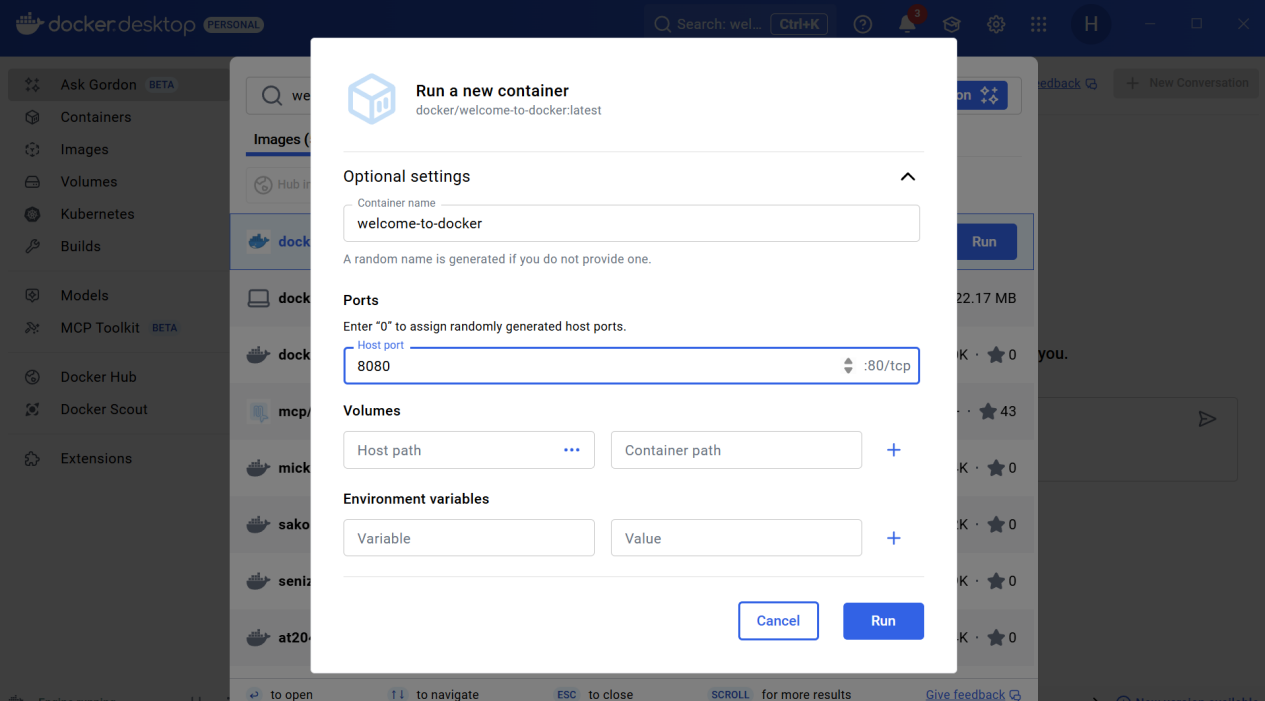
1.A container is an isolated process (plus associated filesystem, network, etc.) that runs on the host but shares the host’s kernel.

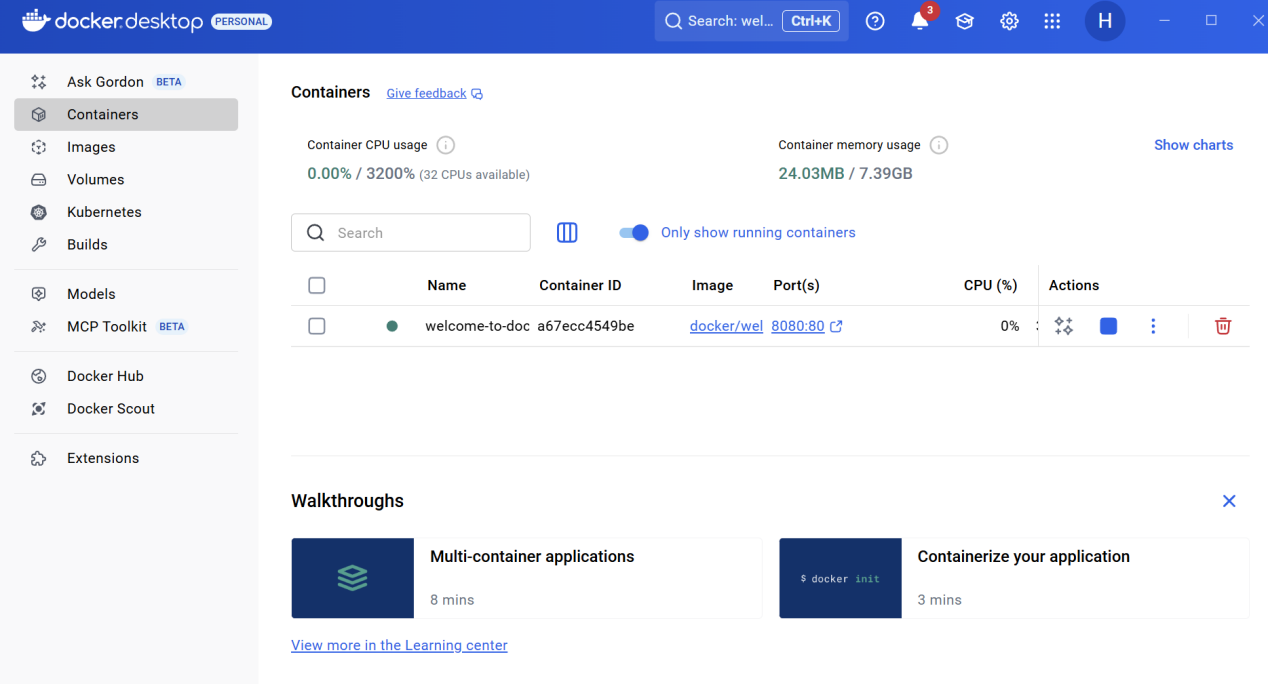
Unlike a VM, a container does not include a full guest OS; containers are lightweight and start fast.

