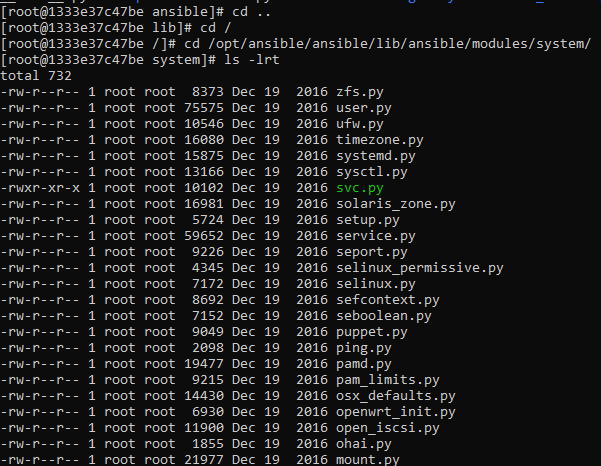


Local –group

Localhost – server or variable

Remove above content in host file and ping





--- play book

* Play

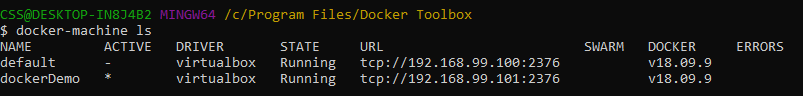
hosts – keyword – start of the play

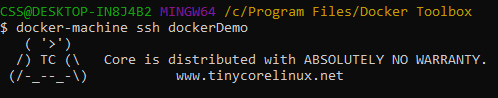
task

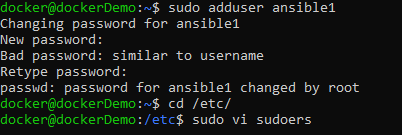
* Include \_vars

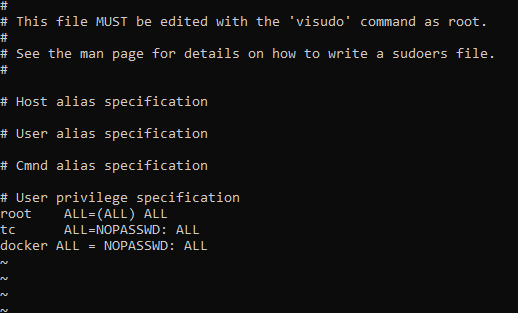
4 space – use this in yml file

Tab is 5 space

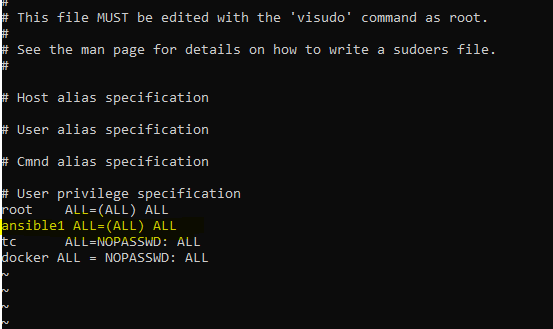


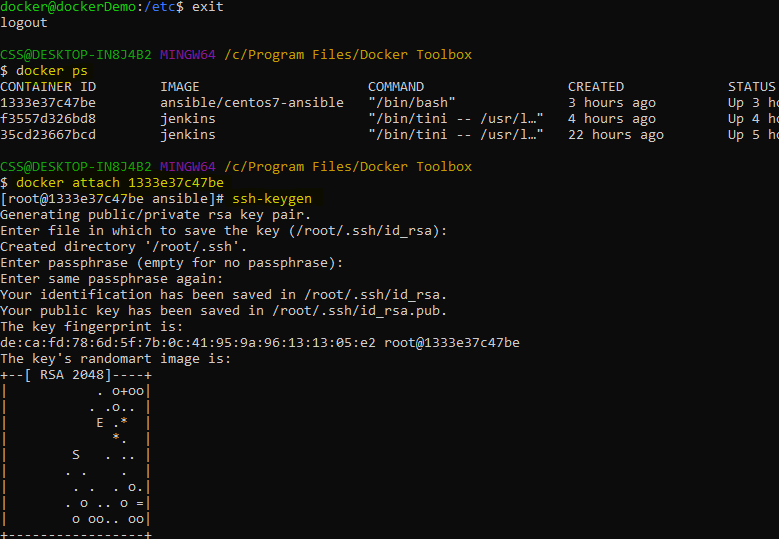


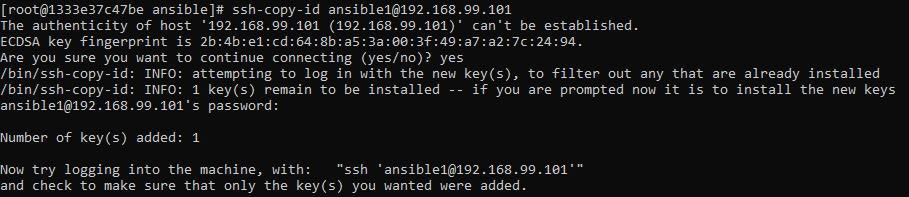


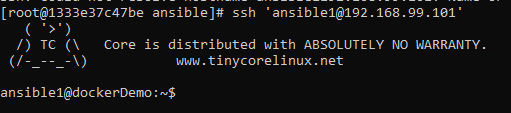


Add ansible user in to that file



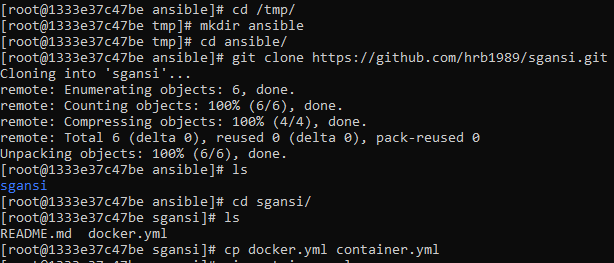




