

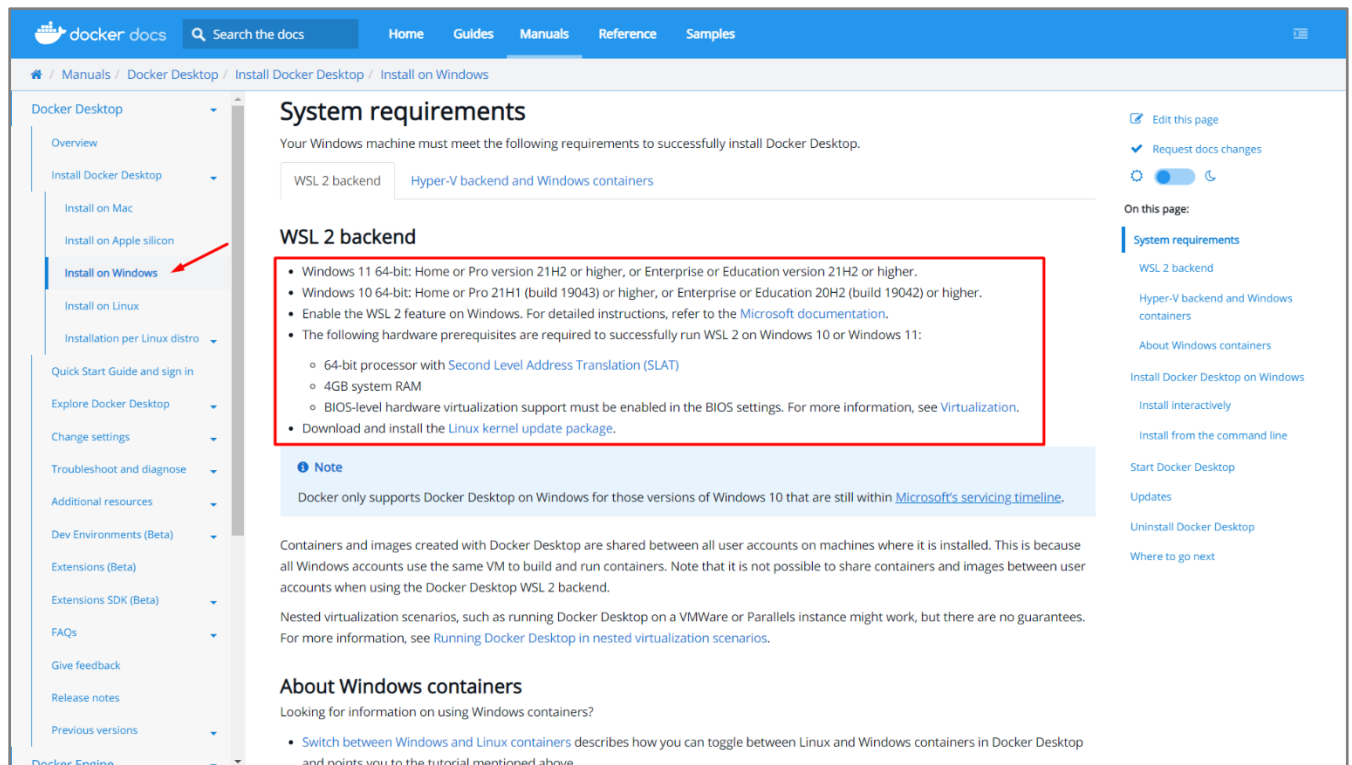
# Docker Desktop Installation Guide

## Download Docker Desktop on Windows

First, you will need to open the Docker documentation <https://docs.docker.com/desktop/>.

Choose **"Windows"** from the menu on the left side of the screen. Before installing make sure that your machine **matches the needed requirements** to use Docker Desktop.

For example, if you use Windows 10+ you could **directly** download and install Docker Desktop because the platform runs natively on Windows 10+:



The screenshot shows the Docker documentation website for Docker Desktop installation on Windows. The left sidebar has a menu with 'Install on Windows' highlighted. The main content area is titled 'System requirements' and lists the prerequisites for WSL 2 backend. A red box highlights the WSL 2 backend requirements.

**System requirements**

Your Windows machine must meet the following requirements to successfully install Docker Desktop.

