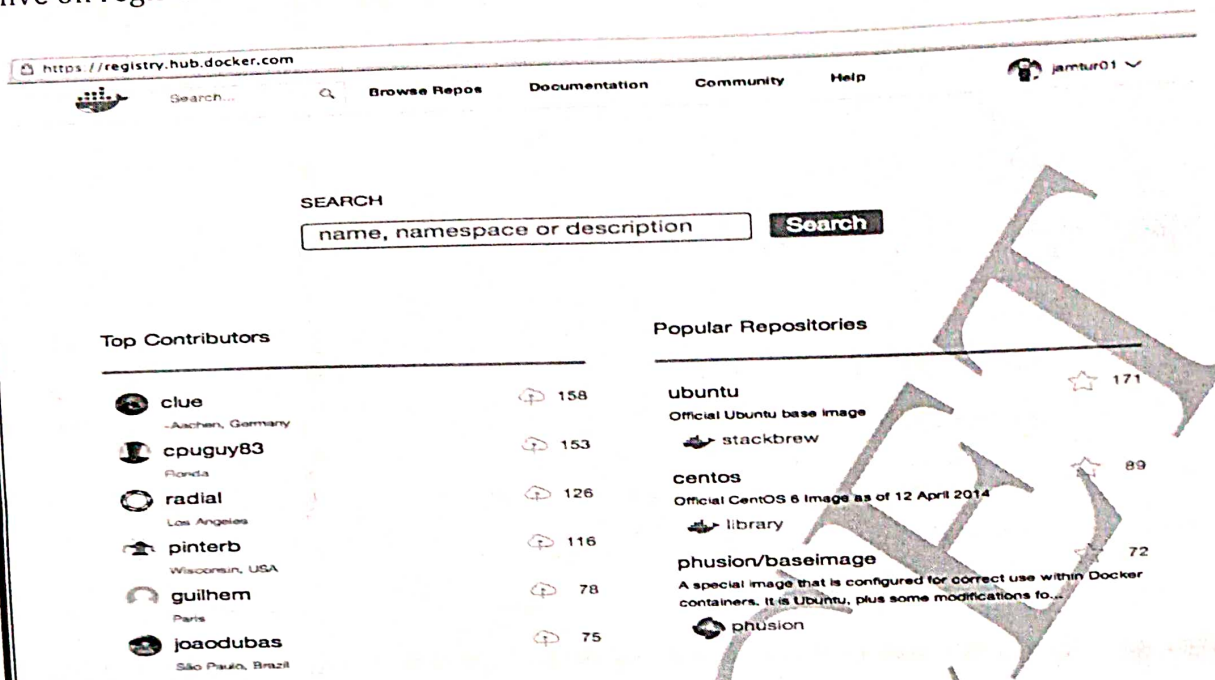
That image was downloaded from a repository. Images live inside repositories, and repositories live on registries. The default registry is the public registry man- aged by Docker, Inc., Docker Hub.



Each repository can contain multiple images (e.g., the ubuntu repository contains images for Ubuntu 12.04, 12.10, 13.04, 13.10, and 14.04). Let's get the rest of the images in the ubuntu repository now.

```
$ sudo docker pull ubuntu
Pulling repository ubuntu
c4ff7513909d: Pulling dependent layers
3db9c44f4520: Pulling dependent layers
75204fdb260b: Pulling dependent layers
...
```

Here we've used the docker pull command to pull down the entire contents of the ubuntu repository.

Let's see what our docker images command reveals now.

```
$ sudo docker images
REPOSITORY TAG      IMAGE ID      CREATED      VIRTUAL SIZE
ubuntu      13.10   5e019ab7bf6d 2 weeks ago 180 MB ubuntu
saucy   5e019ab7bf6d 2 weeks ago 180 MB ubuntu      12.04
74fe38d11401 2 weeks ago 209.6 MB ubuntu      precise
74fe38d11401 2 weeks ago 209.6 MB ubuntu      12.10
a7cf8ae4e998 2 weeks ago 171.3 MB ubuntu      quantal
a7cf8ae4e998 2 weeks ago 171.3 MB ubuntu      14.04
99ec81b80c55 2 weeks ago 266 MB ubuntu      latest
c4ff7513909d 6 days ago  225.4 MB ubuntu      trusty
99ec81b80c55 2 weeks ago 266 MB ubuntu      raring
316b678ddf48 2 weeks ago 169.4 MB ubuntu      13.04
316b678ddf48 2 weeks ago 169.4 MB ubuntu      10.04
3db9c44f4520 3 weeks ago 183 MB ubuntu      lucid
3db9c44f4520 3 weeks ago 183 MB
```

You can see we've now got a series of ubuntu images. We can see that the ubuntu image is actually a series of images collected under a single repository.