Multiple containers can run on the same host, each with their own isolated view (files, network, processes) but sharing the kernel.

Containers package the code + runtime + libraries + dependencies so behavior is consistent across environments.





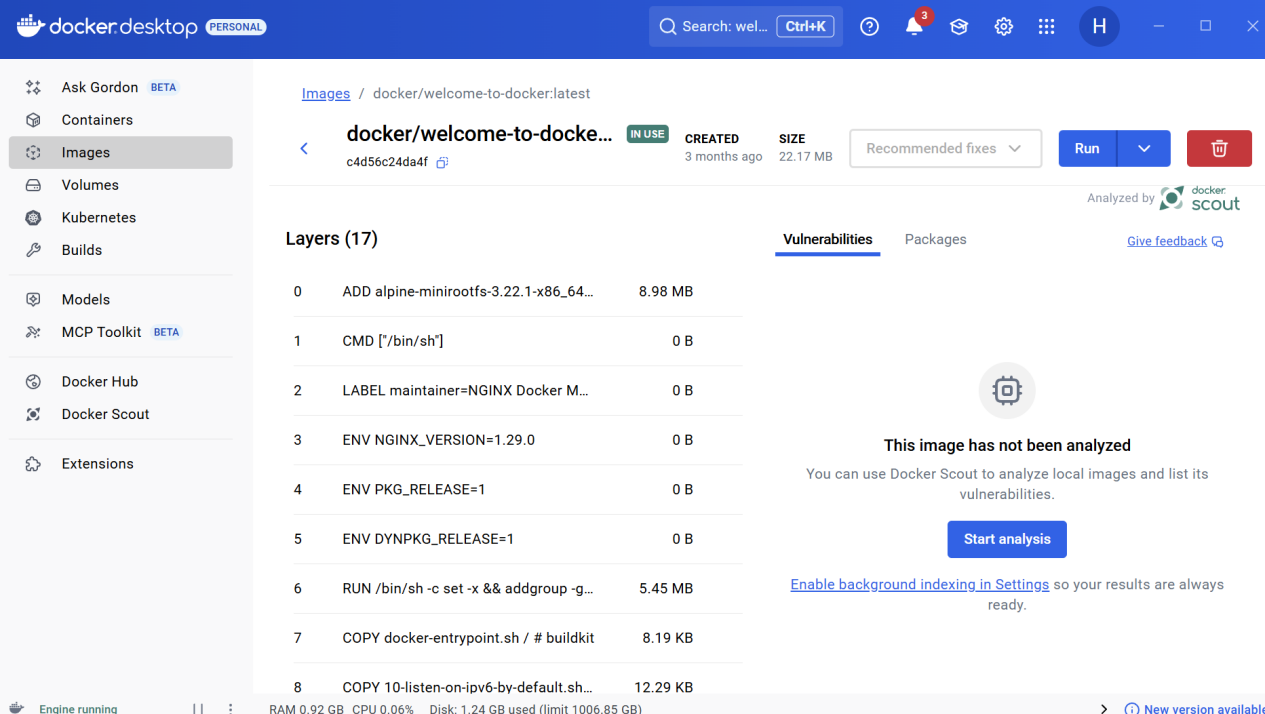


1. An image is a read-only template (blueprint) for containers.

When you run an image, it becomes a container (a live, writable instance).

