**Reference links**

[**https://docs.docker.com/desktop/install/windows-install/**](https://docs.docker.com/desktop/install/windows-install/)

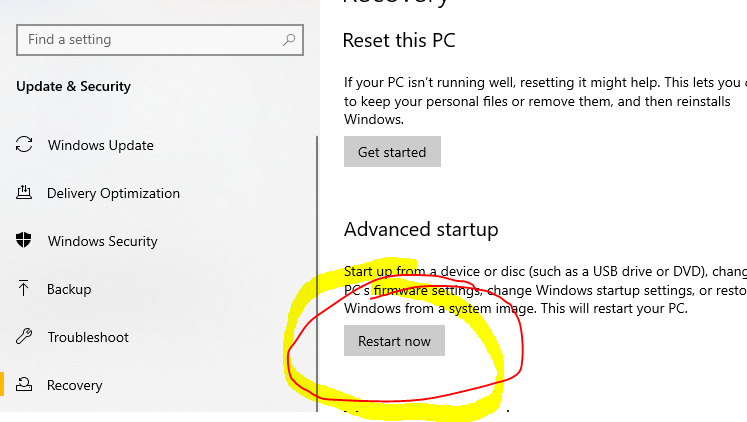
[**https://learn.microsoft.com/en-us/windows/wsl/install**](https://learn.microsoft.com/en-us/windows/wsl/install)

**Installing Docker with WSL on Windows 10/11**

This note will provide detailed steps and instructions to install Docker and signup for a DockerHub account on **Windows**with**WSL**. We will need a DockerHub account so that we can pull images and push the images we will build.

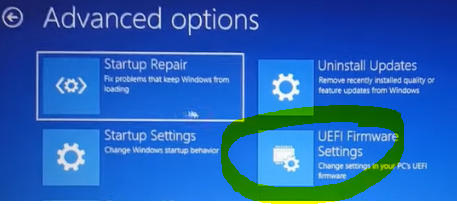
Windows 10 & 11 users will be able to install Docker Desktop if their computer supports the Windows Subsystem for Linux (WSL).

1. First you have to enable virtualisation in BIOS, while starting computer press F2 and you will come to know the option or follow below to goto that BIOS screen



Windows🡪 Settings🡪Recovery🡪Advance startup—Restart now

Now u will get one diff option



1. **Register for a DockerHub account**

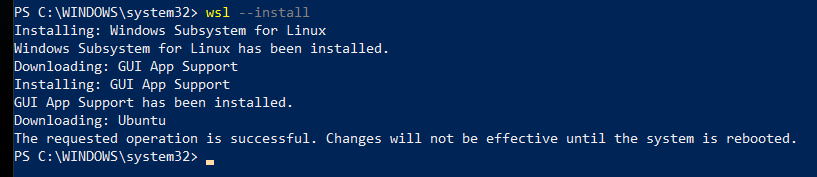
Visit the link below to register for a DockerHub account (this is free)

<https://hub.docker.com/signup>

1. **Download and install all pending Windows OS updates**
2. **Run the WSL install script**

*Note - If you have previously enabled WSL and installed a distribution you may skip to****step #7***

Open PowerShell as Administrator and run: wsl --install  
This will enable and install all required features as well as install Ubuntu.

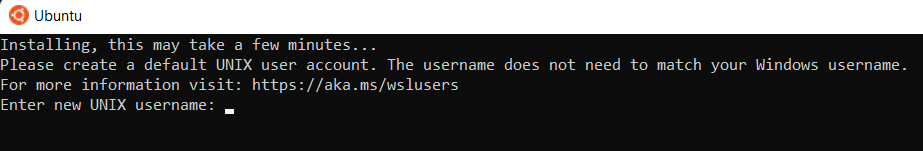


Official documentation:

<https://docs.microsoft.com/en-us/windows/wsl/install#install-wsl-command>

1. **Reboot your computer**
2. **Set a Username and Password in Ubuntu**

After the reboot, Windows will auto-launch your new Ubuntu OS and prompt you to set a username and password.