We can refer to a specific image inside a repository by suffixing the repository name with a colon and a tag name, for example:

Running a tagged Docker image

```
$ sudo docker run -t -i --name new_container ubuntu:12.04 /bin/↩
  bash
root@79e36bff89b4:/#
```

This launches a container from the ubuntu:12.04 image, which is an Ubuntu 12.04 operating system. We can also see that some images with the same ID (see image ID 74fe38d11401) are tagged more than once. Image ID 74fe38d11401 is actu- ally tagged both 12.04 and precise: the version number and code name for that Ubuntu release, respectively.

Pulling images

When we run a container from images with the `docker run` command, if the image isn't present locally already then Docker will download it from the Docker Hub. By default, if you don't specify a specific tag, Docker will download the `latest` tag, for example:

Docker run and the default latest tag

```
$ sudo docker run -t -i --name next_container ubuntu
/bin/bash root@23a42cee91c3:/#
```

will download the `ubuntu:latest` image if it isnt already present on the host. Alternatively, we can use the `docker pull` command to pull images down our- selves. Using `docker pull` saves us some time launching a container from a new image. Let's see that now by pulling down the `fedora` base image.

Pulling the fedora image

```
$ sudo docker pull fedora
Pulling repository fedora
5cc9e91966f7: Download complete b7de3133ff98:
Download complete
511136ea3c5a: Download complete ef52fb1fe610:
Download complete
```

Let's see this new image on our Docker host using the `docker images` command. This time, however, let's narrow our review of the images to only the `fedora` images. To do so, we can specify the image name after the `docker images` command.

g 4.8: Viewing the fedora image

```
$ sudo docker images fedora
```

Here, we've searched the Docker Hub for the term puppet. It'll search images and return:

- Repository names
- Image descriptions
- Stars - these measure the popularity of an image
- Official - an image managed by the upstream developer (e.g., the fedora image managed by the Fedora team)
- Automated - an image built by the Docker Hub's Automated Build process

Let's pull down one of these images.

g 4.11: Pulling down the jamtur01/puppetmaster image

```
$ sudo docker pull jamtur01/puppetmaster
```

This will pull down the jamtur01/puppetmaster image (which, by the way, con- tains a pre-installed Puppet master).

We can then use this image to build a new container. Let's do that now using the docker run command again.

g 4.12: Creating a Docker container from the Puppet master image

```
$ sudo docker run -i -t jamtur01/puppetmaster /bin/bash
root@4655dee672d3:/# facter
architecture => amd64 augeasversion
=> 1.2.0
. . .
root@4655dee672d3:/# puppet --version
3.4.3
```

You can see we've launched a new container from our jamtur01/puppetmaster im- age. We've launched the container interactively and told the container to run the Bash shell. Once inside the container's shell, we've run Facter (Puppet's inventory application), which was

pre-installed on our image. From inside the container, we've also run the puppet binary to confirm it is installed.

Docker Hub:

Docker Hub is a repository service and it is a cloud-based service where people push their Docker Container Images and also pull the Docker Container Images from the **Docker Hub** anytime or anywhere via the internet. It provides features such as you can push your images as private or public. Mainly DevOps team uses the Docker Hub. It is an open-source tool and freely available for all operating systems. It is like storage where we store the images and pull the images when it is required. When a person wants to push/pull images from the Docker Hub they must have a basic knowledge of Docker. Let us discuss the requirements of the Docker tool.

Docker is a tool nowadays enterprises adopting rapidly day by day. When a Developer team wants to share the project with all dependencies for testing then the developer can push their code on **Docker Hub** with all dependencies. Firstly create the **Images** and push the Image on Docker Hub. After that, the testing team will pull the same image from the Docker Hub eliminating the need for any type of file, software, or plugins for running the Image because the Developer team shares the image with all dependencies.

