

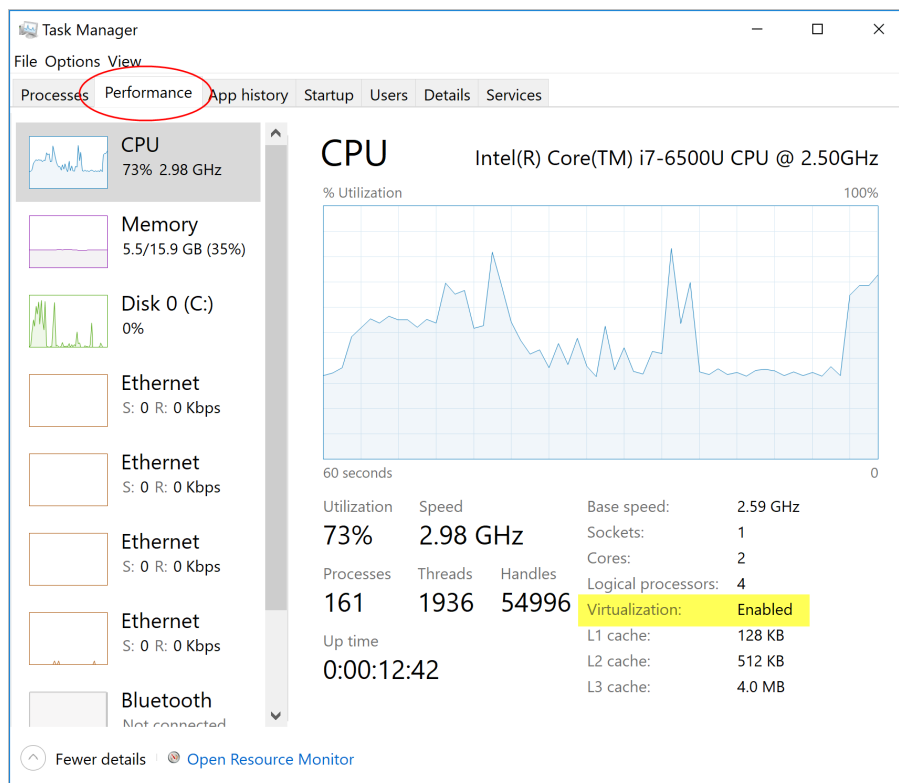
Instructions for running hive and spark on Windows machines using Docker

Prerequisites

Before you can install Docker on your PC, please make sure your computer meets the following conditions:

1. 64-bit processor with Second Level Address Translation (SLAT)
2. 4GB system RAM (minimum)
3. BIOS-level hardware virtualization support must be enabled in the BIOS settings. You can check it in the Performance tab on the Task Manager (like below). If it is not enabled, you can follow this guide to enable it:

