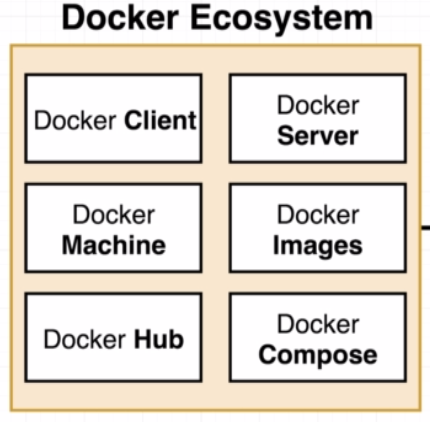
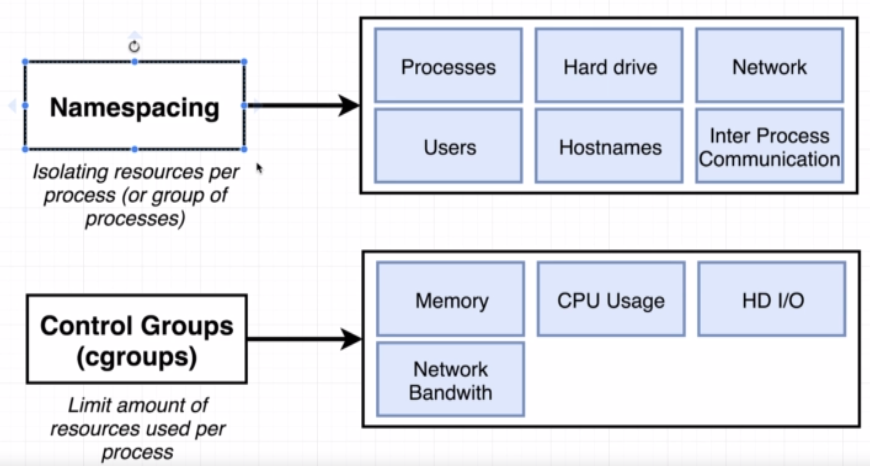
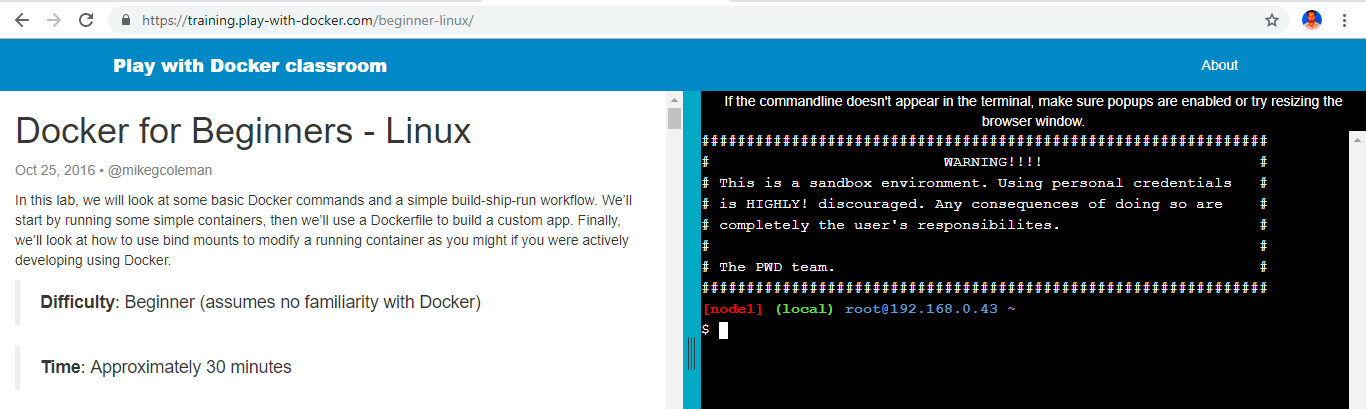
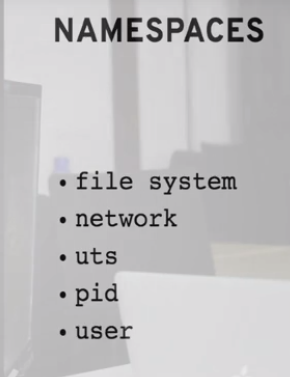
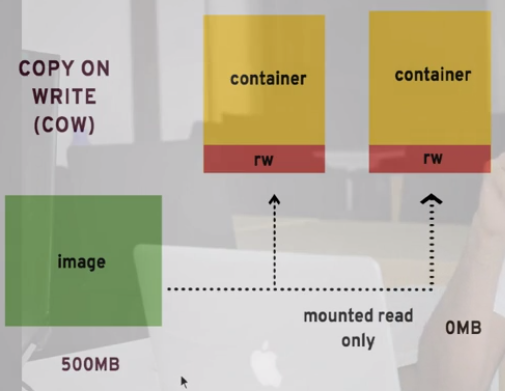
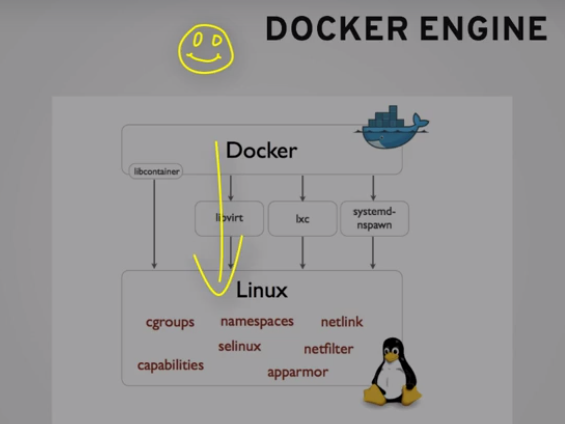
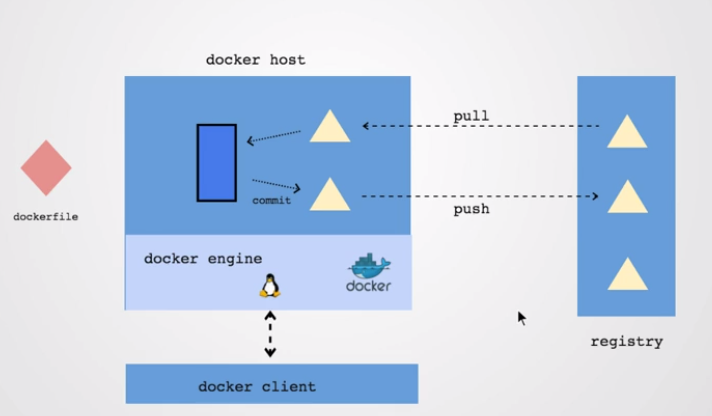
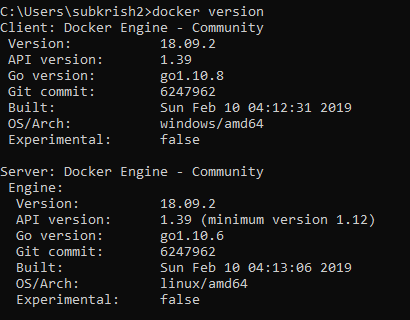
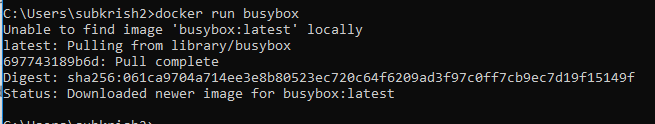
