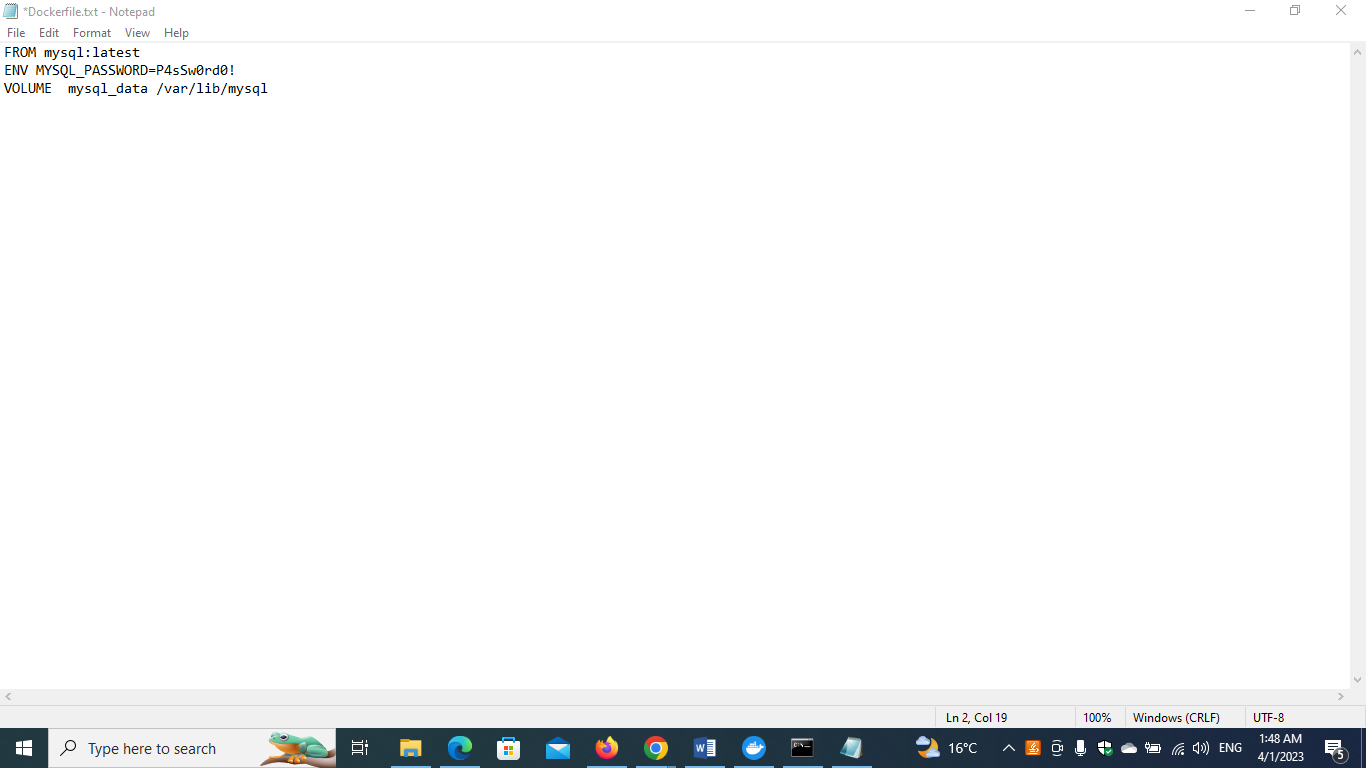
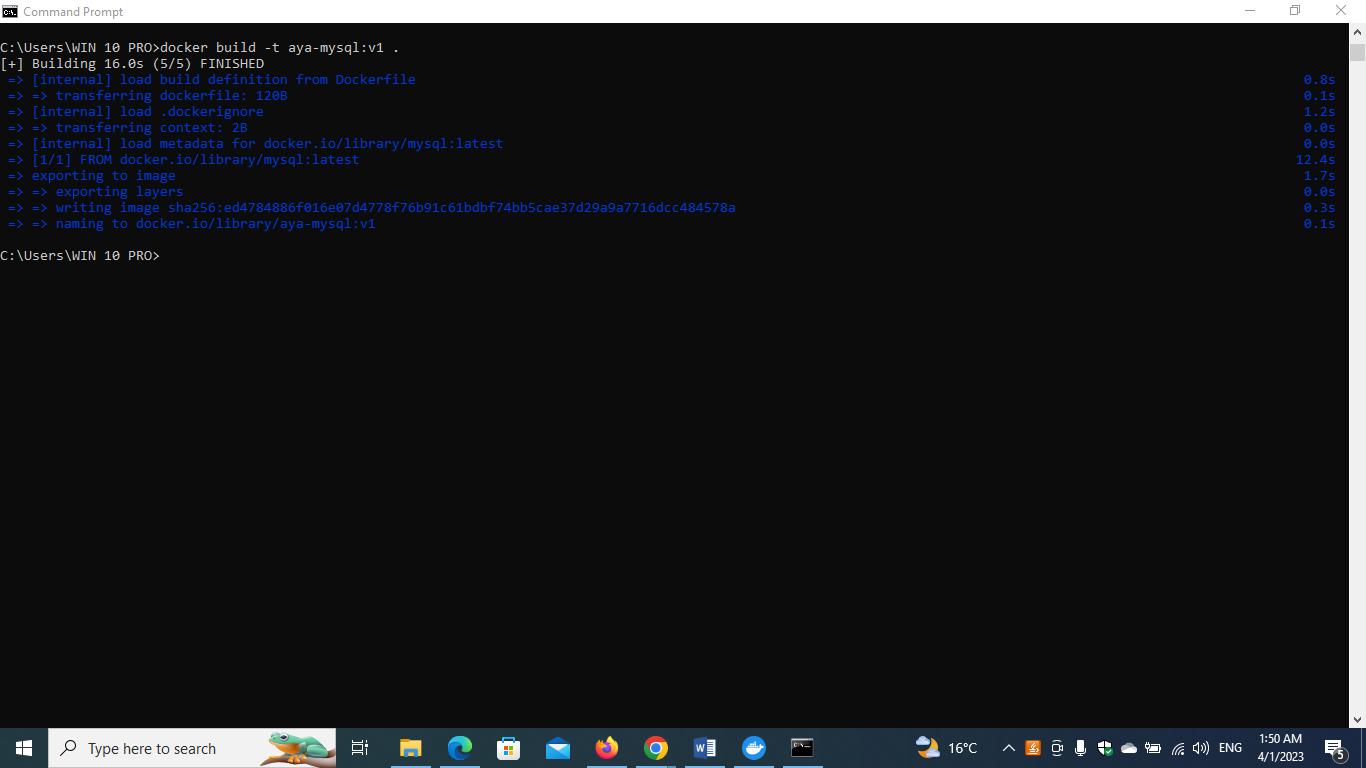
Name: Aya Allah Ali Abbas track: system admin /iti alex