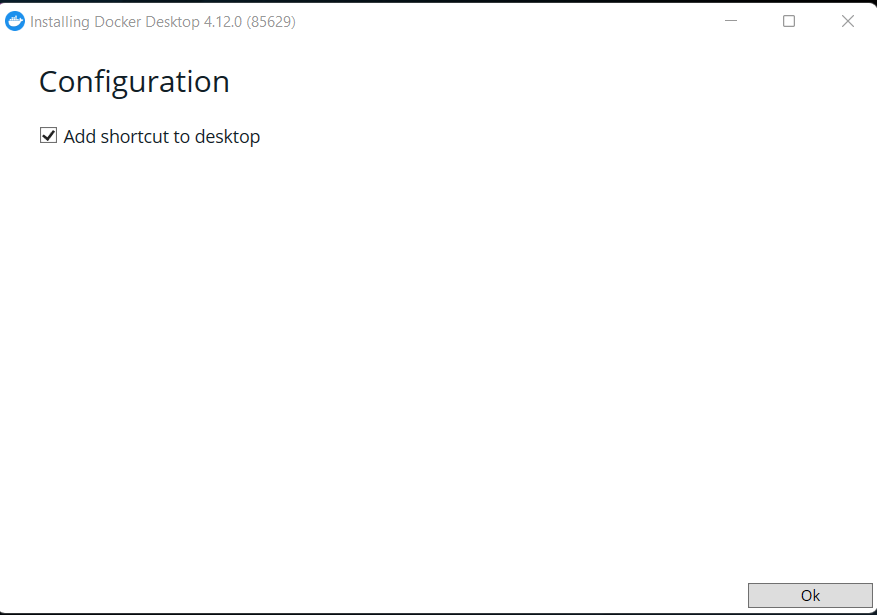
1. **Manually Installing a Distribution**If for some reason Windows did not prompt you to create a distribution or you simply would like to create a new one, you can do so by running the following command:  
   wsl --install -d Ubuntu
2. **Install Docker Desktop**

Navigate to the Docker Desktop installation page and click the Docker Desktop for Windows button:

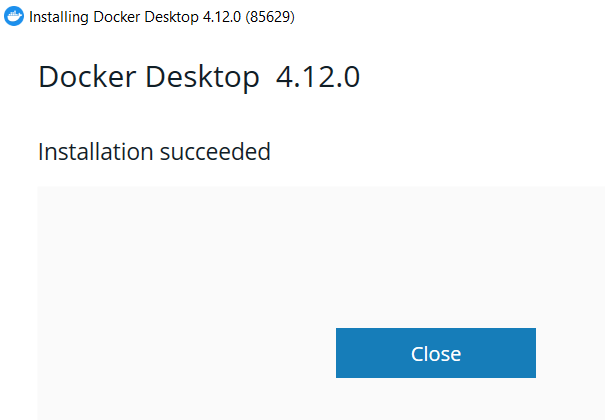
<https://docs.docker.com/desktop/install/windows-install/>