<https://2nwiki.2n.cz/pages/viewpage.action?pageId=75202968>



4. Enable the WSL 2 feature on Windows. (We will walk you through the steps in the following section)

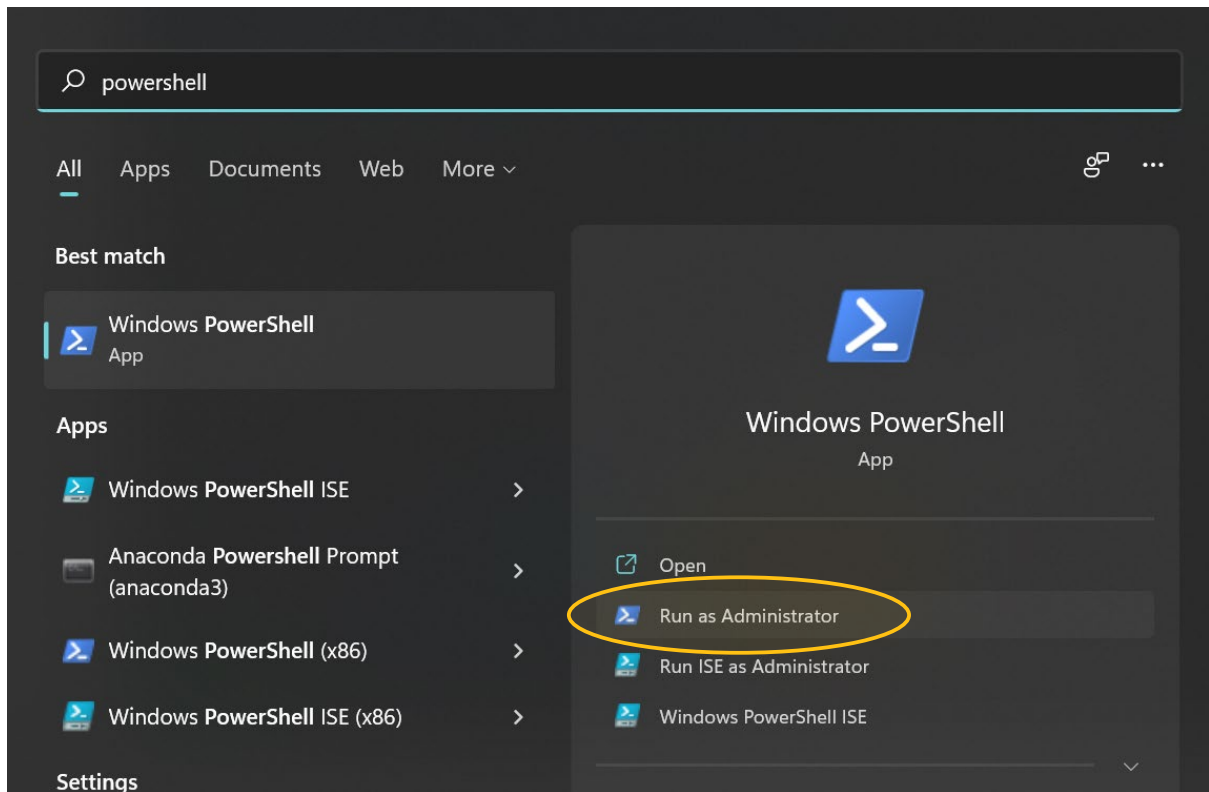
Enabling WSL 2 on Windows

If you have already enabled WSL 2 on your machine, please skip this section.

To enable WSL 2, we will follow the official guide from Microsoft here

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

1. Search for PowerShell or Command Prompt on your machine and click "Run as administrator" (like below). Click "Yes" at the User Account Control screen.



2. Paste the command `wsl --install` into your PowerShell and press enter:

If you see the following screen, that means you have successfully installed WSL 2. Restart your computer.

```
Administrator: Command Promj  X  +  -  □  X
Microsoft Windows [Version 10.0.19043.1151]
(c) Microsoft Corporation. All rights reserved.

C:\>wsl --install
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
Windows Subsystem for Linux has been installed.
Downloading: WSL Kernel
Installing: WSL Kernel
WSL Kernel has been installed.
Downloading: Ubuntu
The requested operation is successful. Changes will not be effective until the system is rebooted.

C:\>|
```

If you see WSL help text (like image below), please try running `wsl --list --online` to see a list of available distros and run `wsl --install -d <DistroName>` to install a distro.

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Windows\system32> wsl --install
Copyright (c) Microsoft Corporation. All rights reserved.

Usage: wsl.exe [Argument] [Options...] [CommandLine]

Arguments for running Linux binaries:

    If no command line is provided, wsl.exe launches the default shell.

    --exec, -e <CommandLine>
        Execute the specified command without using the default Linux shell.

    --
        Pass the remaining command line as is.
```

```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

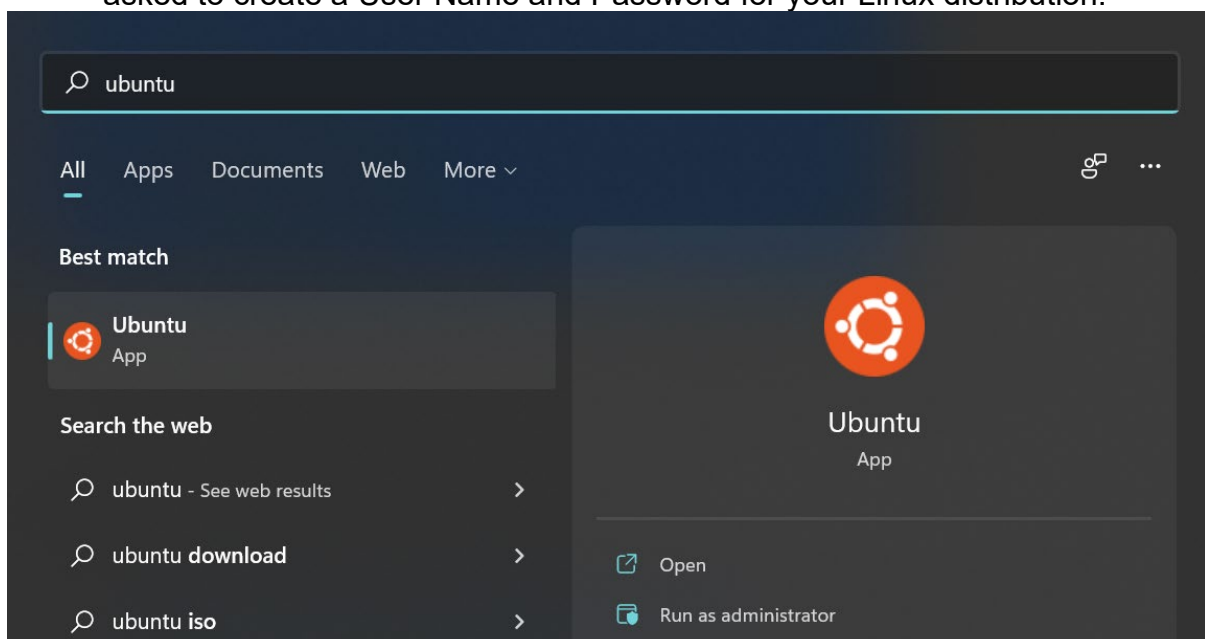
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