**Lab2**

P2)





**What is the rest of Docker Networks ? “Name and Definition”**

Docker networking is primarily used to establish communication between Docker containers and the outside world via the host machine where the Docker daemon is running.

**Docker supports different types of networks**

**Docker Network Drivers**

**1.The Bridge Driver(known as Docker Default Networking (docker0))**:

bridge network uses a software bridge which allows containers connected to the same bridge network to communicate, while providing isolation from containers which are not connected to that bridge network.

* It is a private default network created on the host
* Containers linked to this network have an internal IP address through which they communicate with each other easily
* The Docker server (daemon) creates a virtual ethernet bridge docker0 that operates automatically, by delivering packets among various network interfaces
* These are widely used when applications are executed in a standalone container

Note :

When you start Docker, a [default bridge network](https://docs.docker.com/network/bridge/#use-the-default-bridge-network) (aka bridge) is created automatically, and newly-started containers connect to it unless otherwise specified. You can also create user-defined custom bridge networks which **are superior to the default**

The downside with the bridge driver is that it’s not recommended for production; the containers communicate via IP address instead of automatic service discovery to resolve an IP address to the container name. Every time you run a container, a different IP address gets assigned to it. It may work well for local development but it’s definitely not a sustainable approach for applications running in production.

### **The Host Driver**

host drivers use the networking provided by the host machine. And it removes network isolation between the container and the host machine where Docker is running

One limitation with the host driver is that it doesn’t work on Docker desktop: you need a Linux host to use it

1. **Overlay**:

Overlay networks connect multiple Docker daemons together and enable swarm services to communicate with each other. It also facilitate communication between a swarm service and a standalone container, or between two standalone containers on different Docker daemons. This strategy removes the need to do OS-level routing between these containers.

* Note: Docker Swarm is a service for containers which facilitates developer teams to build and manage a cluster of swarm nodes within the Docker platform

So we can say that The Overlay driver is for multi-host network communication, as with [Docker Swarm](https://docs.docker.com/engine/swarm/) . It allows containers across the host to communicate with each other without worrying about the setup. So it is safe to say that overlay network as a distributed virtualized network that’s built on top of an existing computer network.

### **Macvlan**

Macvlan networks allow you to assign a MAC address to a container, making it appear as a physical device on your network. The Docker daemon routes traffic to containers by their MAC addresses. Using the macvlan driver is sometimes the best choice when dealing with legacy applications that expect to be directly connected to the physical network, rather than routed through the Docker host’s network stack.

* It simplifies the communication process between containers
* This network assigns a MAC address to the Docker container. With this Mac address, the Docker server (daemon) routes the network traffic to a router
* Note: Docker Daemon is a server which interacts with the operating system and performs all kind of services
* It is suitable when a user wants to directly connect the container to the physical network rather than the Docker host

1. **None:**

 For this container, disable all networking. Usually used in conjunction with a custom network driver. none is not available for swarm services.

* In this network driver, the Docker containers will neither have any access to external networks nor will it be able to communicate with other containers
* This option is used when a user wants to disable the networking access to a container
* In simple terms, None is called a loopback interface, which means it has no external network interfaces

**What is the different between docker images tags ?**

In Docker, images are tagged to differentiate them from one another. A tag is a label that is attached to an image, and it is used to identify the version or variant of an image.

There are several types of tags in Docker, including:

Version tags: These tags are used to identify a specific version of an image. For example, an image might have version tags like "1.0", "2.0", etc.

Variant tags: These tags are used to identify different variants of an image. For example, an image might have variant tags like "alpine", "ubuntu", "centos", etc. Each of these variants has a different base operating system and set of pre-installed packages.

Latest tag: This tag is used to identify the latest version of an image. It is often used as a default tag when no specific version or variant is specified.

Custom tags: These tags are user-defined and can be used to identify images in a way that makes sense for a specific project or workflow.

It's important to note that tags are not immutable, and they can be overwritten or deleted. Therefore, it's crucial to use tags carefully and avoid relying on them for critical tasks or long-term storage.

**References :**

<https://docs.docker.com/network/>