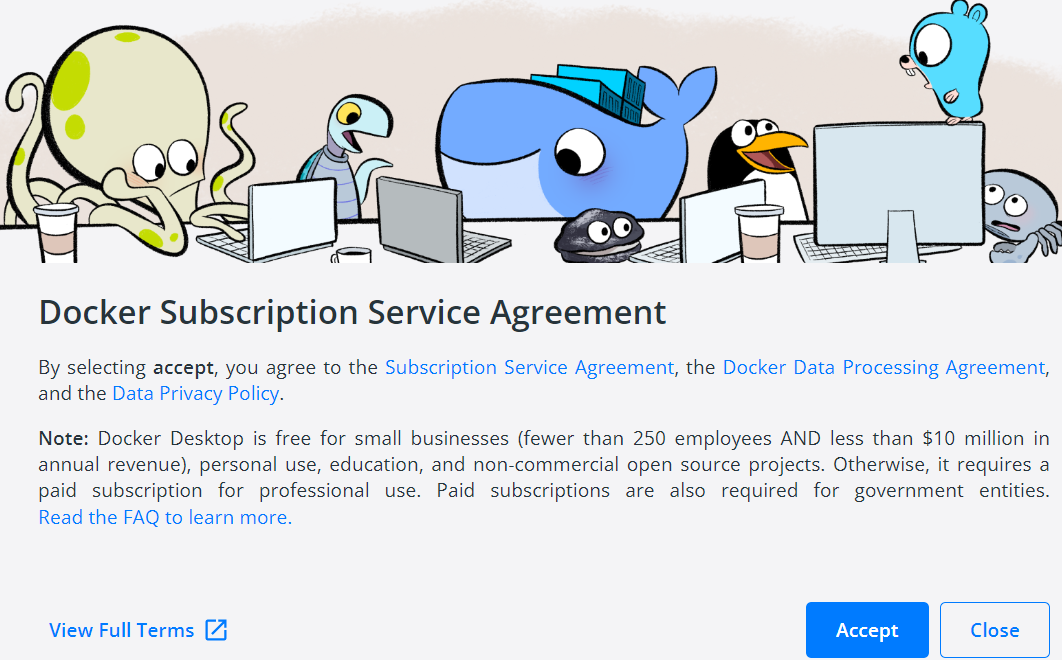
1. **Double-click the Docker Desktop Installer from your Downloads folder**
2. **Click "Install anyway" if warned the app isn't Microsoft-verified**
3. **Click "OK" to Add a shortcut to the Desktop**



1. **Click "Close" when you see Installation succeeded message**

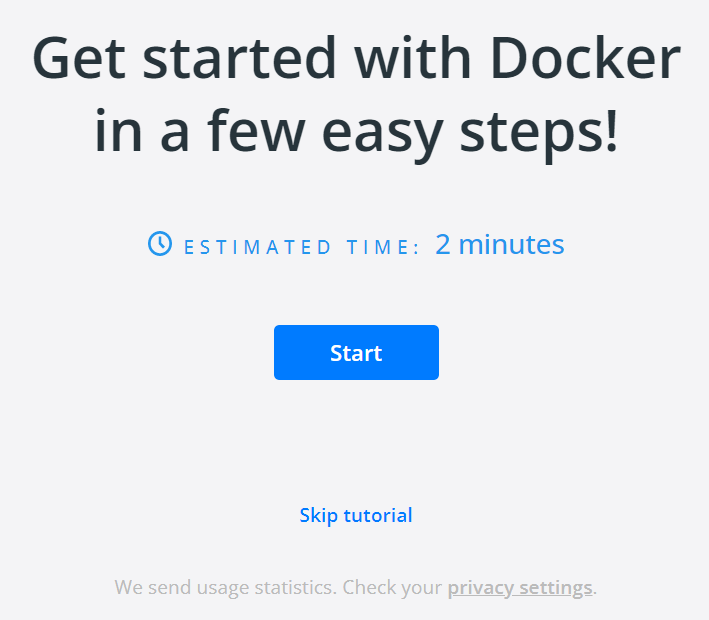


1. **Double-click the Docker Desktop icon on your Desktop**
2. **Accept the Docker Service Agreement**



1. **Docker Desktop will launch for the first time**

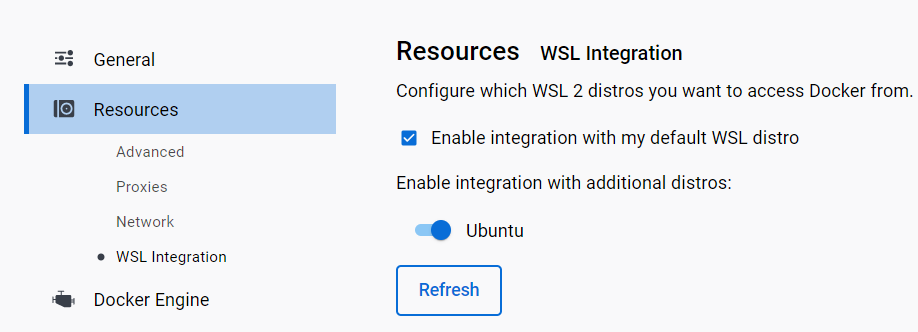
Docker Desktop will launch and present you with a tutorial. You are free to skip this.