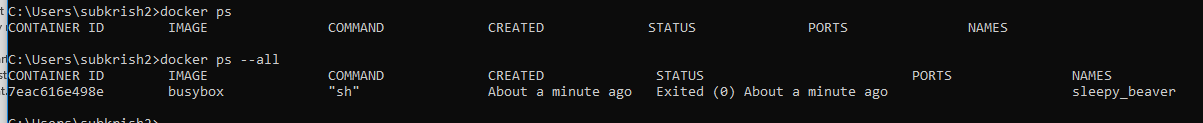
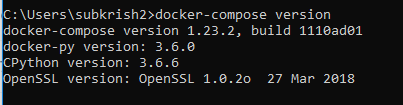
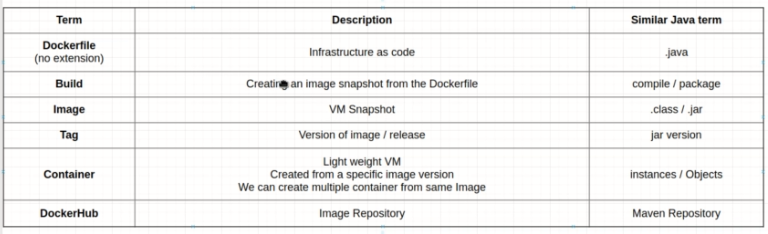
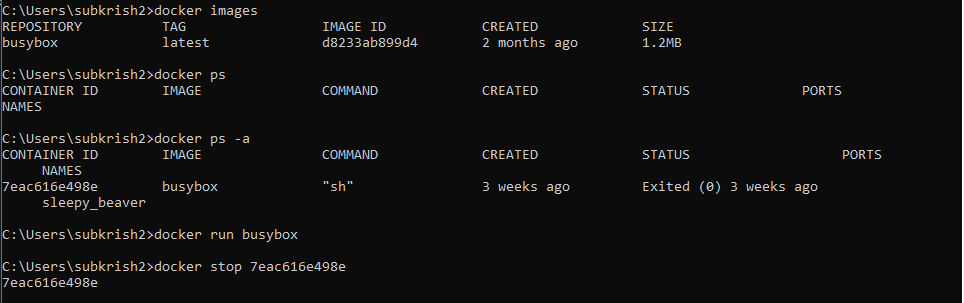
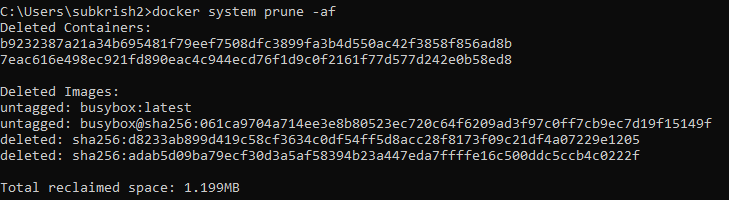
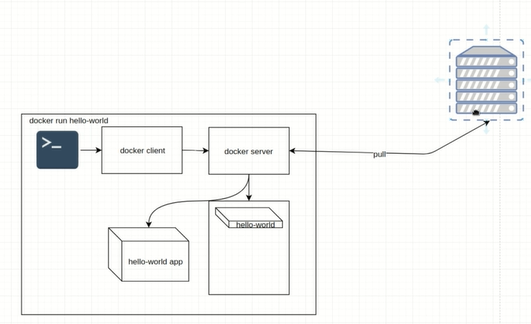
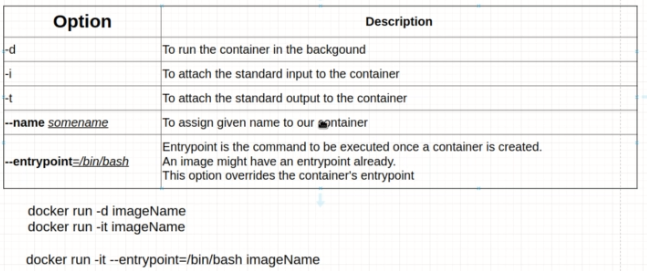
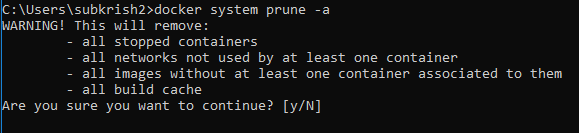
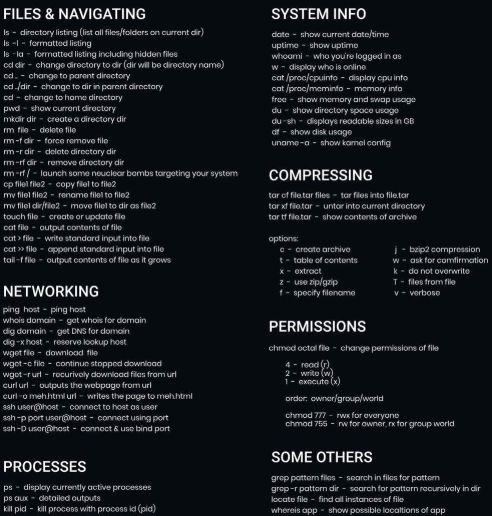