WSL 2 backend    Hyper-V backend and Windows containers

**WSL 2 backend**

- Windows 11 64-bit: Home or Pro version 21H2 or higher, or Enterprise or Education version 21H2 or higher.
- Windows 10 64-bit: Home or Pro 21H1 (build 19043) or higher, or Enterprise or Education 20H2 (build 19042) or higher.
- Enable the WSL 2 feature on Windows. For detailed instructions, refer to the [Microsoft documentation](#).
- The following hardware prerequisites are required to successfully run WSL 2 on Windows 10 or Windows 11:
  - 64-bit processor with [Second Level Address Translation \(SLAT\)](#)
  - 4GB system RAM
  - BIOS-level hardware virtualization support must be enabled in the BIOS settings. For more information, see [Virtualization](#).
- Download and install the [Linux kernel update package](#).

**Note**

Docker only supports Docker Desktop on Windows for those versions of Windows 10 that are still within [Microsoft's servicing timeline](#).

Containers and images created with Docker Desktop are shared between all user accounts on machines where it is installed. This is because all Windows accounts use the same VM to build and run containers. Note that it is not possible to share containers and images between user accounts when using the Docker Desktop WSL 2 backend.

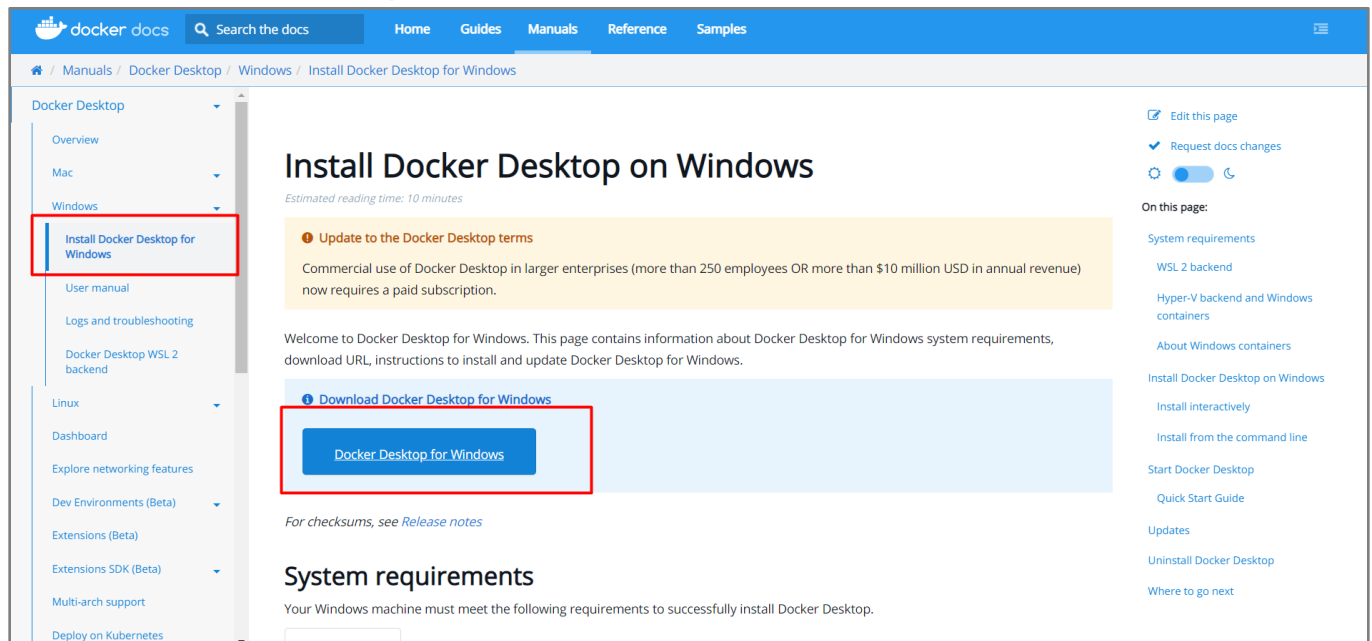
Nested virtualization scenarios, such as running Docker Desktop on a VMWare or Parallels instance might work, but there are no guarantees. For more information, see [Running Docker Desktop in nested virtualization scenarios](#).

**About Windows containers**

Looking for information on using Windows containers?