Docker Hub Features

- Storage, management, and sharing of images with others are made simple via Docker Hub.
- Docker Hub runs the necessary security checks on our images and generates a full report on any security flaws.
- Docker Hub can automate the processes like Continuous deployment and Continuous testing by triggering the Webhooks when the new image is pushed into Docker Hub.
- With the help of Docker Hub, we can manage the permission for the users, teams, and organizations.
- We can integrate Docker Hub into our tools like GitHub, Jenkins which makes workflows easy

Advantages of Docker Hub

- Docker Container Images are light in weight.
- We can push the images within a minute and with help of a command.
- It is a secure method and also provides a feature like pushing the private image or public image.

- Docker hub plays a very important role in industries as it becomes more popular day by day and it acts as a bridge between the developer team and the testing team.
- If a person wants to share their code, software any type of file for public use, you can just make the images public on the docker hub.

Creating First Repository in Docker Hub Using GUI

Step 1: We must open Docker Hub first, then select Create Repository.

Step 2: After that, we will be taken to a screen for configuring the repository, where we must choose the namespace, repository name, and optional description. In the visibility area, as indicated in the picture, there are two options: Public and Private. We can choose any of them depending on the type of organization you are in. If you chose Public, everyone will be able to push-pull and use the image because it will be accessible to everyone. If you select the private option, only those with access to that image can view and utilize it. it.

Step 3: At finally repository is created with the help of the Docker Commands we can push or pull the image.

docker push <your-username>/my-testprivate-repo>.

How To Push or Pull Images from Docker Hub?

To get started with Docker Hub you should be able to get familiar with the below two commands:

1. Push Command

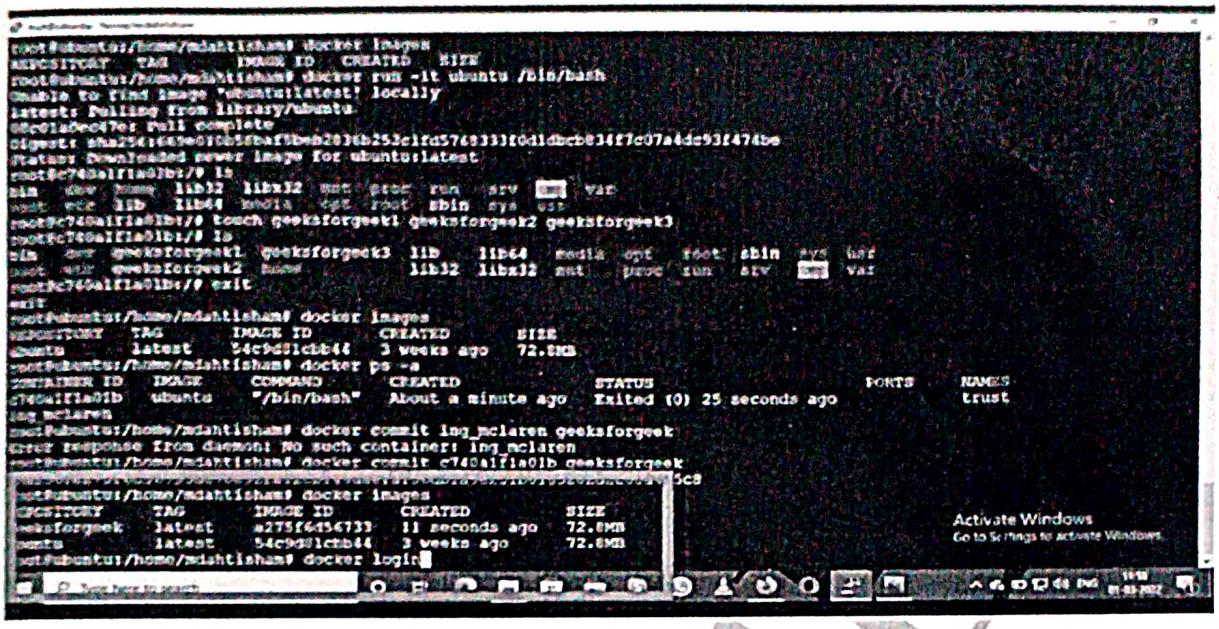
This command as the name suggests itself is used to pushing a docker image onto the docker hub.