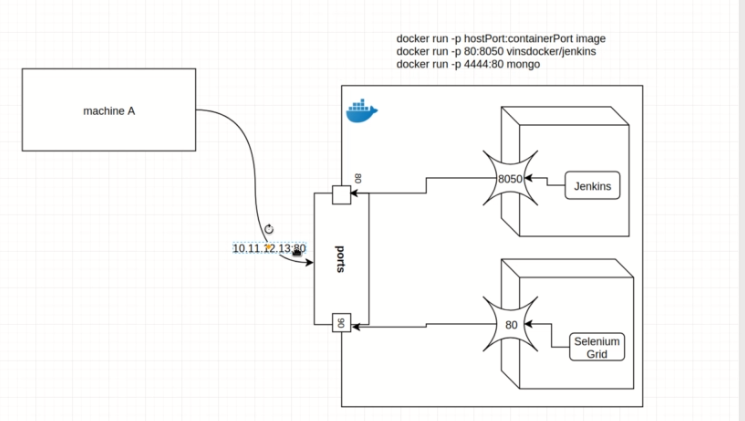
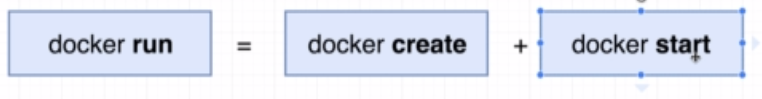
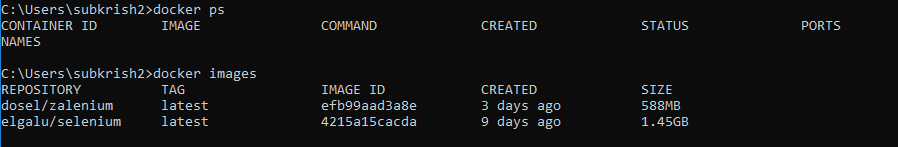
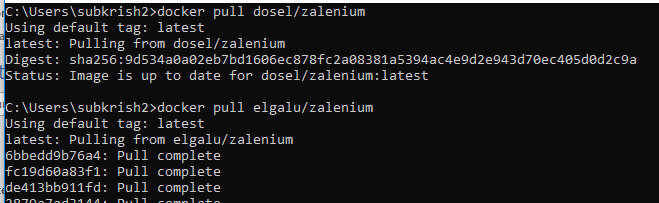
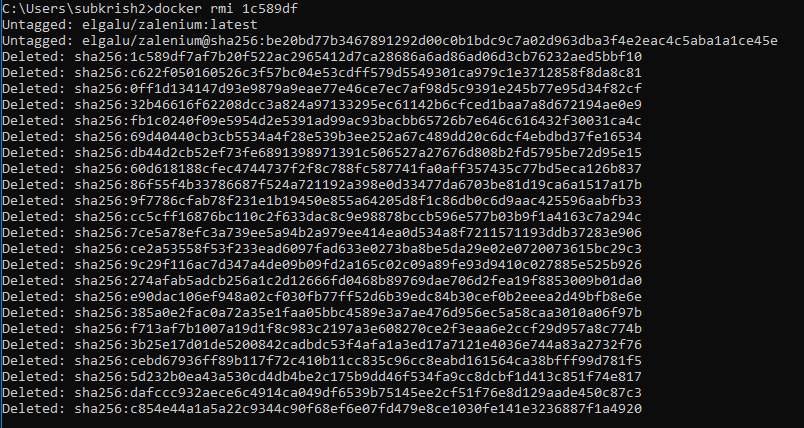
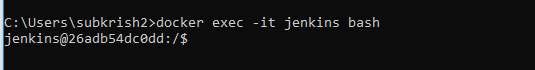
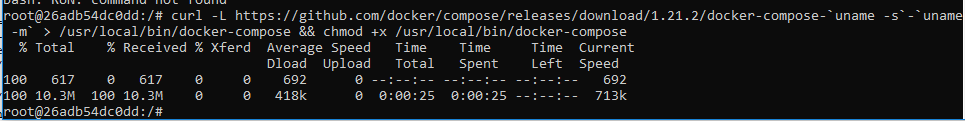
1. Docker is a platform or ecosystem around creating and running containers  
   
