# CS 470 Module One Assignment Guide

## Introduction

In this lesson, you will gain exposure to Docker. In previous courses, you downloaded and installed MongoDB on your computer and then developed applications to use it. Before your application can run on another computer, you must install the correct version of MongoDB there. Installing software alters your computer’s configuration. These changes may impact other software you have installed. There is also a risk that the other computer or its MongoDB installation is not configured identically to your machine.

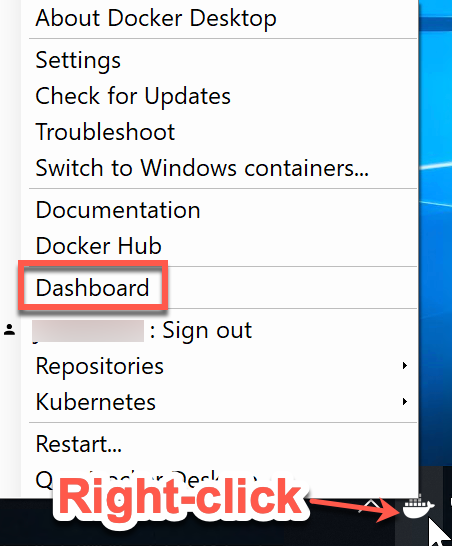
You should run MongoDB inside a container to overcome this common settings inconsistency. Moving the container to a new computer also brings all the configuration and settings along. In Module Two, you will practice using a pre-built Docker container with MongoDB already installed and configured.

## Summary Steps

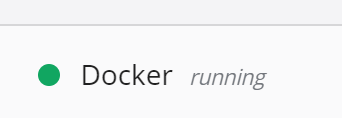
1. Install [Docker Desktop](https://www.docker.com/products/docker-desktop).
2. Open Docker Dashboard.
3. Ensure Docker is running.
4. Open the PowerShell prompt.
5. Create a default MongoDB container with the following commands:
   1. docker pull mongo
   2. docker run -d -p 27017-27019:27017-27019 --name mongodb mongo
   3. docker exec -it mongodb mongosh
      1. use admin
      2. show dbs
      3. **CTRL+D** (exit mongo shell)

## Detailed Steps

Open Docker Dashboard by right-clicking the Windows System Tray icon.



Ensure Docker is running. The icon to the lower left of the dashboard should be green.



Use the Docker pull command below to retrieve a pre-built and configured image from the Docker Registry. To learn more, visit the [Docker Pull](https://docs.docker.com/engine/reference/commandline/pull/) and [Docker Registry](https://hub.docker.com/) webpages.

> docker pull mongo

Use the docker pull command to retrieve a pre-built and configured image from the Docker Registry: 
> docker pull mongo


**Note:** The Docker Registry has official container images from many popular vendors and open-source projects, including operating systems, databases, web servers, applications, and programming environments. Be careful, as you may find yourself up all night checking out the software available at your fingertips!

Use the Docker run command below to start the container. To learn more, visit the [Docker Run](https://docs.docker.com/engine/reference/commandline/run/) webpage.

> docker run -d -p 27017-27019:27017-27019 --name mongodb mongo

Use the docker run command to start the container: 
> docker run -d -p 27017-27019:27017-27019 --name mongodb mongo

* **-d** means “detach” the console from the container so it runs in the background and lets you use additional commands in your window.
* **-p 27017-27019:27017-27019** means to map the range of ports on your computer to the range of ports listed after the colon inside the container.
* **--name mongodb** is the friendly name you want to assign to the container.
* **mongo** is the name of the image you just pulled down from the registry.

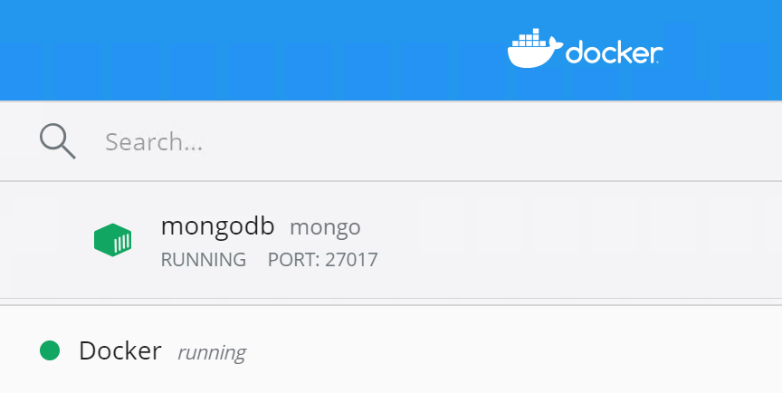
**Note:** The long string of characters displayed after the run command is the unique instance identifier of the container.

Use the Docker images command below to view the installed images on your computer. To learn more, visit the [Docker Images](https://docs.docker.com/engine/reference/commandline/images/) webpage.

> docker images

Use the docker images command to view the installed images on your computer:
> docker images

You will also see the new container in the Docker Dashboard.



Use the Docker exec command to execute a command within the running container. To open a bash shell session inside the container, use the command below. Visit the [Docker Exec](https://docs.docker.com/engine/reference/commandline/exec/) webpage to learn more.

> docker exec -it mongodb mongosh

* **-it** means the command will be run interactively and the input and output will be reflected in your local console terminal (window).
* **mongodb** is the friendly name of the container that you want the command to run in.
* **mongosh** is the name of the command to execute.

**Note:** The **uname -a** command prints information about the machine. As you can see, even though the host computer is running Windows 10 and the blue window is a PowerShell command window, the remote container is running Linux. The **root@e1d5ffe938fb** on the shell prompt indicates the user and hostname of the machine running inside the container. Advanced tip: The hostname is the first several characters of the unique container ID shown earlier.

Until you exit the remote bash shell, anything you type will run inside the container as if your display and keyboard were connected directly to the Linux computer in there. Try the following commands:

> use admin #use the admin database

> show dbs #list the databases

> exit #exit Docker remote bash shell

**Note:** You can press **CTRL+D** to signal “exit” instead of typing out the word. Over time, shortcuts like this will save you time.

Notice the container is running MongoDB version 4.2.6. You can use different versions by specifying the version number on the Docker pull command.

Use the Docker stop command below to stop the container and free up resources like memory and processing on your computer. Visit the [Docket Stop](https://docs.docker.com/engine/reference/commandline/stop/) webpage to learn more.

> docker stop mongodb

## Optional

You can visit the [Docker Hub](https://hub.docker.com/search?type=image&image_filter=store%2Cofficial) and take other images for a spin. Note that containers are meant to be smaller and lighter than full virtual machines (VMs), so graphical interfaces are not possible. For example, if you want to try out Ubuntu and get the full experience, you should instead use a VM because VMs support the full set of hardware on your machine, including mice, USB memory sticks, printers, scanners, and so on.