Images are built in layers—each instruction in a Dockerfile typically adds a layer.

You can pull images from a registry (e.g. Docker Hub) or build them locally.

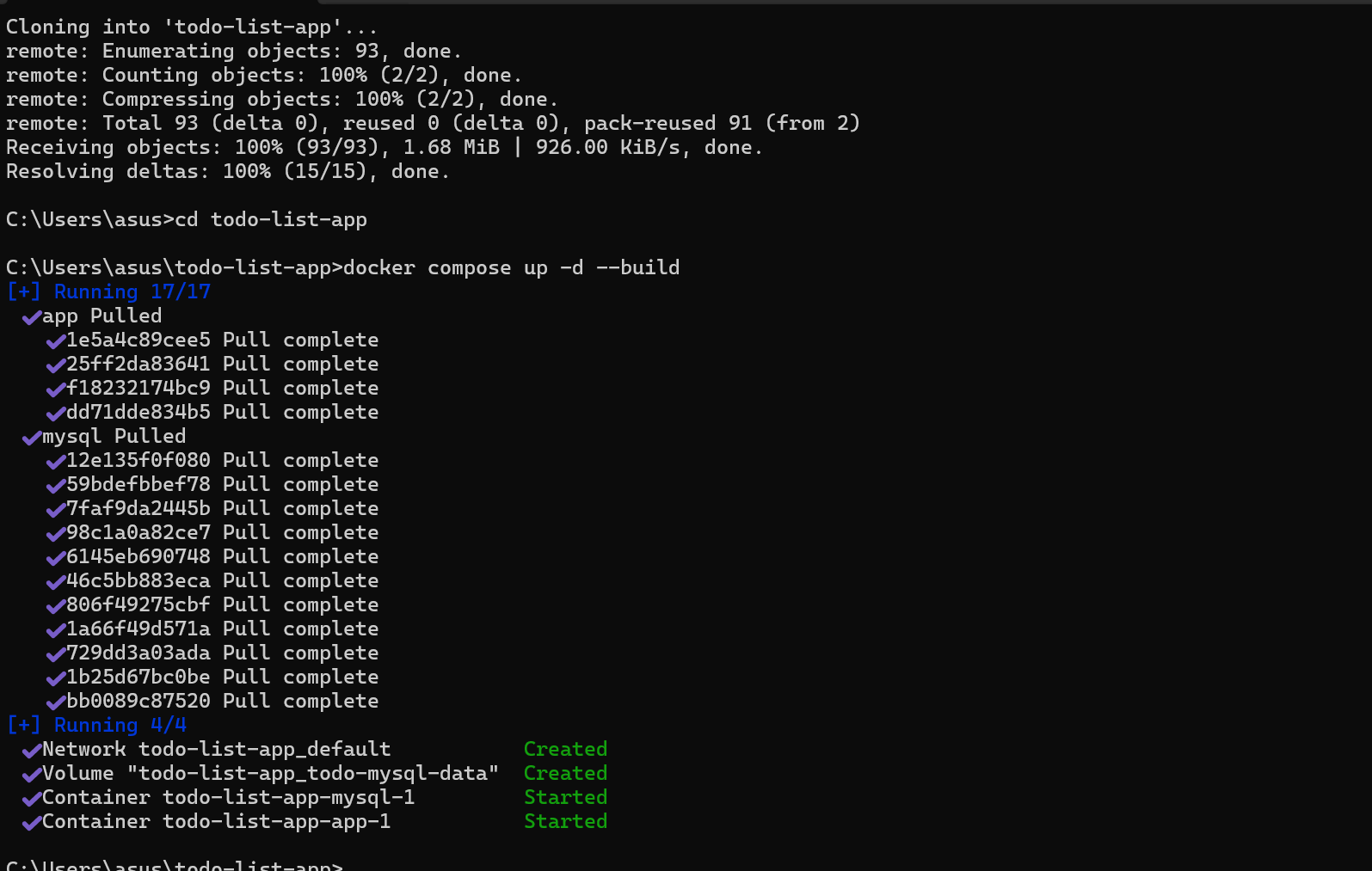


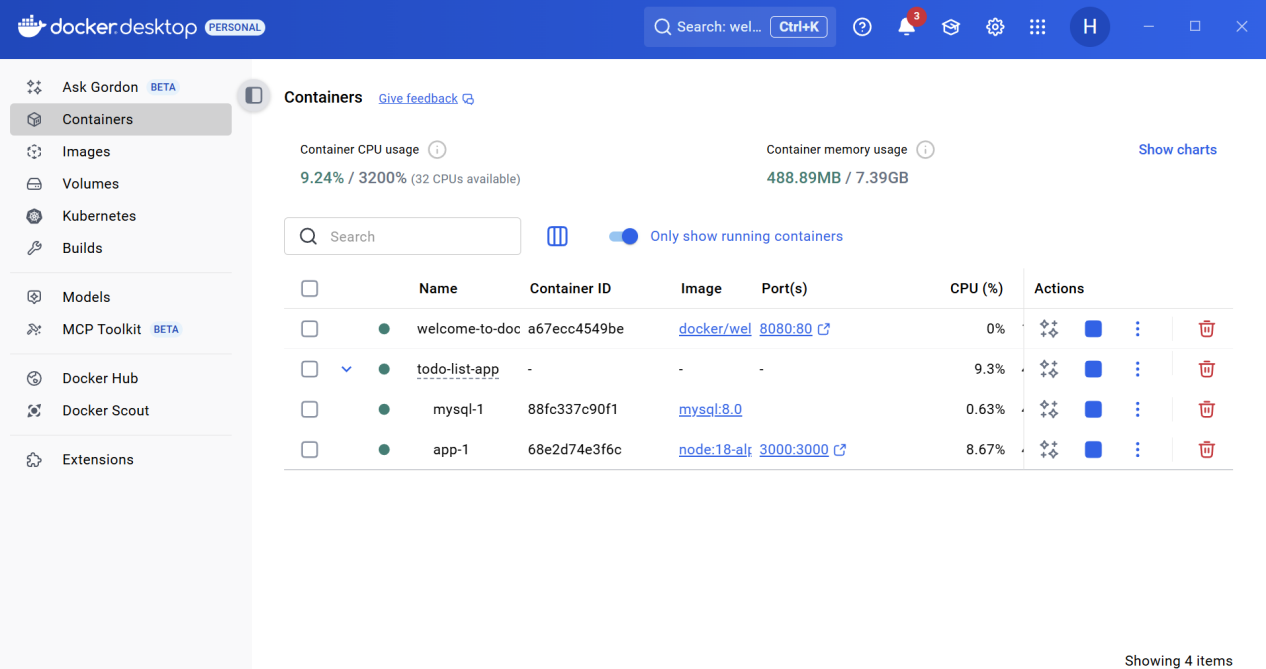
3.Docker Compose is a tool for defining and running multi-container applications via a single YAML file.

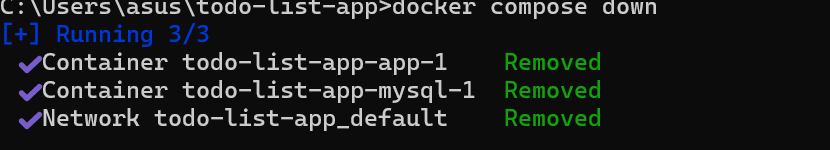
In a docker-compose.yml, you define **services**, **networks**, **volumes**, etc.

Compose lets you start, stop, rebuild all services with one command

It works in development, staging, production, CI workflows.