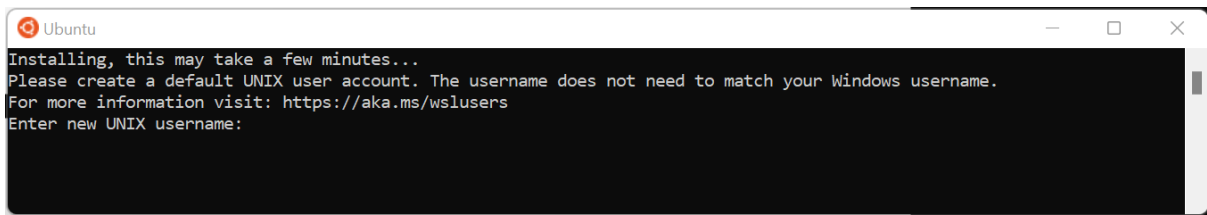
PS C:\Windows\system32> wsl --list --online
The following is a list of valid distributions that can be installed.
Install using 'wsl --install -d <distro>'.

NAME                FRIENDLY NAME
Ubuntu              Ubuntu
Debian              Debian GNU/Linux
kali-linux           Kali Linux Rolling
openSUSE-42          openSUSE Leap 42
SLES-12             SUSE Linux Enterprise Server v12
Ubuntu-16.04         Ubuntu 16.04 LTS
Ubuntu-18.04         Ubuntu 18.04 LTS
Ubuntu-20.04         Ubuntu 20.04 LTS
PS C:\Windows\system32> wsl --install -d Ubuntu
Downloading: Ubuntu
Installing: Ubuntu
Ubuntu has been installed.
Launching Ubuntu...
PS C:\Windows\system32>
```

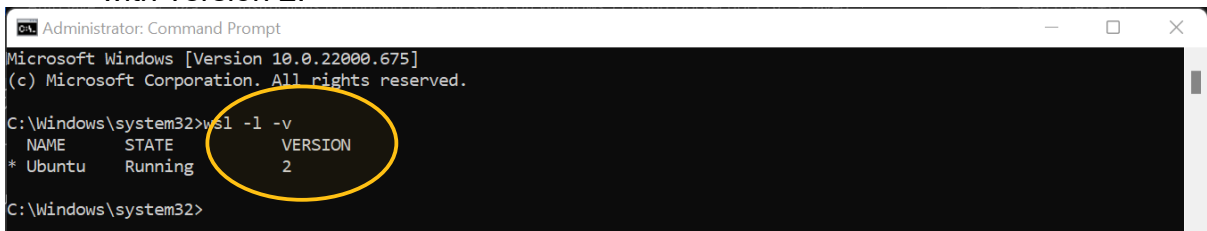
Once the process of installing your Linux distribution with WSL is complete, we will follow Microsoft's instructions for setting up your Linux username and password: <https://docs.microsoft.com/en-us/windows/wsl/setup/environment#set-up-your-linux-username-and-password>