Implementation

Follow this example to get an idea of the push command:

- Open Docker in your system.
- Locate the Images that you want to push using the below command:

docker images



The above command will list all the images on your system.

Step 1: Go to the browser and search *hub.docker.com.*

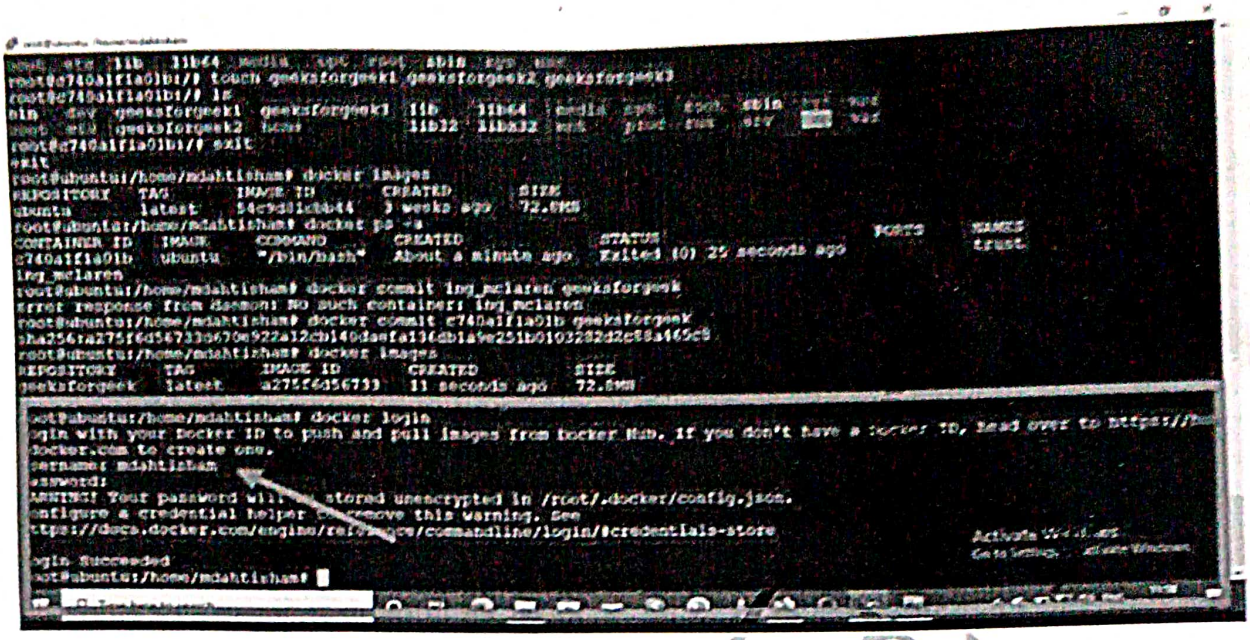
Step 2: Sign up on the docker hub if you do not have a docker hub account, after login on to docker hub.

Step 3: Back to the docker terminal and execute the below command:

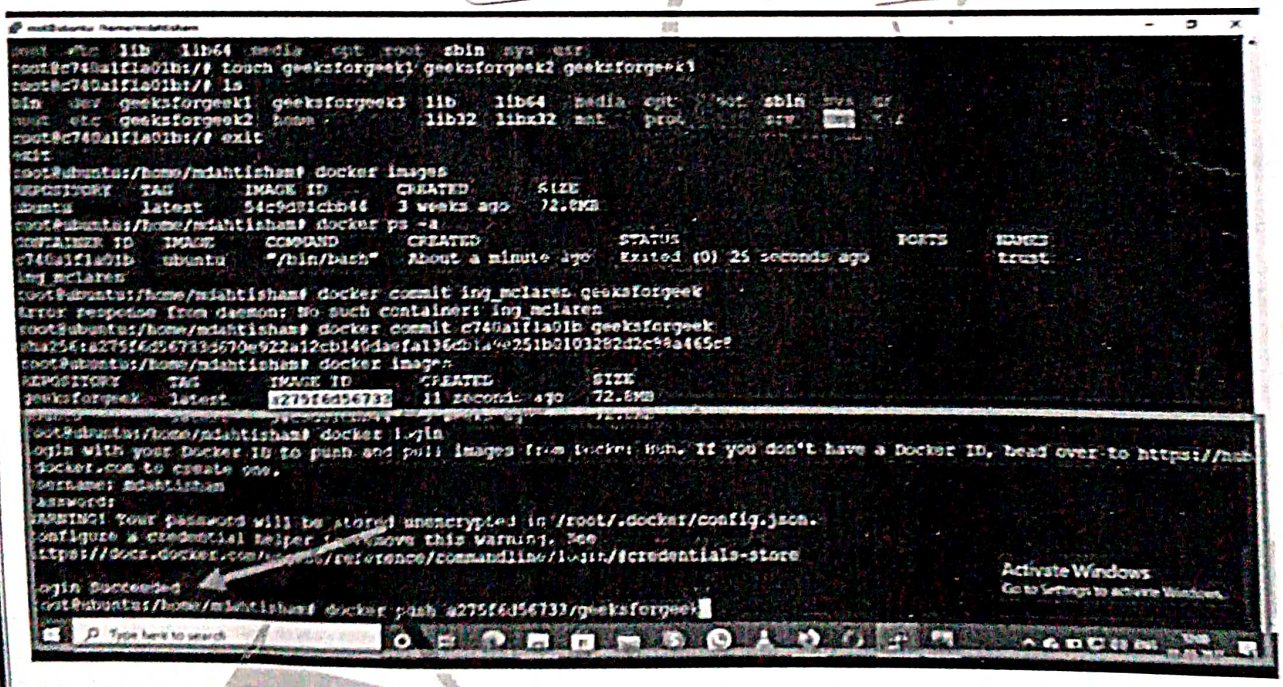
docker login

Step 4: Then give your credential and type in your docker hub username or password.

- username
- password



Step 5: After that hit the Enter key you will see login success on your screen.



Step 7: Then type the tag images name, docker hub username, and give the name it appears on the docker hub using the below command:

docker tag geeksforgeek mdahtisham/geeksimage

geeksforgeek - Image name

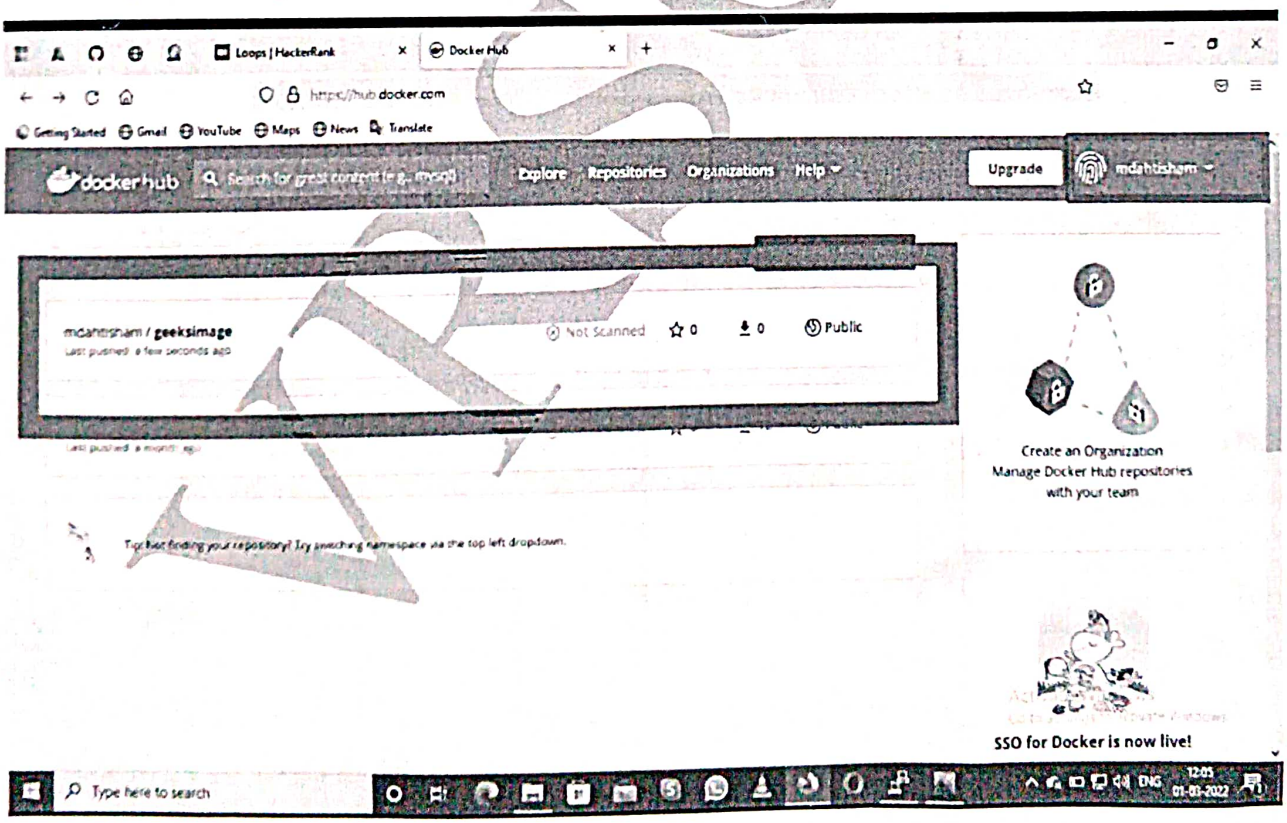
mdahtisham - Docker hub username

geeksimage - With this name Image will appear on the docker hub

Step 8: Now push your image using the below command:

docker push mdahtisham/geeksimage

Note: Below you can see the Docker Image successfully pushed on the docker hub: mdahtisham/geeksimage



2. Pull Command

The pull command is used to get an image from the Docker Hub.

Implementation:

Follow the example to get an overview of the pull command in Docker:

Step 1: Now you can search the image using the below command in docker as follows:

docker search *imagename*

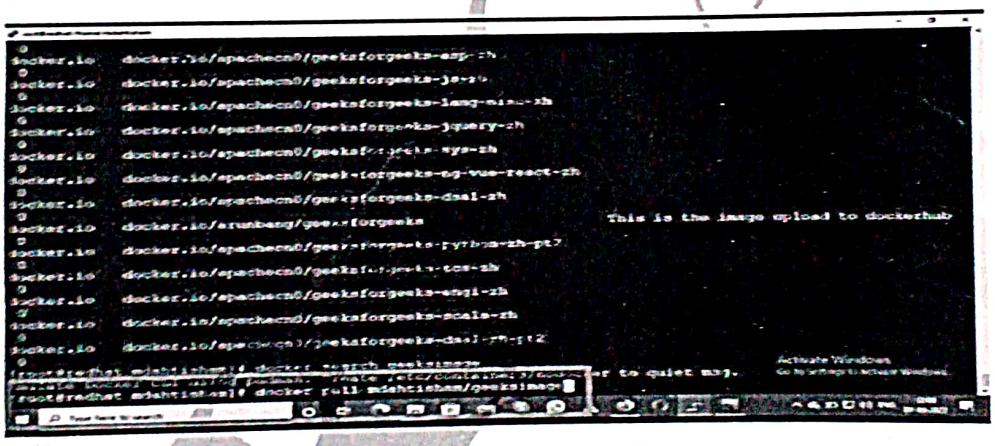
One can see all images on your screen if available images with this name.One can also pull the images if one knows the exact name

Step 2: Now pull the image see the below command.

docker pull mdahtisham/geeksimage

mdahtisham - Docker Hub username

geeksimage - With this name Image will appear on the docker hub



Step 3: Now check for the pulled image using the below command as follows:

docker images

PART-A

1. **Define Virtualization.**

Virtualization is a process that allows a computer to share its hardware resources with multiple digitally separated environments. Each virtualized environment runs within its allocated resources, such as memory, processing power, and storage.

2. **List the types of Virtualization in cloud.**

Server Virtualization
Network Virtualization
Storage Virtualization
Desktop Virtualization
Application Virtualization

3. **What is meant by Desktop Virtualization?**

The processing of multiple virtual desktops occurs on one or a few physical servers, typically at the centralized data center. The copy of the OS and applications that each end user utilizes will typically be cached in memory as one image on the physical server.