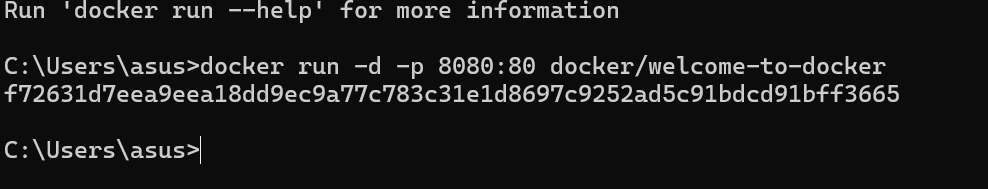


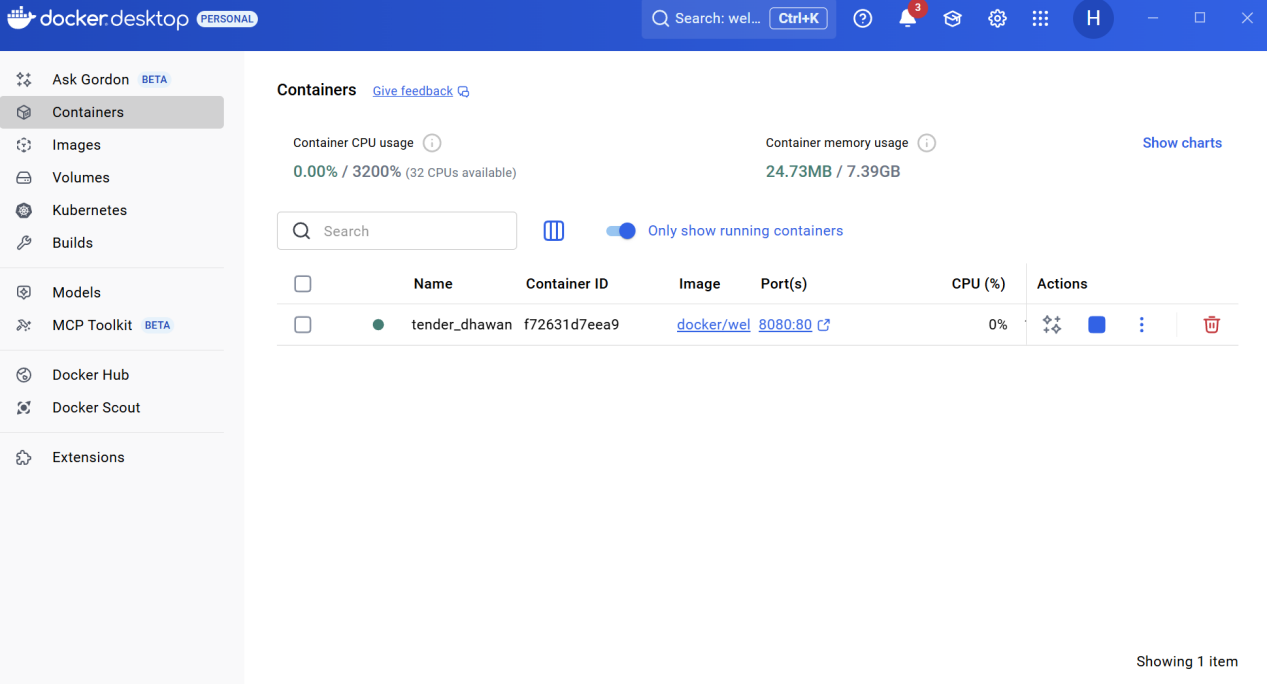




4.Containers are isolated by default.

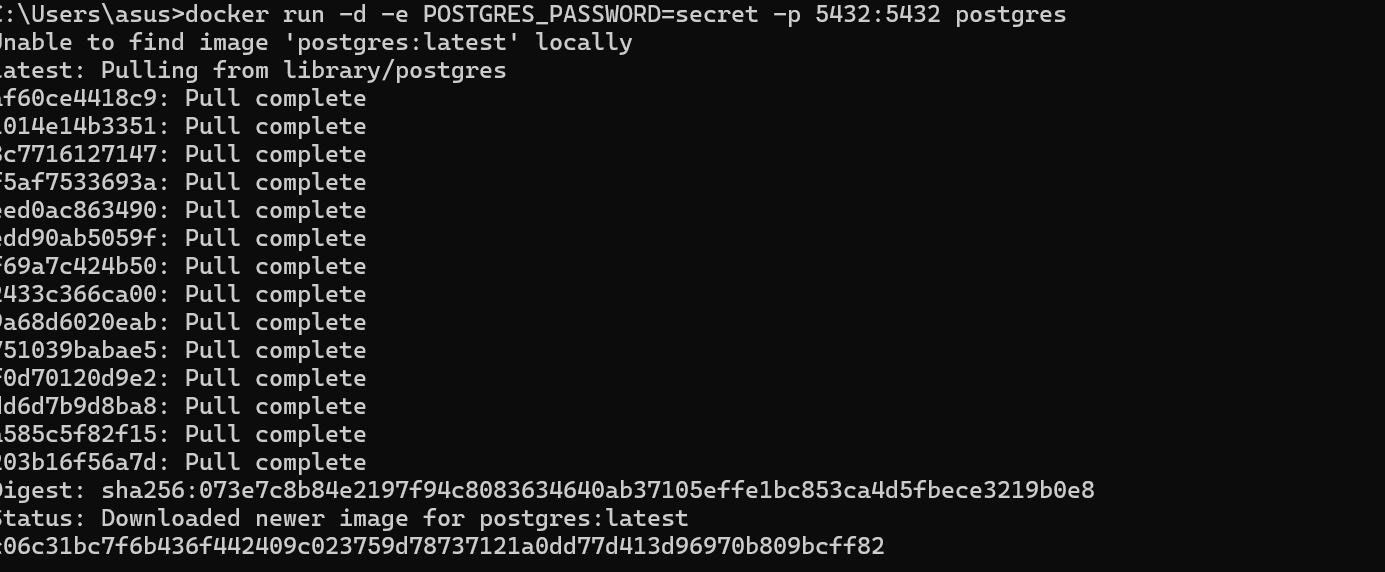
To access them from outside (e.g., browser), you must publish ports.