3. Open the distribution (Ubuntu by default) using the Start menu. You will be asked to create a User Name and Password for your Linux distribution.



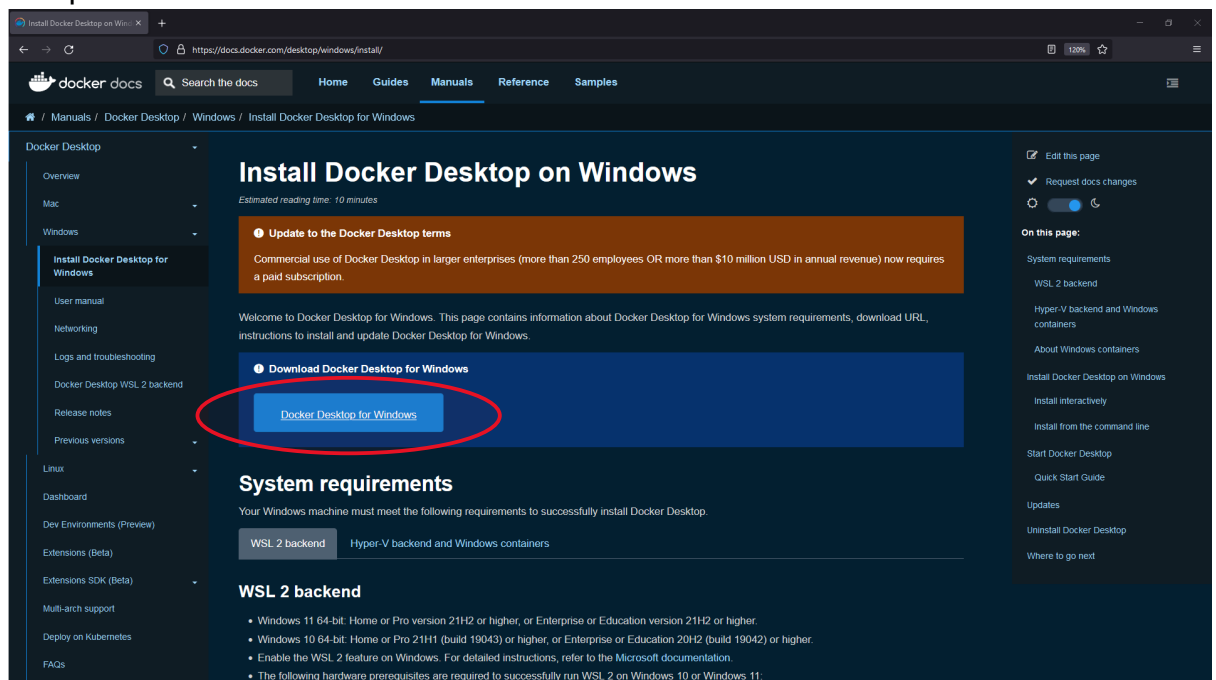


4. Choose your username and password. This User Name and Password is specific to each separate Linux distribution that you install and has no bearing on your Windows user name.
5. Finally, to make sure you have WSL 2 installed correctly, type in `wsl -l -v` in PowerShell or Windows Command Prompt and you should see your distro with version 2.

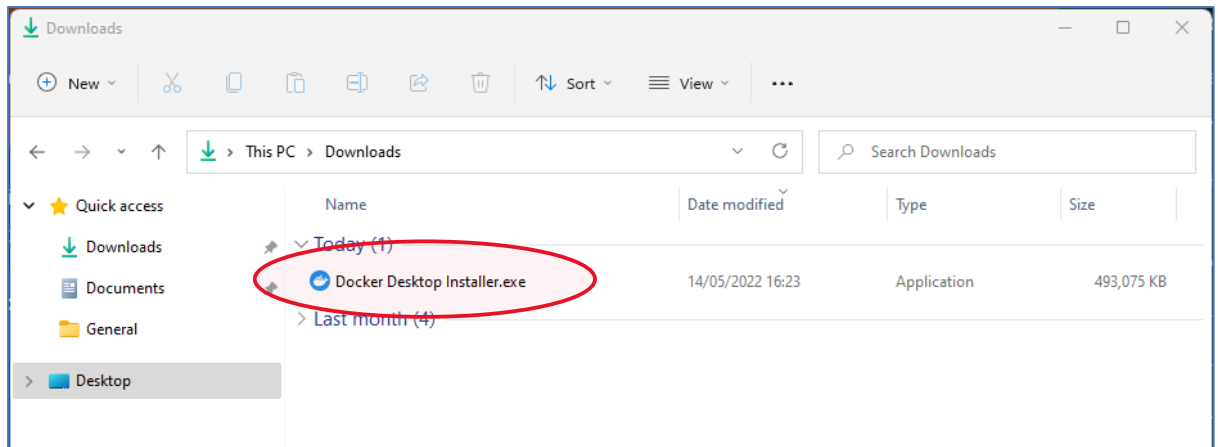


Installing Docker engine on your Windows PC

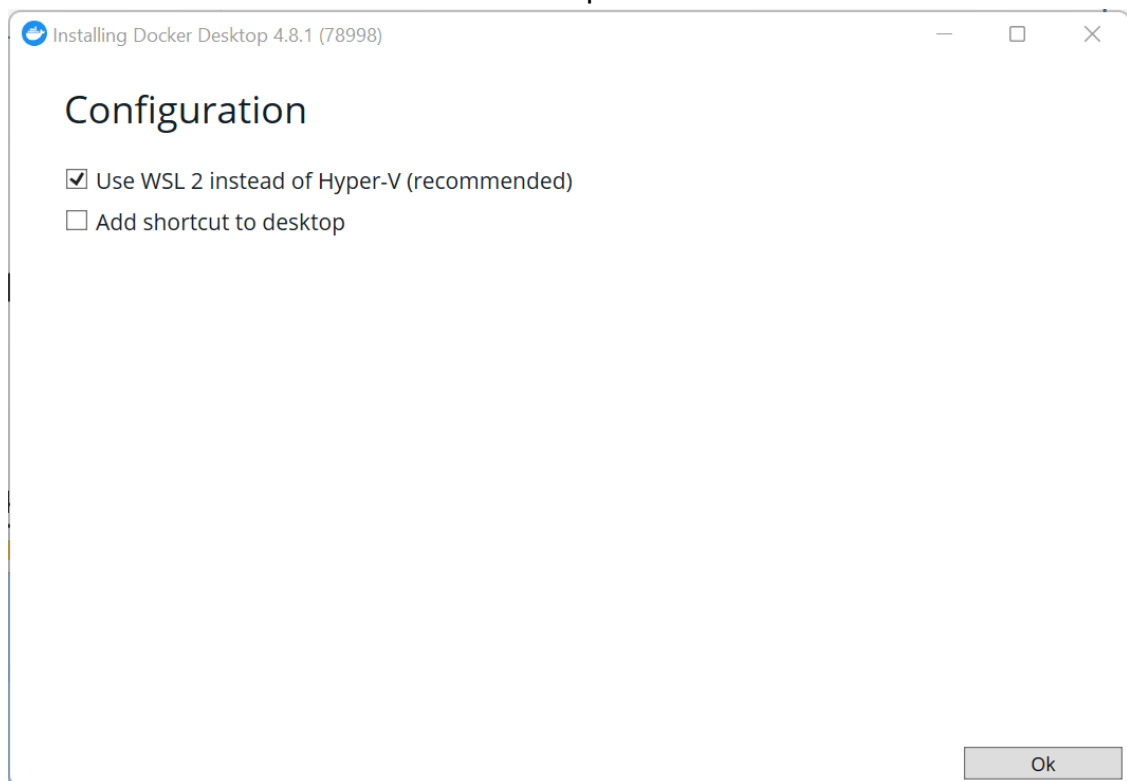
1. Please go to the following link and download the Docker engine for Windows <https://docs.docker.com/desktop/windows/install/>
2. Click on “Docker Desktop for Windows” button and wait for the download to complete.



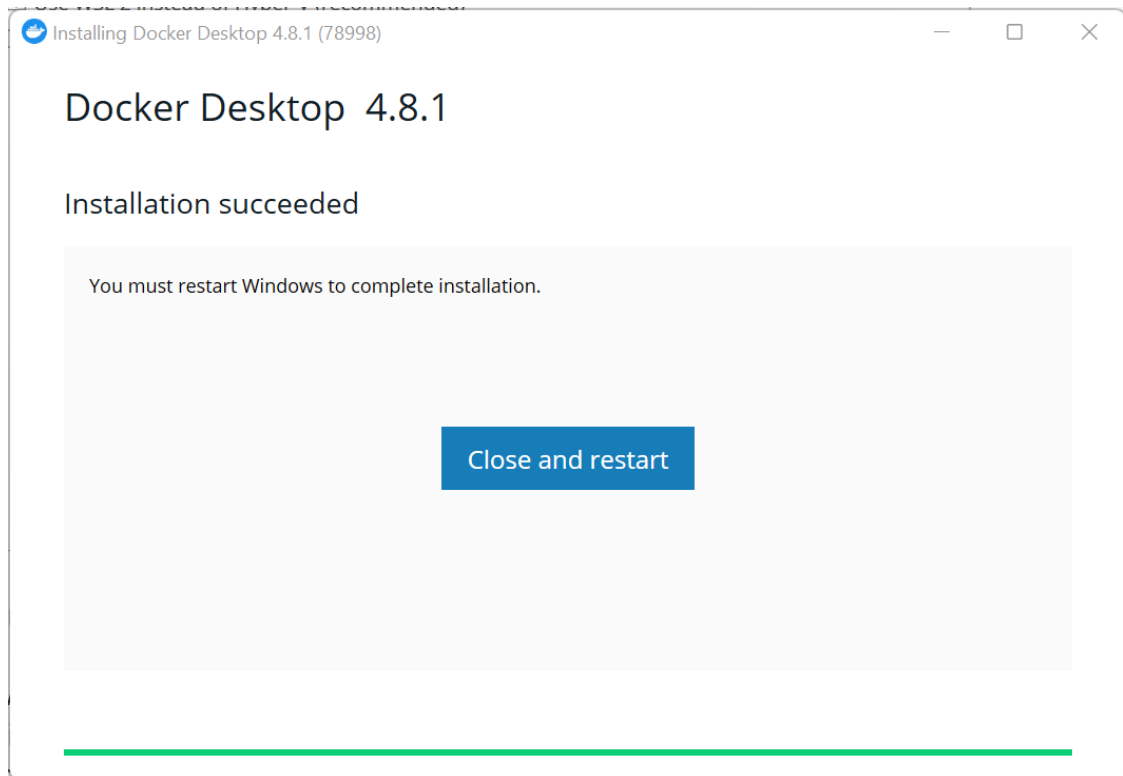
3. Now go to “Downloads” directory and double click on “Docker Desktop Installer.exe” file to open it.



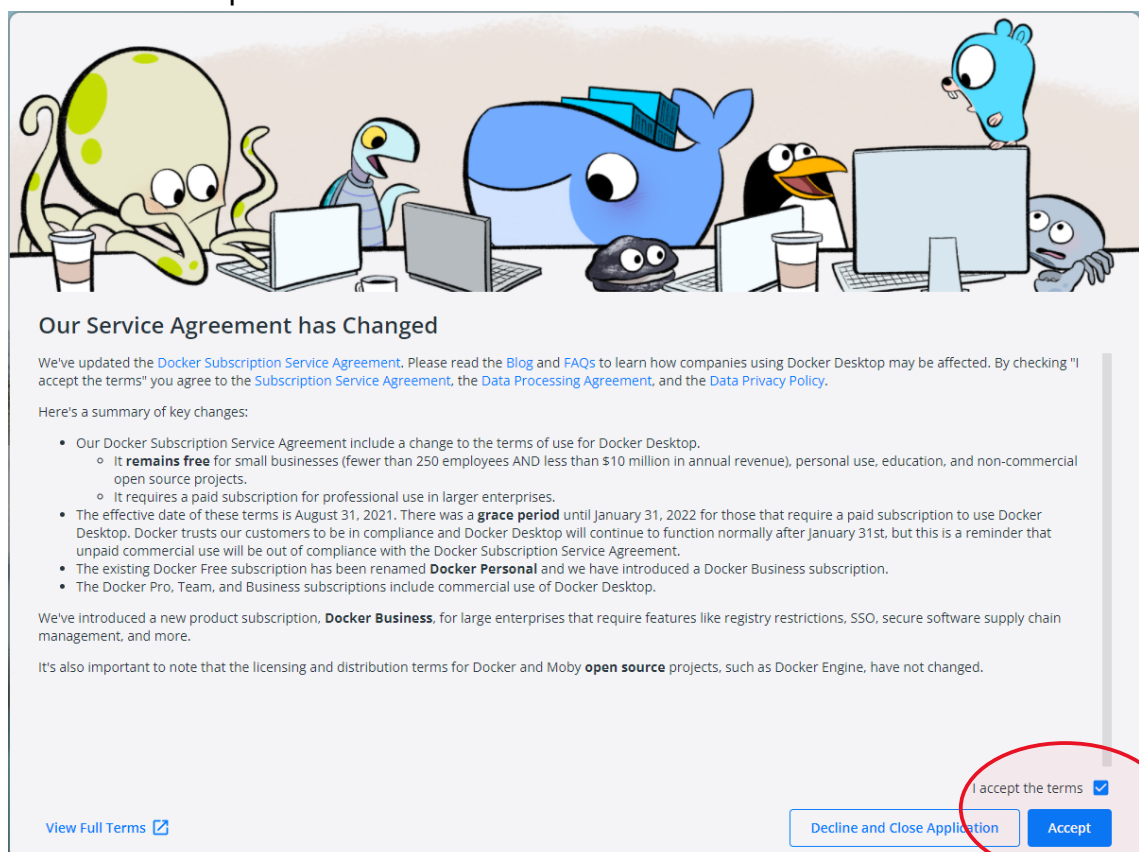