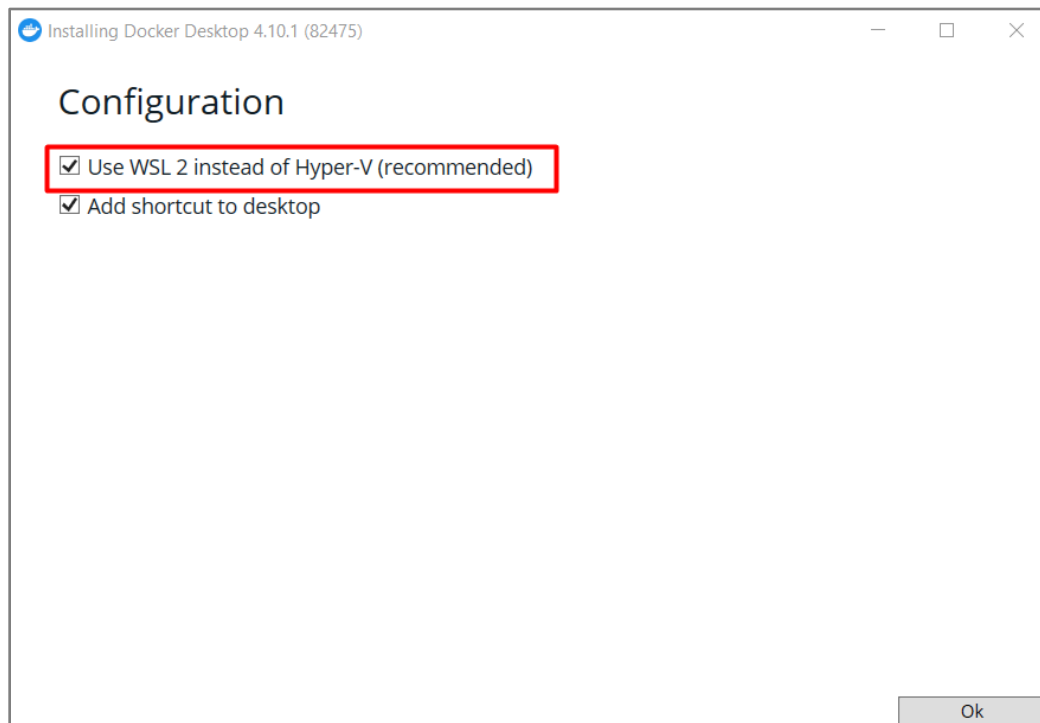
- [Switch between Windows and Linux containers](#) describes how you can toggle between Linux and Windows containers in Docker Desktop and points you to the tutorial mentioned above.

## Download the **Docker Desktop Installer**:

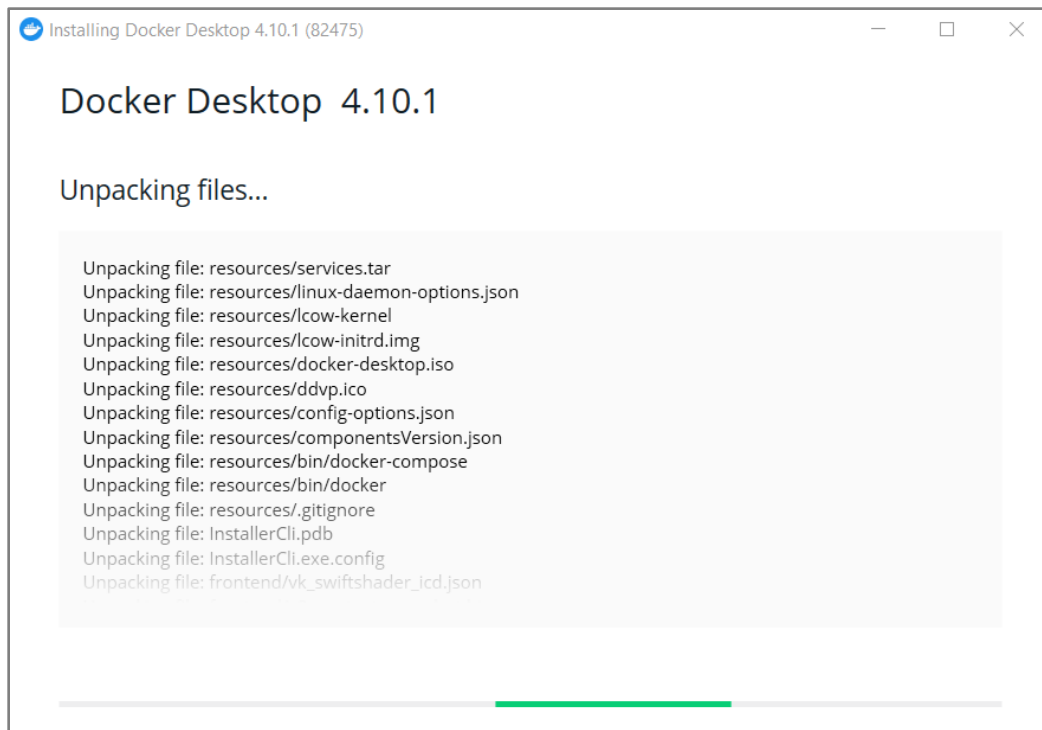


**Double-click Docker Desktop Installer.exe** to run it (for this example, Docker Desktop 4.10.1 will be installed; you do **NOT** need to download the same version of the software - it would be best to download the newest one).