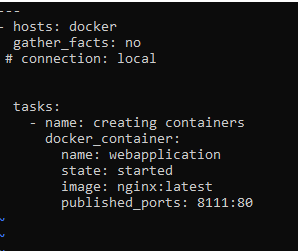




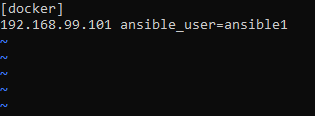
**Nginx:**



Keep only below lines







Docker ps

Pwd

Cd /var/lib/awx/mjit/projects

Ls

Cd docker-ansible

Ls

Cat docker.yml

Ls

Cat docker\_create\_empty\_container.yml

Docker ps

Ansible-playbook docker\_create\_empty\_container.yml

Ansible-playbook –vvv docker\_create\_empty\_container.yml

Docker ps –a

Vim docker create\_empty\_container.yml

Docker rm webapplication

Docker ps –a

Ansible-playbook docker\_create\_empty\_container.yml

**Nagios:**

Control any of the infrastructure and monitor the infrastructure

It is consider everything is host

check\_service is plugin

Agent for linux based - NRPEE

Windows based - NT client

**Disadvantage:**

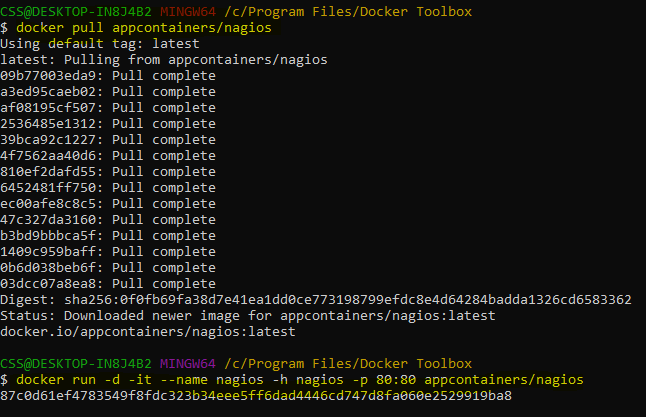
Network manage will not happen here

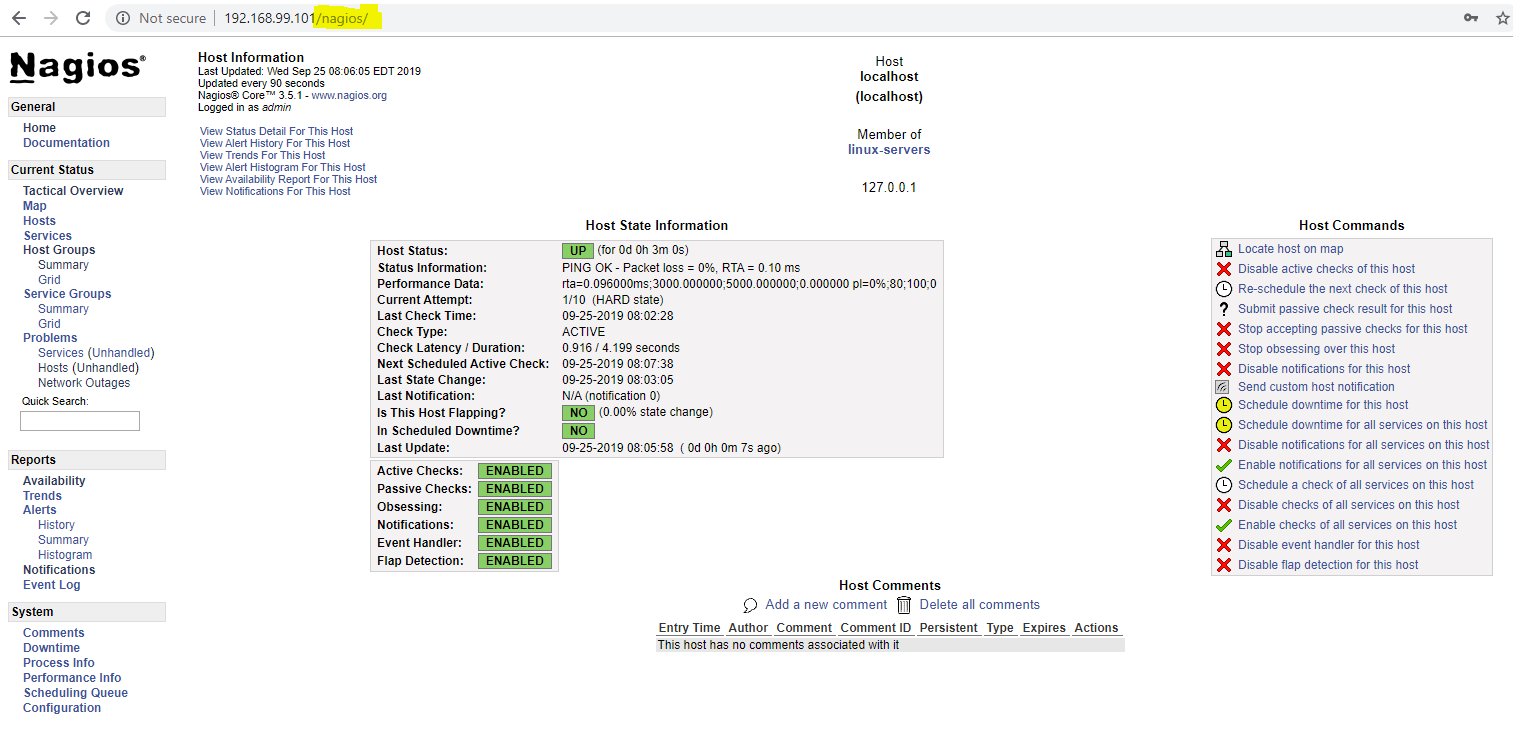
Dashboard little clumsy

Nagios XL – Enterprise

Nagios Score – Open source

<https://hub.docker.com/r/appcontainers/nagios>





**Competitor:** Solarwinds

**Extras commands:**

history >> /d/day2

$ history | tail -20

$ docker image save -o E:\Share\jenkins\_jenkins.tar jenkins/Jenkins

esc dd- delete line in vi

esc u

Azure - Microsoft

AWS – Amazon

Terraform –it’s a template

<https://www.meetup.com/meetup-group-YuWxljJl/events/mgtctqyzmblc/>