4. Click “Yes” at the User Account Control screen to allow the app to make changes to your machine. You will then see this window. You should leave the option “Use WSL 2 instead of Hyper-V” ticked (as recommended). Then click “Ok”. It will take a few minutes to unpack and install the files.



After it's done, you should see this screen. Click the “Close and restart” button and restart your Windows machine.



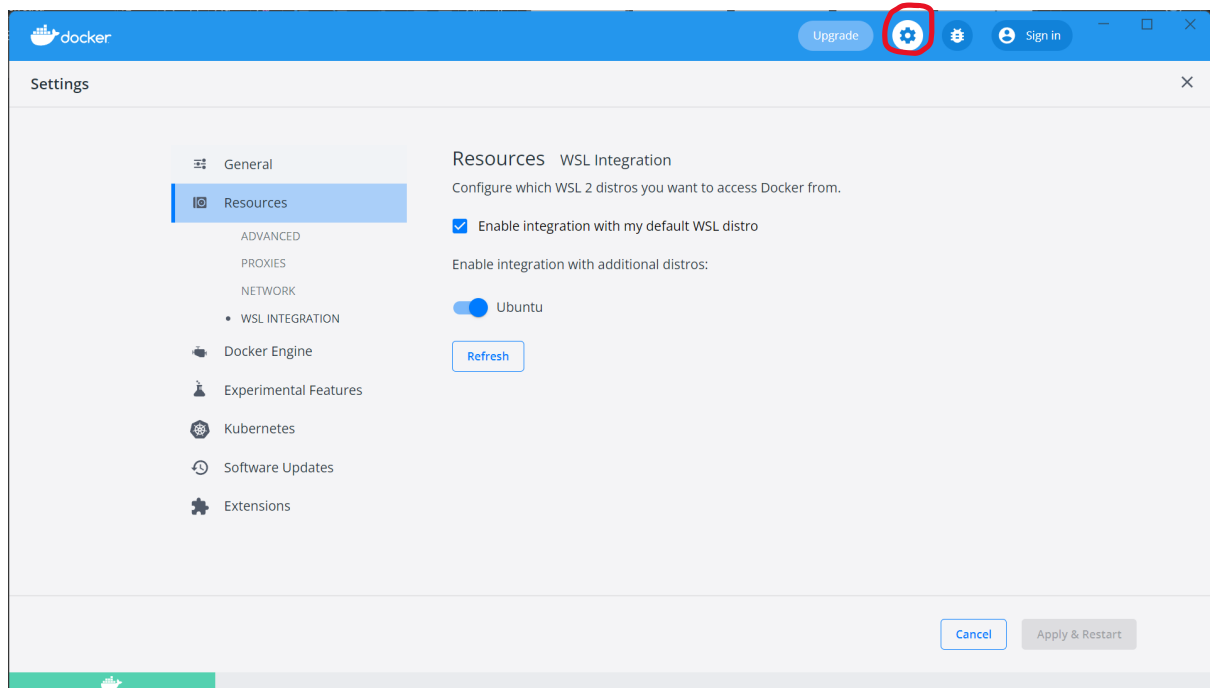
5. After restarting your machine, you should be greeted with the following screen. Tick “I accept the term” and click “Accept”. If you don’t see the screen after logging in, you can search for “Docker” on your computer and select “Docker Desktop”



6. If your admin account is different to your user account, you must add the user to the docker-users group. Run **Computer Management** as an administrator and navigate to **Local Users and Groups > Groups > docker-users**. Right-click to add the user to the group. Log out and log back in for the changes to take effect.