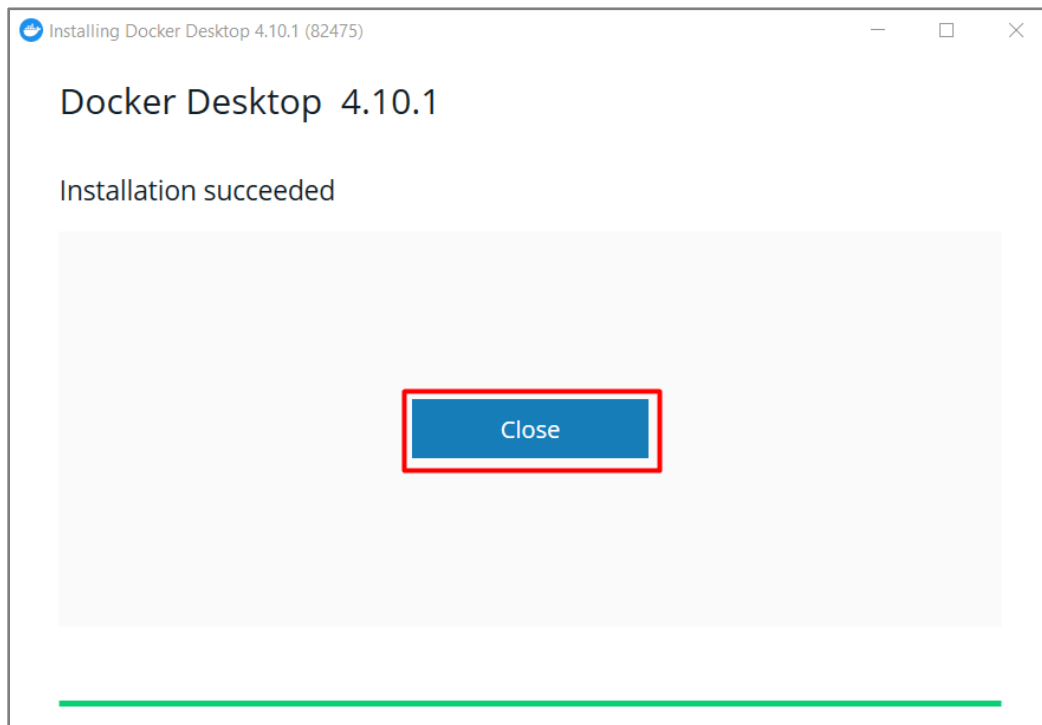
When prompted, ensure the **"Use WSL 2 instead of Hyper-V"** option on the Configuration page is **selected** or not depending on your choice of backend. If your system only supports one of the two options, you will **NOT** be able to select which backend to use:



The installation could take couple of minutes:




When the installation is successful, click **[Close]** to complete the installation process:



Docker Desktop does **NOT start automatically** after installation. To start Docker Desktop **double-click on the "Docker Desktop" icon**. First, it will display the **Docker Subscription Service**

**Agreement** window. Read the terms and click the checkbox to indicate that you accept the updated terms and then click **Accept** to continue:



### Our Service Agreement has Changed

We've updated the [Docker Subscription Service Agreement](#). Please read the [Blog](#) and [FAQs](#) to learn how companies using Docker Desktop may be affected. By checking "I accept the terms" you agree to the [Subscription Service Agreement](#), the [Data Processing Agreement](#), and the [Data Privacy Policy](#).