The Desktop virtualization provides a virtual desktop environment where client can access the system resources remotely through the network.

4. **List out the advantages of Desktop virtualization.**

- It provides easier management of devices and operating systems due to centralized management.
- It reduces capital expenditure and maintenance cost of hardware due to consolidation of multiple operating systems into a single physical server,
- Upgradation of operating system is easier
- It can facilitate Work from Home feature for IT Employees due to the desktop operating system delivery over the internet.

5. **What is meant by Network Virtualization?**

The Network virtualization is the ability to create virtual networks that are decoupled from the underlying network hardware. This ensures the network can better integrate with and support increasingly virtual environments. It has capability to combine multiple physical networks into one virtual, or it can divide one physical network into separate, independent virtual networks.

6. **List out the advantages of network virtualization.**

The benefits of network virtualization are

- It consolidates the physical hardware of a network into a single virtual network that reduce the management overhead of network resources.
- It gives better scalability and flexibility in network operations.
- It provides automated provisioning and management of network resources.

7. **What is meant by Storage Virtualization?**

Storage virtualization is the process of grouping multiple physical storages using software to appear as a single storage device in a virtual form.

It pools the physical storage from different network storage devices and makes it appear to be a single storage unit that is handled from a single

console. Storage virtualization helps to address the storage and data management issues by facilitating easy backup, archiving and recovery tasks in less time.

8. **List out the advantages of storage virtualization.**

The benefits provided by storage virtualization are

- Automated management of storage mediums with estimated of down time.
- Enhanced storage management in heterogeneous IT environment.
- Better storage availability and optimum storage utilization.
- It gives scalability and redundancy in storage.
- It provides consummate features like disaster recovery, high availability, consistency, replication & re-duplication of data.

9. **What is meant by Server Virtualization?**

A Server virtualization is the process of dividing a physical server into multiple unique and isolated virtual servers by means of software. It partitions a single physical server into the multiple virtual servers; each virtual server can run its own operating system and applications independently.

10. **List out the advantages of server virtualization.**

The benefits of server virtualization are

- It gives quick deployment and provisioning of virtual operating system.
- It has reduced the capital expenditure due to consolidation of multiple servers into a single physical server which eliminate the cost of multiple physical hardware.
- It provides ease in development & testing.
- It makes optimum resource utilization of physical server.

11. **What is meant by Application Virtualization**

Application virtualization is a technology that encapsulates an application from the underlying operating system on which it is executed. It enables access to an application without needing to install it on the local or target device. From the user's perspective, the application works and interacts like it's native on the device.

It allows to use any cloud client which supports BYOD like Thin client, Thick client, Mobile client, PDA and so on.

12. **List out the advantages of Application virtualization.**

- It allows for cross-platform operations like running Windows applications on Linux or android and vice versa.
- It allows to run applications that have legacy issues like supported on older Operating systems.
- It avoids conflict between the other virtualized applications
- It allows a user to run more than one instance of an application at same time
- It reduces system integration and administration costs by maintaining a common software baseline across multiple diverse computers in an organization.

13. **List out the Live VM Migration Steps.**

Live migration of a VM consists of the following six steps:

Steps 0 and 1: Start migration.

Steps 2: Transfer memory.

Step 3: Suspend the VM and copy the last portion of the data.

Steps 4 and 5: Commit and activate the new host.

14. **What is File Migration?**

- To support VM migration, a system must provide each VM with a consistent, location-independent view of the file system that is available on all hosts.
- A simple way to achieve this is to provide each VM with its own virtual disk, which the file system is mapped to, and transport the contents of this virtual disk along with the other states of the VM.

15. **What is Memory Migration?**

- Memory Migration is one of the most important aspects of VM migration. Moving the memory instance of a VM from one physical host to another can be approached in any number of ways.
- Memory migration can be in a range of hundreds of megabytes to a few gigabytes in a typical system today, and it needs to be done in an efficient manner.

16. **What is Network Migration ?**

A migrating VM should maintain all open network connections without relying on forwarding mechanisms on the original host or on support from ~~mobility or~~ redirection mechanisms.