Integrating WSL Ubuntu with Docker

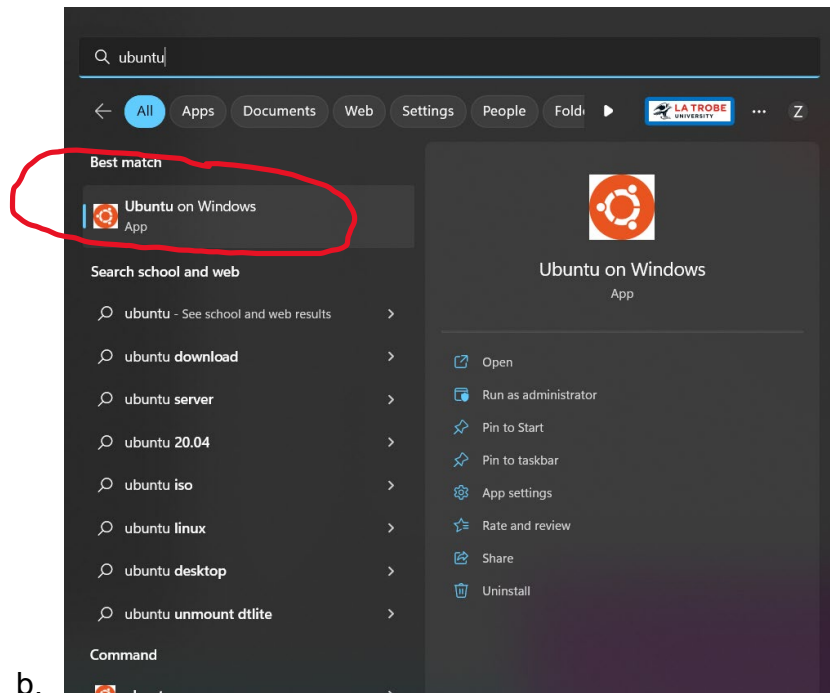
Open Docker Desktop and go to **Settings > Resources > WSL Integration**. Then toggle on for Ubuntu (or your distro of choice). Click "Apply & Restart".



Pulling the Docker image and running the container

Let's pull the Docker-image now to start working on your labs.

1. Download the **run.sh** script from here:
 - a. https://drive.google.com/file/d/1dgoFxChGBMoqYKoSpOwaK_zlStEiF13w/view?usp=share_link
2. Create a directory on your Documents folder called BDC_files
 - a. You can put your BDC lab and assignment files here
3. Copy the **run.sh** script into the BDC_files folder.
4. Open Ubuntu terminal using the Start menu (like we have done above).
 - a. You can type "ubuntu" on the search window and then select the Ubuntu terminal like the following (note for the future you will always need to have docker desktop open before you go into the terminal):



b.

5. Move to the directory that contains the **run.sh** script, by typing the command below:

```
cd /mnt/c/Users/<your username>/Documents/BDC_files
```

- Please substitute <your username> with your windows username

6. Then type the following to see what options you can use when running docker:

```
bash run.sh
```

Use “**bash run.sh /mnt/c/Users/<your username>/Documents/BDC_files local**” if you want to run **local** mode. This will run a lot faster than **yarnmode**. I suggest you use this mode for everything, except for one of the early exercises in lab 2 where you need to look at the Map Reduce jobs where you should use the **yarnmode** instead.

Use “**bash run.sh /mnt/c/Users/<your username>/Documents/BDC_files yarnmode**” if you want to run using the yarn scheduler. This is the pseudo cluster mode it will take much more resources since it tries to mimic a real cluster. So, it will be slower, but you can see your map reduce job history.

Finally, the last parameter is if you want to limit the amount of RAM you allocate to the **docker** image. I suggest you select at least 2GB, or just leave it blank so the default RAM amount will be used.

7. Wait for docker to finish pulling the image and for the container to start. This may take quite a long time if you have slow internet. However, docker will store the image in your computer so next time it will be much faster since it will not need to pull the image again.
8. Now that the docker container is running. The terminal should be inside the container. We can now try out a Hive script to see if everything is working as they should.

9. Download the following test_vm.zip file and **copy** and **unzip** it into the **BDC_files** directory.
10. Now change to the **labfiles** directory (by typing: **cd labfiles**). This will take you to the **BDC_files** directory. Now change to the **test_vm** directory (by typing: **cd test_vm**).
 - a. https://drive.google.com/file/d/1f_9_5KEunR8_Ew6Z2v0LQ1Ph783gavX/view?usp=share_link

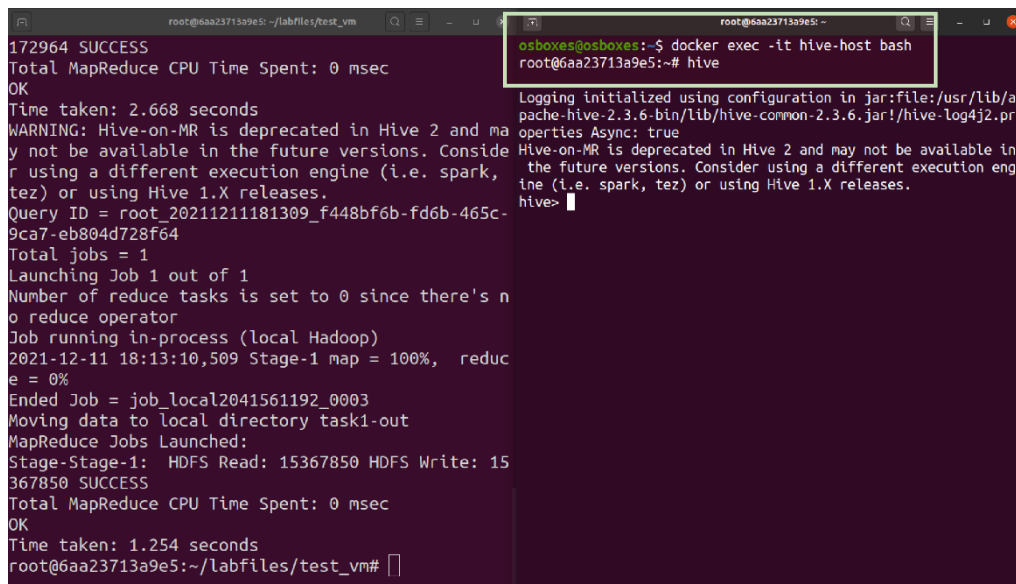
Then type the following command to run hive to test if your machine is fast enough to run the labs: **hive -f t1-wordcount.hql** (command also included in screenshot below)

If everything finishes within 1 or 2 mins then it should be good enough to do the labs. If this takes more than 5 mins to finish, then your machine maybe too slow.

11. Now let's start another **Ubuntu terminal window** from the Start menu and use that for the hive interpreter. For the second window we will use the following command.

docker exec -it hive-host bash

Then open the hive interpreter using the **hive** command. See the right window