2. 
3. Docker start and docker run difference -> docker run shows all logs, docker start –a does not show logs
4. 
5. Difference between a VM and a container is, VM by itelf must have a Virtual BIOS, kernel and launches all the processes then the app to be started, but in case of a container, it has a app alone which are just one or two processes to be running on a container hence its light weight than a VM
6. Inside a container we have a set of PIDs  
   
7. Docker uses overlay file Systems as it supports layers
8. 
9. 
10. Docker engine port: 2376
11. 
12. Container is an instance of an image(is a single file with all dependency and config to run a program)
13. Docker Client(Docker CLI) --🡪 Docker Server(Docker daemon)
14. Verify docker version  
    
15. For the first time when there is no image for the container, on executing the command docker run, image gets downloaded as below  
    
16. 
17. Docker run = docker create+ docker start
18. Basic docker commands:  
    docker ps  
    docker ps –a  
    docker run <image\_name>  
    docker start <container\_name>  
    docker stop <container\_name>  
    docker rm <container\_name>  
    docker rmi <image\_name>  
    docker pull <image\_name>  
    docker run –d <container\_name> -> to run container in background  
    docker run it <image\_name>  
    docker run –name <use\_defined> <image\_name>

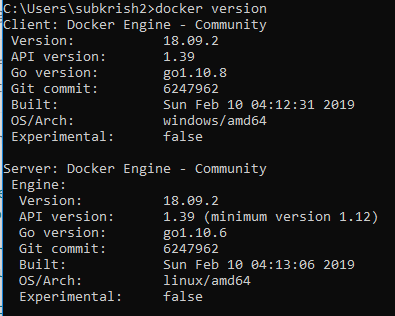
docker run timer  
docker run –p <host\_port>:<container\_port> <image\_name>:<custom\_name>