- To enable remote systems to locate and communicate with a VM, each VM must be assigned a virtual IP address known to other entities.

17. **What is Dynamic Deployment of Virtual Clusters?**

The Cellular Disco at Stanford is a virtual cluster built in a shared-memory multiprocessor system. The INRIA virtual cluster was built to test parallel algorithm performance. The COD (Cluster-on-Demand) project is a virtual cluster management system for dynamic allocation of servers from a computing pool to multiple virtual clusters.

18. **What is Docker?**

Docker is an open-source centralized platform designed to create, deploy, and run applications. Docker uses container on the host's operating system to run applications. It allows applications to use the same Linux kernel as a system on the host computer, rather than creating a whole virtual operating system. Containers ensure that our application works in any environment like development, test, or production.

Docker includes components such as Docker client, Docker server, Docker machine, Docker hub, Docker composes, etc.

19. **List out the Advantages of Docker.**

There are the following advantages of Docker –

It runs the container in seconds instead of minutes.

- It uses less memory.
- It provides lightweight virtualization.
- It does not a require full operating system to run applications.

20. **List out the Disadvantages of Docker.**

There are the following disadvantages of Docker -

- It increases complexity due to an additional layer.
- In Docker, it is difficult to manage large amount of containers.
- Some features such as container self -registration, containers self-inspects, copying files form host to the container, and more are missing in the Docker.

21. **Define Docker Engine.**

It is a client server application that contains the following major components.

- A server which is a type of long-running program called a daemon process.
- The REST API is used to specify interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface client.

22. **What are Docker components?**

Let's look at the core components that compose Docker:

- The Docker client and server
- Docker Images
- Registries
- Docker Containers

23. **Define Docker images.**

Images are the building blocks of the Docker world. You launch your containers from images. Images are the "build" part of Docker's life cycle. They are a layered format, using Union file systems, that are built step-by-step using a series of instructions. For example:

- Add a file.
- Run a command.
- Open a port.

24. **Define Containers.**

Docker helps you build and deploy containers inside of which you can package your applications and services. As we've just learnt, containers are launched from images and can contain one or more running processes. You can think about images as the building or packing aspect of Docker and the containers as the running or execution aspect of Docker.

A Docker container is:

- An image format.
- A set of standard operations.
- An execution environment.

25. **What is a Docker image?**

A Docker image is made up of filesystems layered over each other. At the base is a boot filesystem, bootfs, which resembles the typical Linux/Unix boot filesystem. A Docker user will probably never interact with the boot filesystem. Indeed, when a container has booted, it is moved into memory,

and the boot filesystem is unmounted to free up the RAM used by the initrd disk image.

26. **Difference between Docker Vs VM (Virtual Machine).**

Virtual Machines	Docker Containers
Need more resources	Less resources are used
Process isolation is done at the hardware level	Process Isolation is done at Operating System-level
Separate Operating System for each VM	Operating System resources can be shared within Docker
VMs can be customized	Custom container setup is easy
Takes time to create a Virtual Machine	The creation of docker is very quick
Booting takes minutes	Booting is done within seconds.

27. **What is a Docker container?**

This is a very important question so just make sure you don't deviate from the topic and I will advise you to follow the below mentioned format:

- Docker containers include the application and all of its dependencies, but share the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers are not tied to any specific infrastructure: they run on any computer, on any infrastructure, and in any cloud.
- Now explain how to create a Docker container, Docker containers can be created by either creating a Docker image and then running it or you can use Docker images that are present on the Dockerhub.
- Docker containers are basically runtime instances of Docker images.

28. **What is a Docker hub?**

Docker hub is a cloud-based registry service that allows you to link to code repositories, build your images and test them, store manually pushed images, and link to the Docker cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution and change management, user and team collaboration, and workflow automation throughout the development pipeline.

29. **What command can you run to export a docker image as an archive?**

This can be done using the docker save command and the syntax is: docker save -o <exported_name>.tar <container-name>

30. **What command can be run to import a pre-exported Docker image into another Docker host?**

This can be done using the docker load command and the syntax is docker load -i <export_image_name>.tar

31. **Can a paused container be removed from Docker?**

No, it is not possible! A container MUST be in the stopped state before we can remove it.