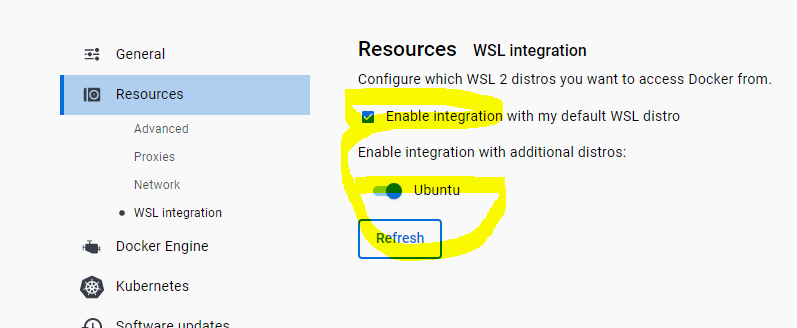


1. **Ensure that WSL Integration is Enabled**

In Docker Desktop, click the Settings **Gear** icon. Then choose **Resources,** and finally **WSL Integration**. Make sure that the **Enable Integration with my default WSL distro** is checked. Also, if you are using multiple distributions, make sure that these additional distros are toggled on:

## Check if docker is running or not

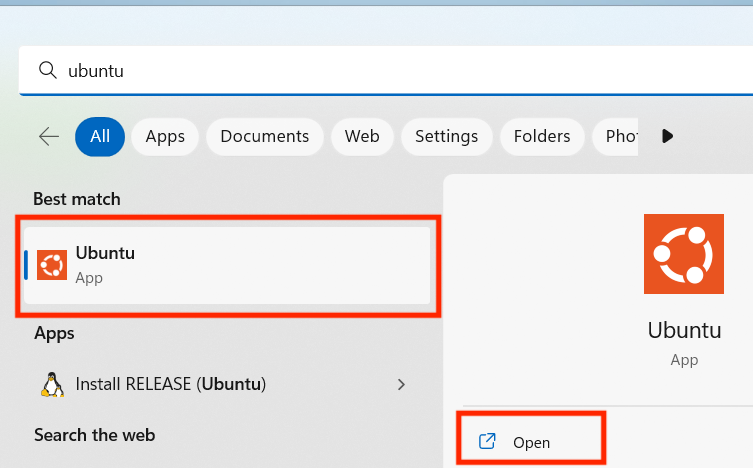




If this is not enabled, u will get error message saying is docker running

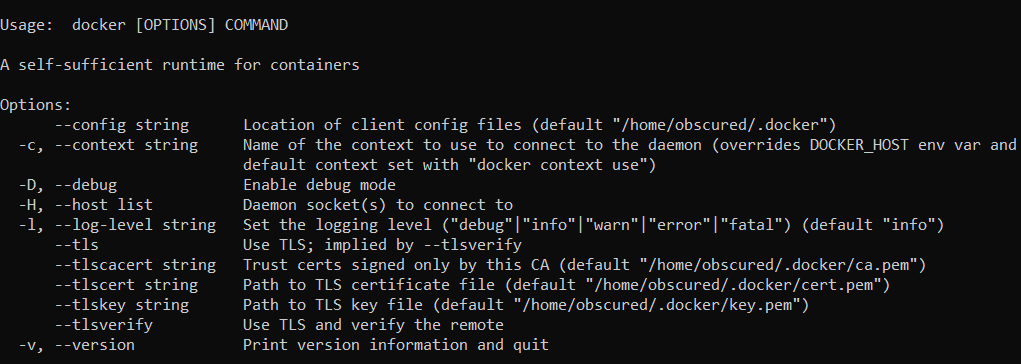
1. **Open your Distro**

Using the Windows Search feature in the toolbar, type the name of your distribution (by default it is Ubuntu) and click **Open:**