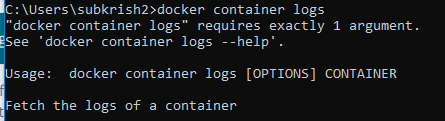
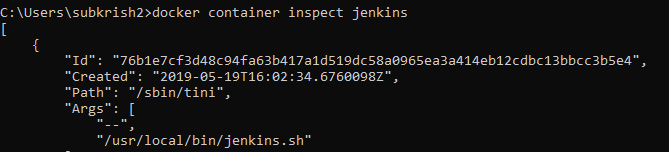
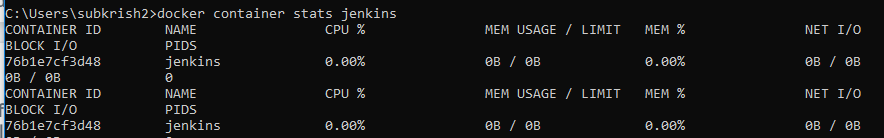
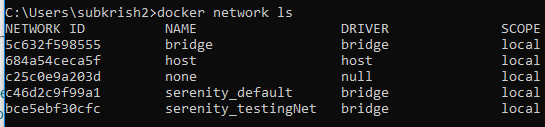
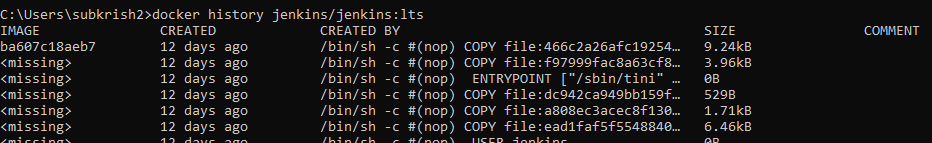
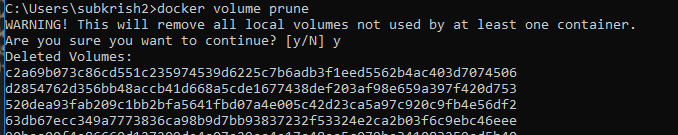
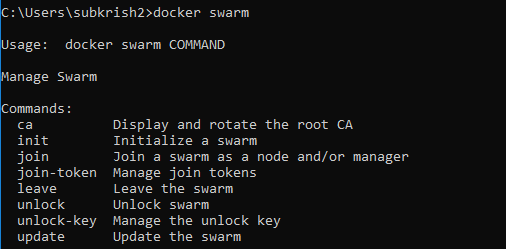