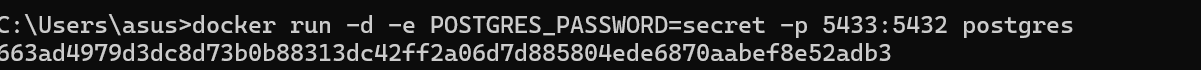


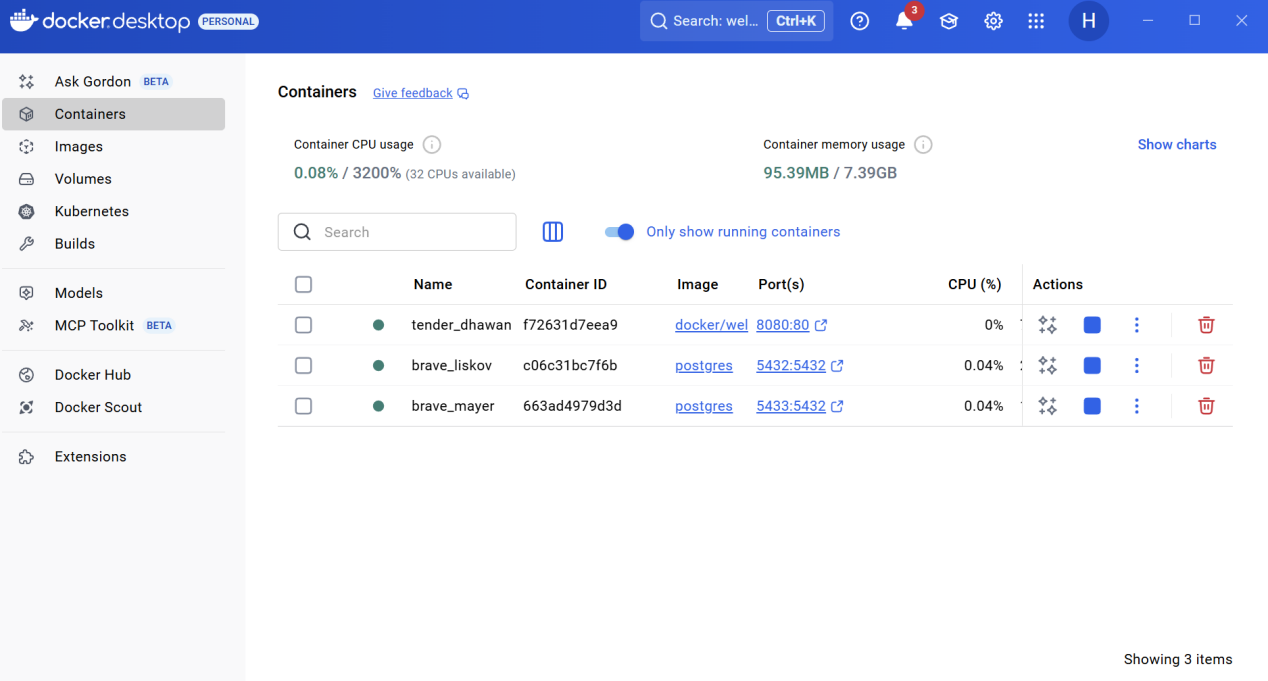


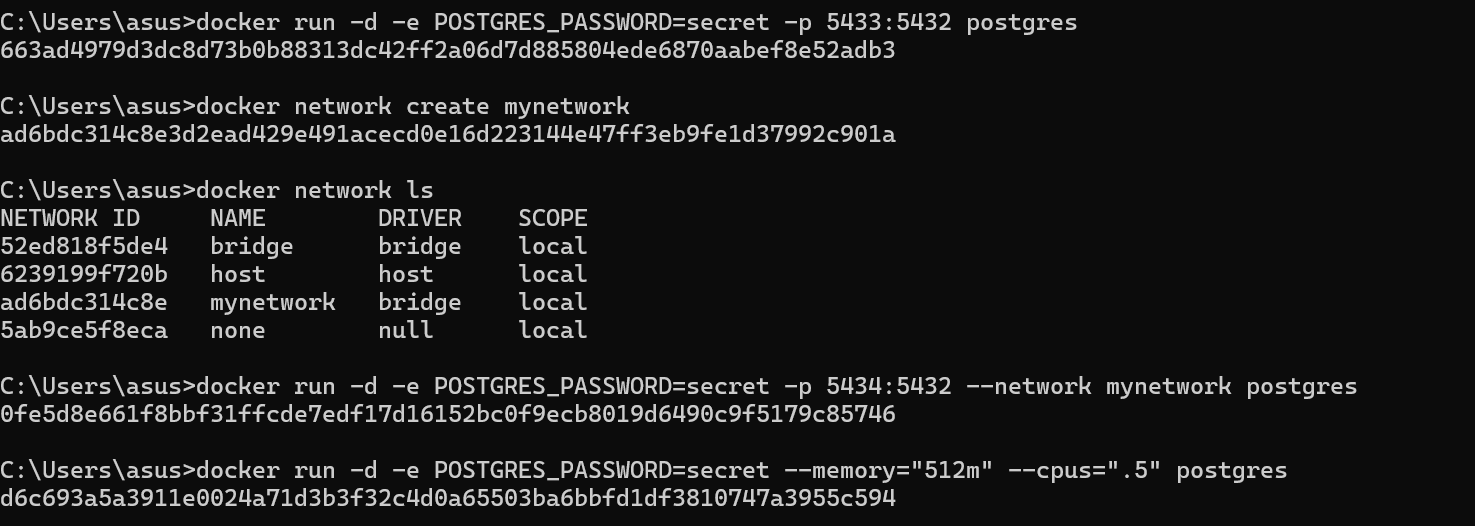


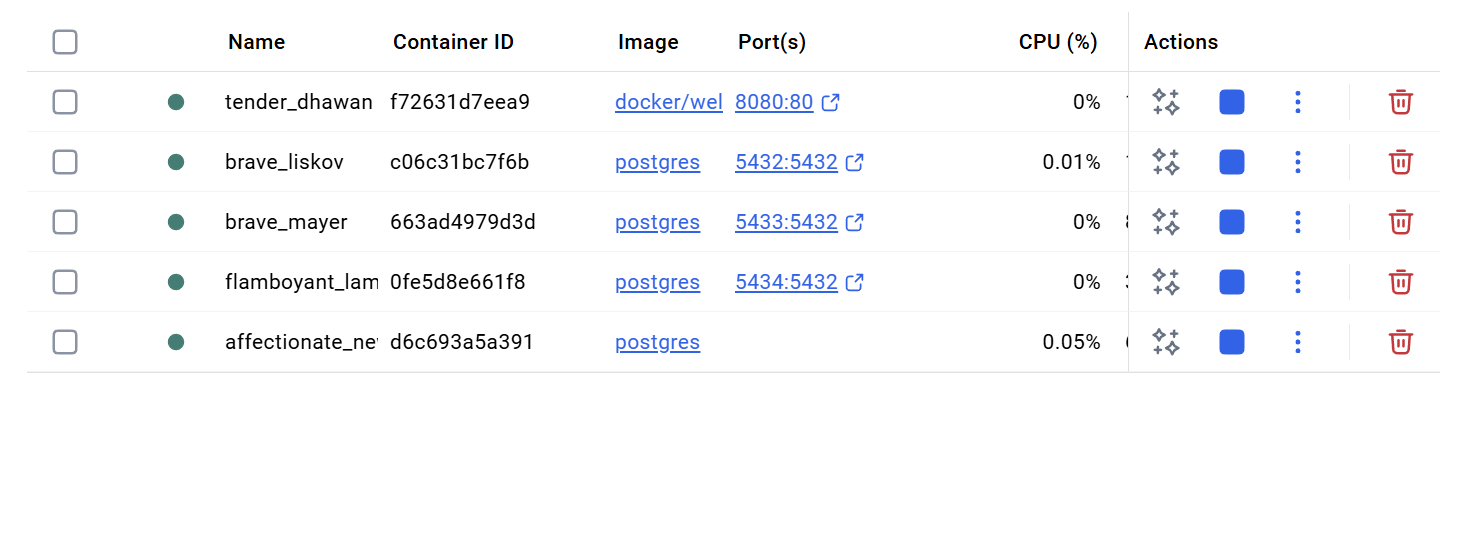
1. You can override the default command