```
root@6aa23713a9e5: ~/labfiles/test_vm
172964 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 2.668 seconds
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = root_20211211181309_f448bf6b-fd6b-465c-9ca7-eb804d728f64
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Job running in-process (local Hadoop)
2021-12-11 18:13:10,509 Stage-1 map = 100%, reduce = 0%
Ended Job = job_local2041561192_0003
Moving data to local directory task1-out
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 15367850 HDFS Write: 15367850 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 1.254 seconds
root@6aa23713a9e5: ~/labfiles/test_vm#
```

```
osboxes@osboxes:~$ docker exec -it hive-host bash
root@6aa23713a9e5:~# hive
Logging initialized using configuration in jar:file:/usr/lib/apache-hive-2.3.6-bin/lib/hive-common-2.3.6.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive>
```

below:

12. Now on the **right** window you can type the following in the hive interpreter to see what tables were created.

show tables;

If everything works correctly you should see three tables (myinput, mywords and wordcount)

```
root@6aa23713a9e5: ~/labfiles/test_vm
172964 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 2.668 seconds
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
Query ID = root_20211211181309_f448bf6b-fd6b-465c-9ca7-eb804d728f64
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Job running in-process (local Hadoop)
2021-12-11 18:13:10,509 Stage-1 map = 100%, reduce = 0%
Ended Job = job_local2041561192_0003
Moving data to local directory task1-out
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 15367850 HDFS Write: 15367850 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 1.254 seconds
root@6aa23713a9e5:~/labfiles/test_vm#

osboxes@osboxes:~$ docker exec -it hive-host bash
root@6aa23713a9e5:~# hive

Logging initialized using configuration in jar:file:/usr/lib/apache-hive-2.3.6-bin/lib/hive-common-2.3.6.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.
hive> show tables;
OK
myinput
mywords
wordcount
Time taken: 2.296 seconds, Fetched: 3 row(s)
hive>
```

Note the first time you use the hive interpreter it will be slow. But if you run it again it should be super-fast. Try the above command again in the hive interpreter.

If you do not currently have a good code editor. I recommend that you install the **vscode** to edit your code files. You can get **vscode** for windows from the link below (please download the stable version):

<https://code.visualstudio.com/>

After you finish your session and type **exit** in the terminal, the containers should be automatically deleted. However, if it is not automatically deleted then you can manually delete it in the docker desktop by clicking the trash icon below (delete both the hue and hive-host containers):