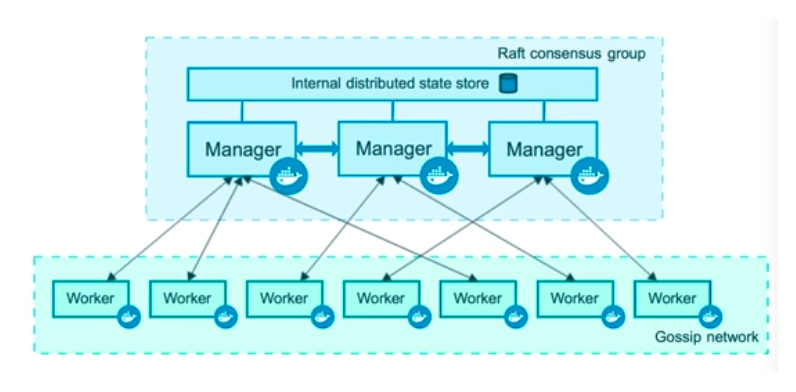
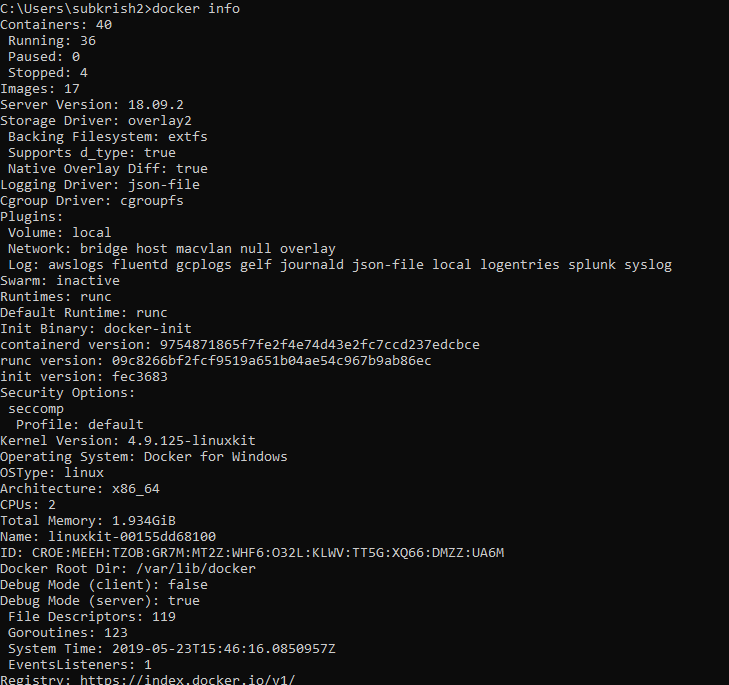
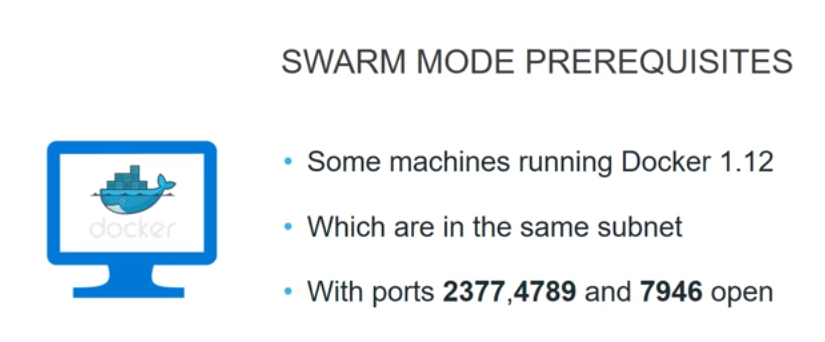
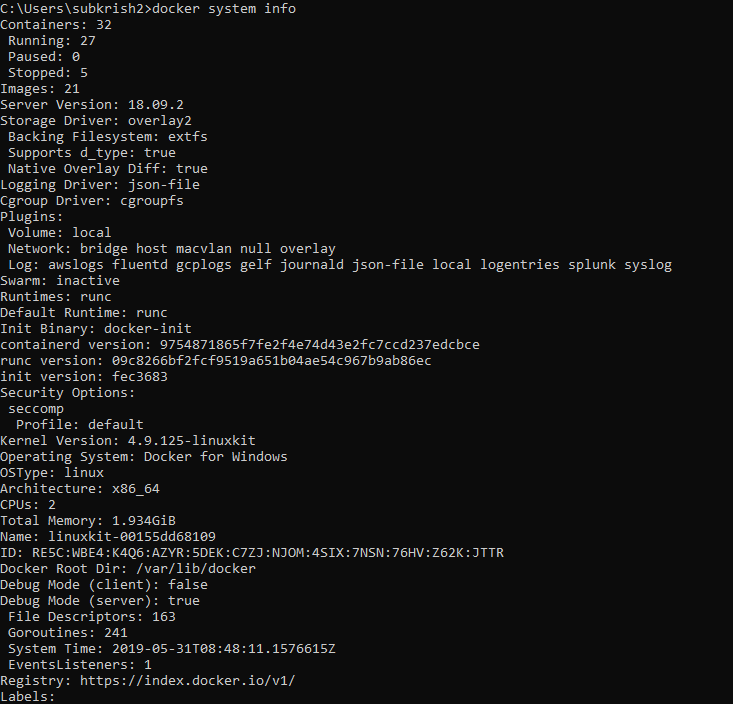
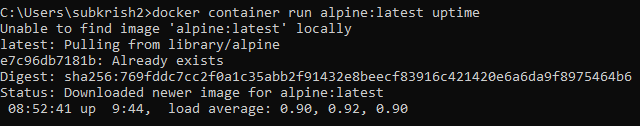
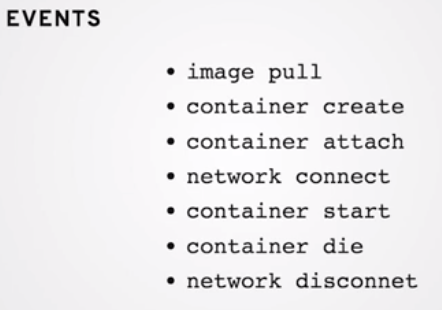
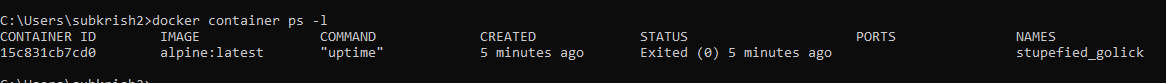
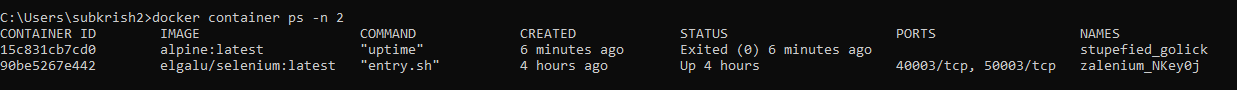