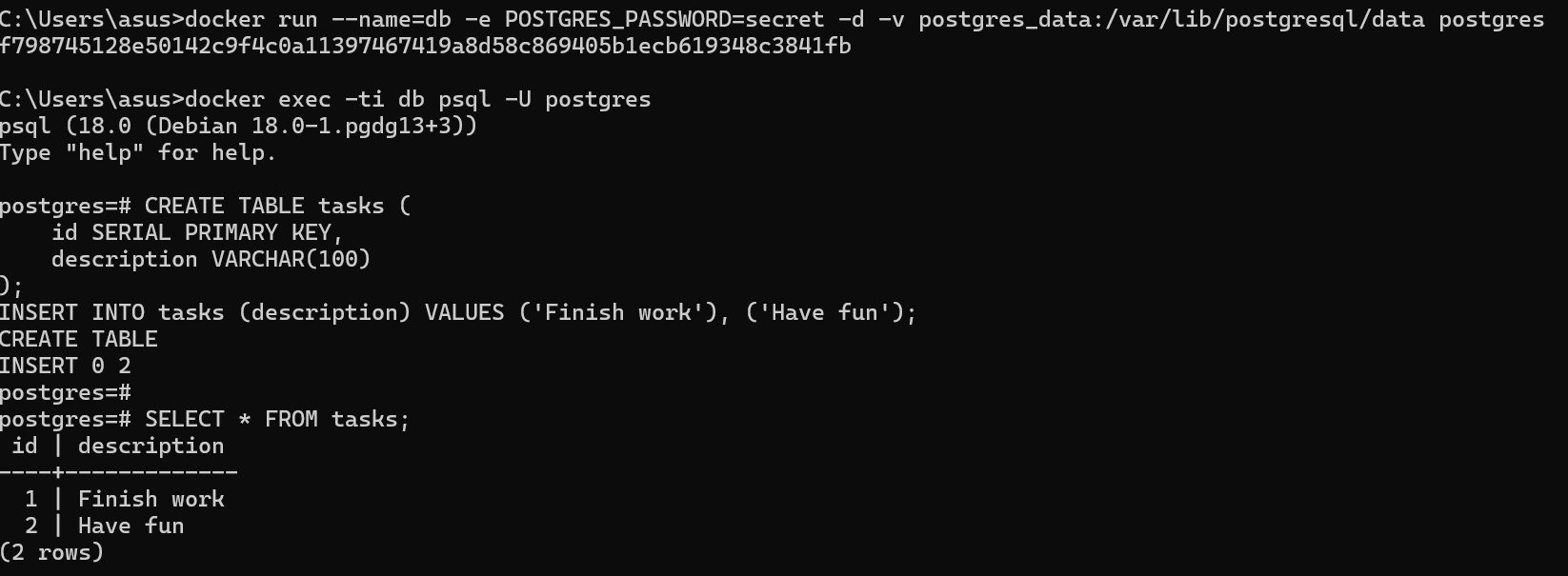




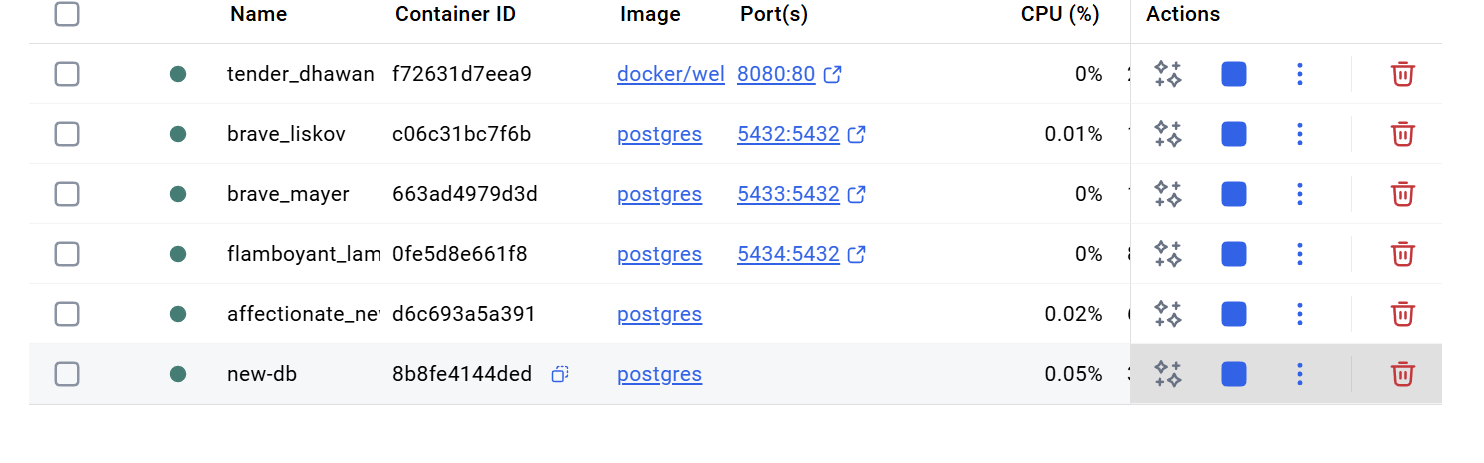
Containers are temporary — data is lost when they’re removed.  
To persist data, use **volumes** or **bind mounts**.

6.Containers are temporary — data is lost when they’re removed.

To persist data, use volumes or bind mounts.





7.You can mount a local folder inside a container

