1. **docker --version**
   * Displays the current version of Docker installed on your machine.
2. **docker ps**
   * Lists all running Docker containers.
3. **docker ps -a**
   * Lists all containers, including the stopped ones.
4. **docker run <image>**
   * Runs a container from the specified image.
5. **docker run -it <image>**
   * Runs a container interactively with a terminal session.
6. **docker run -d <image>**
   * Runs a container in detached mode (in the background).
7. **docker exec -it <container\_id> bash**
   * Opens an interactive bash shell inside a running container.
8. **docker stop <container\_id>**
   * Stops a running container.
9. **docker start <container\_id>**
   * Starts a stopped container.
10. **docker restart <container\_id>**
    * Restarts a running or stopped container.
11. **docker kill <container\_id>**
    * Immediately stops a container by sending a SIGKILL signal.
12. **docker rm <container\_id>**
    * Removes a container from the system (requires the container to be stopped).
13. **docker rmi <image\_id>**
    * Removes an image from the system.
14. **docker images**
    * Lists all Docker images available locally.
15. **docker build -t <image\_name>:<tag> <path>**
    * Builds an image from a Dockerfile at the specified path.
16. **docker pull <image>**
    * Downloads a Docker image from a registry (e.g., Docker Hub).
17. **docker push <image>**
    * Uploads a Docker image to a registry.
18. **docker logs <container\_id>**
    * Displays the logs of a running or stopped container.
19. **docker inspect <container\_id>**
    * Provides detailed information about a container or image in JSON format.
20. **docker network ls**
    * Lists all Docker networks.
21. **docker network create <network\_name>**
    * Creates a new Docker network.
22. **docker network inspect <network\_name>**
    * Displays detailed information about a specific Docker network.
23. **docker volume ls**
    * Lists all Docker volumes.
24. **docker volume create <volume\_name>**
    * Creates a new Docker volume.
25. **docker volume inspect <volume\_name>**
    * Displays detailed information about a specific Docker volume.
26. **docker-compose up**
    * Starts and runs containers defined in a docker-compose.yml file.
27. **docker-compose up -d**
    * Starts and runs containers in detached mode, defined in a docker-compose.yml file.
28. **docker-compose down**
    * Stops and removes containers, networks, and volumes created by docker-compose up.
29. **docker-compose logs**
    * Shows logs for services defined in the docker-compose.yml file.
30. **docker-compose exec <service\_name> bash**
    * Opens a shell in a running service container.
31. **docker stats**
    * Displays real-time statistics for all running containers.
32. **docker system prune**
    * Removes unused Docker data (e.g., stopped containers, unused images, and networks).
33. **docker system df**
    * Displays disk space usage for Docker images, containers, and volumes.
34. **docker login <registry>**
    * Logs into a Docker registry (e.g., Docker Hub).
35. **docker logout <registry>**
    * Logs out of a Docker registry.
36. **docker update <container\_id>**
    * Updates the configuration of a running container (e.g., resource limits).
37. **docker exec -it <container\_id> sh**
    * Opens a shell inside a running container if bash is unavailable.
38. **docker pull <image>:<tag>**
    * Pulls a specific version (tag) of an image from a Docker registry.
39. **docker tag <image\_id> <new\_image\_name>:<new\_tag>**
    * Tags an image with a new name or version.
40. **docker volume rm <volume\_name>**
    * Removes a Docker volume.
41. **docker run --rm <image>**
    * Runs a container and removes it automatically when it stops.
42. **docker build --no-cache -t <image\_name> .**
    * Builds a Docker image without using the cache from previous builds.
43. **docker exec -u <user\_id>:<group\_id> <container\_id> <command>**
    * Executes a command as a specific user inside a container.
44. **docker cp <container\_id>:<path\_in\_container> <host\_path>**
    * Copies files or directories from a container to the host machine.
45. **docker cp <host\_path> <container\_id>:<path\_in\_container>**
    * Copies files or directories from the host machine to a container.
46. **docker commit <container\_id> <new\_image\_name>**
    * Creates a new image from the current state of a container.
47. **docker push <registry>/<image\_name>:<tag>**
    * Pushes a Docker image to a remote registry (e.g., Docker Hub).
48. **docker history <image\_name>**
    * Displays the history of an image, including all layers.
49. **docker attach <container\_id>**
    * Attaches to a running container's main process and provides interactive access.
50. **docker exec <container\_id> <command>**
    * Executes a command in a running container.

Here are additional Docker commands with explanations:

1. **docker-compose build**
   * Builds images for services defined in a docker-compose.yml file.
2. **docker-compose stop**
   * Stops services defined in a docker-compose.yml file without removing them.
3. **docker-compose restart**
   * Restarts services defined in a docker-compose.yml file.
4. **docker-compose run <service\_name> <command>**
   * Runs a one-off command in a new container for the specified service.