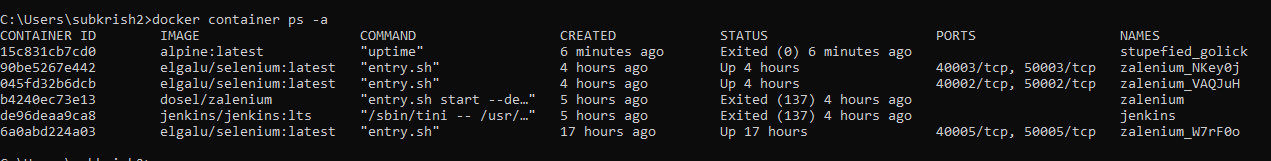
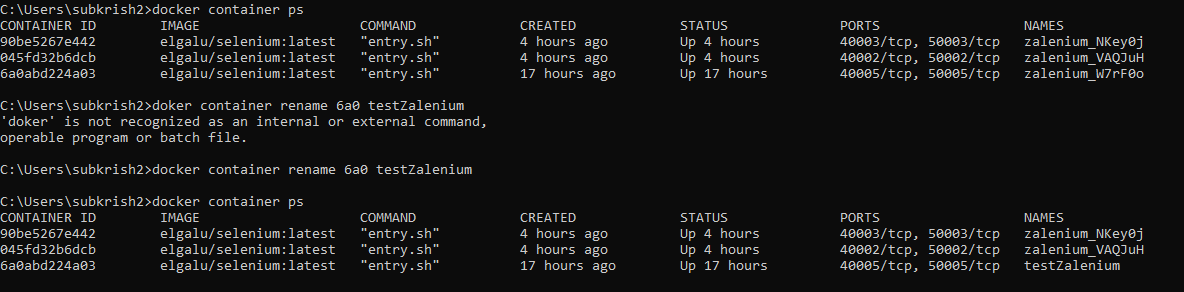
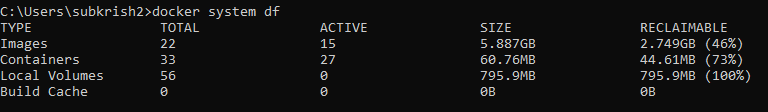
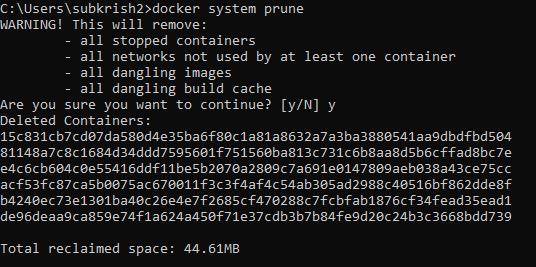
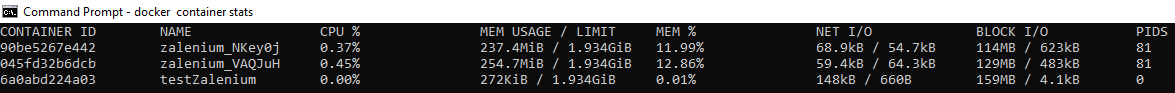
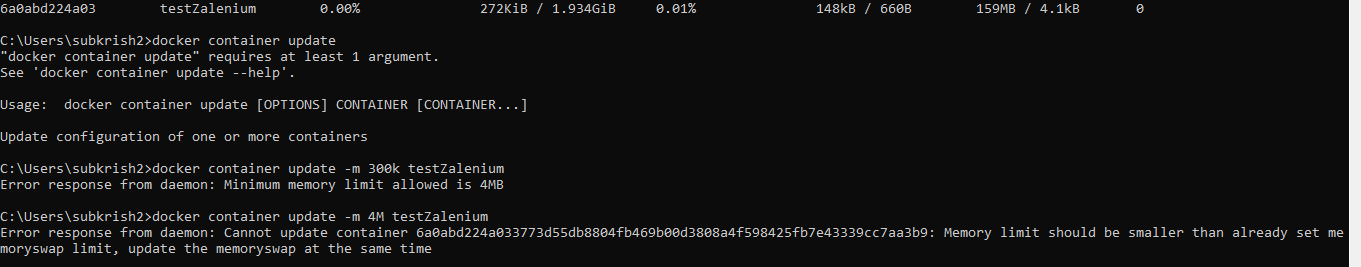
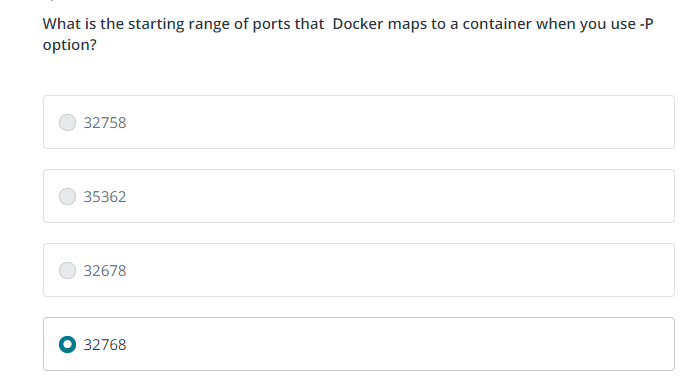
docker run -v