1. **Check that Docker is working**

Using the terminal for your distro, run the docker command. If all is well you should see some helpful instructions in the output similar to below:



1. **Log in to Docker**

Using the terminal for your distro, run the docker login command. You will be prompted to enter the username and password (or your Personal Access Token) you created earlier when registering for a DockerHub account. Once you see **Login Succeeded**, the setup is complete and you are free to continue to the next lecture.

**Appendix**

A significant difference when using WSL is that you will need to create and run your project files from within the Linux filesystem, not the Windows filesystem. This will be very important in later lectures when we cover volumes.

You can access your Linux system by using the Windows Search feature in the toolbar and typing the name of your distribution (by default it is Ubuntu) and clicking open (see step #16 above).  This terminal should automatically open to the home directory on the Linux filesystem

Going forward, all Docker commands should be run within **WSL** and not on the Windows file system

# Creating Images

**Buildkit for Docker Desktop**

Most students who have the most recent versions of Docker will now have **Buildkit** enabled by default. If so, you will notice a slightly different output in your terminal when building from a Dockerfile.

The main difference for students will be the final step in the build process. As shown in the lecture the last step would say:

1. ---> fc60771eaa08
2. Successfully Built fc60771eaa08

Now, with Buildkit, the final step would say:

1. => => exporting layers
2. 0.0s => => writing image sha256:ee59c34ada9890ca09145cc88ccb25d32b677fc3b61e921  0.0s

Both **fc60771eaa08**and**ee59c34ada9890ca09145cc88ccb25d32b677fc3b61e921**are the resulting image ID's that you would use to run a container.

eg:

1. docker run fc60771eaa08

or

1. docker run ee59c34ada9890ca09145cc88ccb25d32b677fc3b61e921

#### **Disabling Buildkit to match course output**

If you wish to disable the Buildkit feature so that you can more accurately follow along with the course, do the following:

1. Click the Docker Icon in the system tray (Windows) or menu bar (macOS)
2. Select **Preferences**
3. Select **Docker Engine**
4. Change buildkit from **true** to **false**
   1. {
   2. "features": {
   3. "buildkit": false
   4. },
   5. "experimental": false
   6. }
5. Apply and Restart.

#### **Buildkit Features and Documentation**

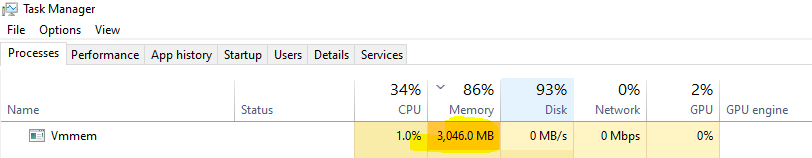
If you want to learn more about features Buildkit has to offer, please check out the following pages:

<https://docs.docker.com/develop/develop-images/build_enhancements/>

<https://docs.docker.com/engine/reference/commandline/build/#specifying-external-cache-sources>

<https://www.docker.com/blog/advanced-dockerfiles-faster-builds-and-smaller-images-using-buildkit-and-multistage-builds/>

# Docker consuming high memory



Vnmem=related to docker, virtual machine memory , sometimes its taking more than 5GB, to avoid that

Create a file called “.wslconfig” in “C:\Users\Manideep”this path with below contents

That its ,now docker will not consume more than what is specified.now the computer will not freeze

**[wsl2]**

**memory=3GB # Limits VM memory in WSL 2 up to 3GB**

**processors=4 # Makes the WSL 2 VM use two virtual processors**