5. **docker-compose exec <service\_name> <command>**
   * Executes a command inside a running container of the specified service.
6. **docker-compose ps**
   * Lists all the containers related to the services defined in a docker-compose.yml file.
7. **docker-compose config**
   * Validates and displays the configuration of a docker-compose.yml file.
8. **docker build -t <image\_name> --build-arg <arg\_name>=<value> .**
   * Builds a Docker image, passing build-time variables to the Dockerfile.
9. **docker network create --driver <network\_driver> <network\_name>**
   * Creates a Docker network with a specified driver (e.g., bridge, overlay).
10. **docker network connect <network\_name> <container\_id>**
    * Connects a running container to a Docker network.
11. **docker network disconnect <network\_name> <container\_id>**
    * Disconnects a running container from a Docker network.
12. **docker run --name <container\_name> <image>**
    * Runs a container with a specific name.
13. **docker run -v <host\_path>:<container\_path>**
    * Mounts a volume or directory from the host into a container.
14. **docker run --memory <memory\_limit> <image>**
    * Limits the memory available to a container.
15. **docker run --cpu-shares <cpu\_shares> <image>**
    * Specifies the CPU shares (relative weight) for the container.
16. **docker run --cpus <cpu\_limit> <image>**
    * Limits the number of CPUs available to a container.
17. **docker run --network <network\_name> <image>**
    * Specifies the network the container should connect to.
18. **docker volume prune**
    * Removes all unused Docker volumes.
19. **docker network prune**
    * Removes all unused Docker networks.
20. **docker system prune --volumes**
    * Removes unused containers, networks, images, and volumes.
21. **docker build -t <image\_name> --no-cache .**
    * Builds a Docker image without using any cached layers.
22. **docker inspect --format '{{.NetworkSettings.IPAddress}}' <container\_id>**
    * Displays the IP address of a container using Go templating.
23. **docker stats <container\_id>**
    * Shows real-time resource usage (CPU, memory, I/O, etc.) for a container.
24. **docker update --memory <memory\_limit> --cpus <cpu\_limit> <container\_id>**
    * Updates the memory and CPU limits of a running container.
25. **docker run --entrypoint <command> <image>**
    * Overrides the default entry point of an image.
26. **docker exec <container\_id> <command>**
    * Executes a command inside a running container.
27. **docker volume create --driver <driver\_name> <volume\_name>**
    * Creates a volume with a specific driver.
28. **docker-compose scale <service\_name>=<number\_of\_instances>**
    * Scales the number of container instances for a specified service in docker-compose.yml.
29. **docker-compose logs -f <service\_name>**
    * Follows the logs of a specific service defined in docker-compose.yml.
30. **docker cp <container\_id>:<path\_in\_container> <host\_path>**
    * Copies files or directories from a container to the host machine.
31. **docker cp <host\_path> <container\_id>:<path\_in\_container>**
    * Copies files or directories from the host to a container.
32. **docker exec <container\_id> cat <file\_path>**
    * Runs the cat command to display the contents of a file inside a container.
33. **docker exec <container\_id> tail -f <log\_file>**
    * Monitors the log file inside a running container in real-time.
34. **docker save -o <output\_file.tar> <image\_name>**
    * Saves a Docker image to a tarball file.
35. **docker load -i <input\_file.tar>**
    * Loads a Docker image from a tarball file.
36. **docker info**
    * Displays detailed information about the Docker installation (e.g., version, number of containers, and images).
37. **docker events**
    * Displays a stream of Docker events in real-time (e.g., container creation, image pull).
38. **docker volume inspect <volume\_name>**
    * Displays detailed information about a Docker volume.
39. **docker network inspect <network\_name>**
    * Displays detailed information about a Docker network.
40. **docker build --file <dockerfile\_path> .**
    * Specifies the Dockerfile path when building an image.
41. **docker run -p <host\_port>:<container\_port> <image>**
    * Maps a container's port to a host's port.
42. **docker run --link <container\_name>:<alias> <image>**
    * Links a container to another container, allowing them to communicate via a hostname alias.
43. **docker run --user <user\_id>:<group\_id> <image>**
    * Runs a container with a specific user and group ID.
44. **docker run --privileged <image>**
    * Grants a container additional privileges (e.g., access to all devices on the host).
45. **docker run --cap-add=<capability> <image>**
    * Adds Linux capabilities to a container (e.g., SYS\_ADMIN, NET\_ADMIN).
46. **docker run --cap-drop=<capability> <image>**
    * Removes Linux capabilities from a container.
47. **docker build -t <image\_name>:<tag> --target <target\_stage> .**
    * Builds a specific stage from a multi-stage Dockerfile.
48. **docker info --format '{{json .}}'**
    * Displays Docker system information in JSON format.
49. **docker volume prune --force**
    * Forces removal of unused volumes without confirmation.
50. **docker run --init <image>** - Runs a container with an init system to handle zombie processes.