1. Verify docker-compose  
     
   Docker-compose is used to connect two containers and start multiple docker containers
2. 
3. 
4. Deletes containers  
   
5. 
6. Docker image format:  
   
7. 
8. 
9. 
10. 
11. Docker run <image\_name> <override\_command>
12. Docker container life cycle:  
    
13. 
14. 
15. 
16. Docker run –it <image\_name>  
    i-> allows to provide input to container
17. Docker run –p 5000<localPort>:8080<containerport> image\_name

<http://testnblog.com/ui-automation-framework-on-docker/>  
<https://code-maze.com/ci-jenkins-docker/>

1. 
2. 
3. 
4. 
5. 

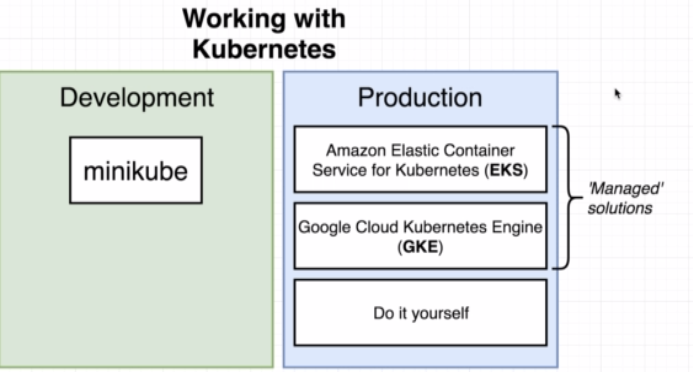
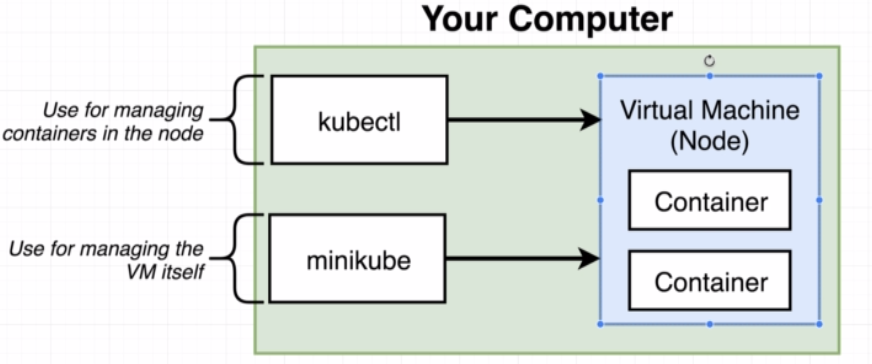
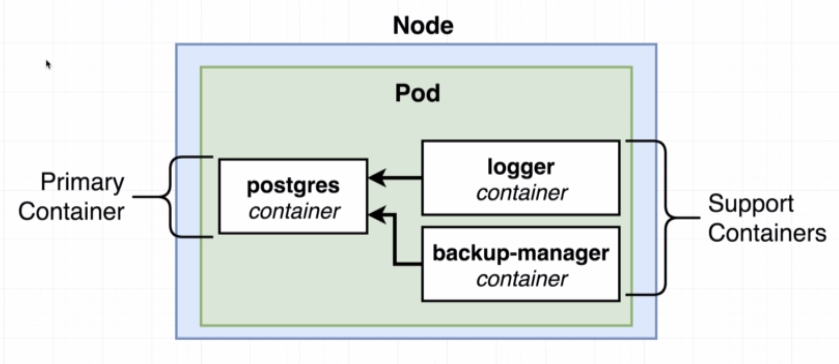
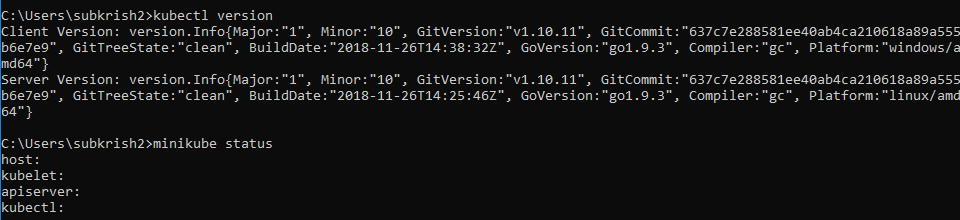
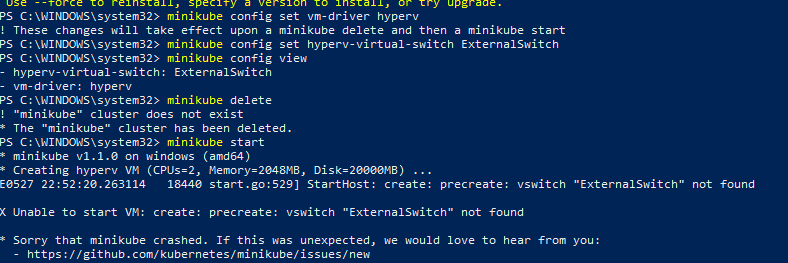
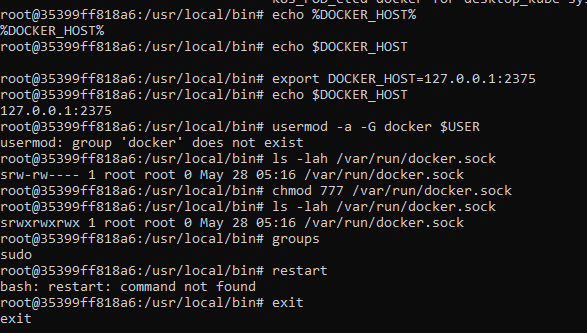


1. Image ->is a binary, library and source course   
   Container -> is an instance of image, we can have many containers running on the same image
2. 
3. Containers Vs VMs: Container is a process on the host:
4. 
5. 
6. 
7. Image is an ordered collection of root file system changes and the corresponding execution parameters for use within a container runtime
8. Image Layers:   
   
9. 
10. Docker force remove a container   
    
11. 
12. Docker stop <container\_ID> -> sends SIGTERM command
13. Docker kill <container\_ID> sends SIGKILL command
14. 
15. 
16. 
17. 
18. Find docker0 IP address:  
    
19. 
20. 
21. 
22.   
    displays last run container
23.   
    displays last two containers run
24.   
    displays all containers
25. Docker container run –idt 🡪 d is to run the container in detach mode
26. Rename a container  
    
27. Show disk usage  
    
28. Remove unused containers  
    
29. Docker container inspect <container\_name>
30. 
31.   
    Set memory limit for docker container
32. 
33. 
34. 

Kubernetes

1. System for running may different containers with different images

on many different machines

1. 
2. 
3. Pod -> allows set of containers to be grouped  
   
4. 
5. 
6. 
7. <http://www.testautomationguru.com/embedding-zalenium-live-execution-in-jenkins/>
8. <https://bulldogjob.com/articles/726-exploring-jenkins-pipelines-a-simple-delivery-flow>