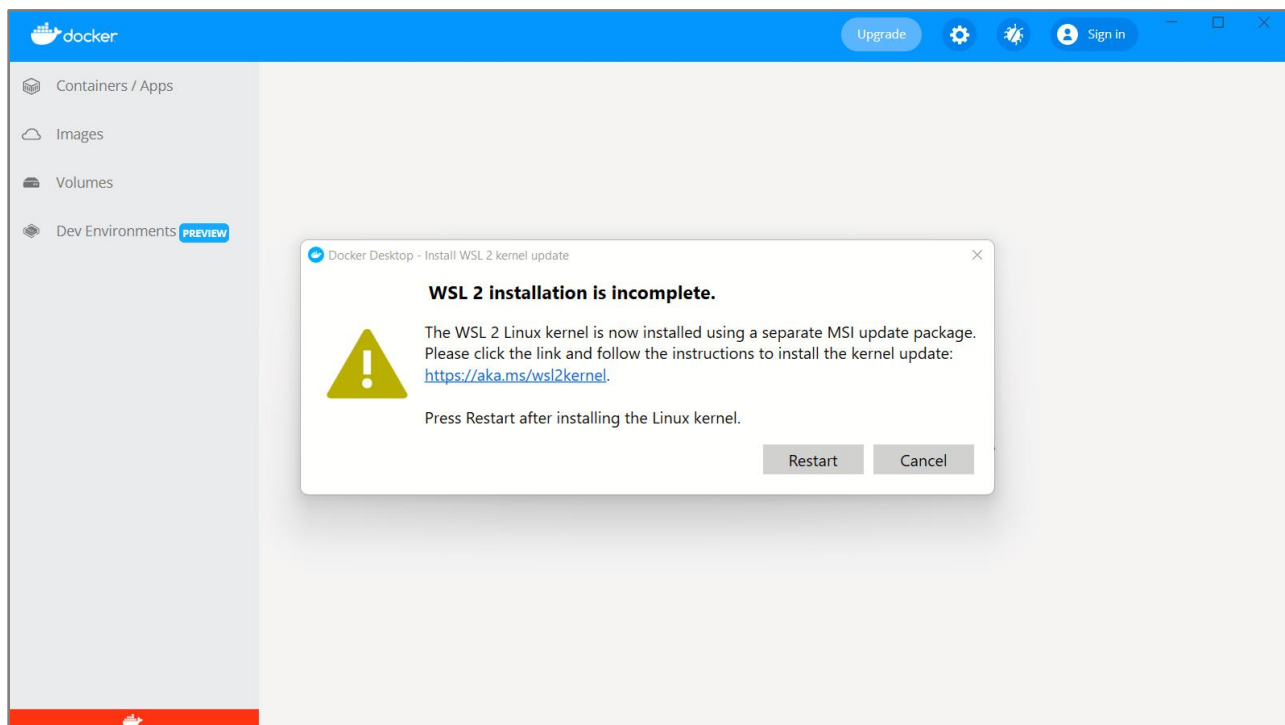
Here's a summary of key changes:

- Our Docker Subscription Service Agreement include a change to the terms of use for Docker Desktop.
  - It **remains free** for small businesses (fewer than 250 employees AND less than \$10 million in annual revenue), personal use, education, and non-commercial open source projects.
  - It requires a paid subscription for professional use in larger enterprises.
- The effective date of these terms is August 31, 2021. There was a **grace period** until January 31, 2022 for those that require a paid subscription to use Docker Desktop. Docker trusts our customers to be in compliance and Docker Desktop will continue to function normally after January 31st, but this is a

☒ I accept the terms

[View Full Terms](#) [Decline and Close Application](#) **Accept**

Docker Desktop starts after you accept the terms. You may see **the following pop-up message**:



If you do - **click on the link**, do **NOT close** the pop-up message, and install the **Subsystem for Linux update**:

The screenshot shows the Microsoft documentation page for 'Step 4 - Download the Linux kernel update package'. On the left is a navigation sidebar with links like 'WSL Documentation', 'Overview', 'Install', 'Manual install steps for older versions', 'Tutorials', 'Concepts', 'How-to', 'Frequently Asked Questions', 'Troubleshooting', and 'Release Notes'. The main content area has the title 'Step 4 - Download the Linux kernel update package'. Below the title, a red box highlights the instruction: '1. Download the latest package:' followed by a bullet point: '• WSL2 Linux kernel update package for x64 machines'. Below this is a pink 'Note' box with detailed instructions for ARM64 machines and non-English Windows versions, including commands like `systeminfo | find "System Type"` and `find "Systemtyp"`. On the right, an 'In this article' section lists steps from 'Step 1 - Enable the Windows Subsystem for Linux' to 'Install Windows Terminal (optional)', with 'Step 4 - Download the Linux kernel update package' highlighted.

If you installed it **correctly**, you should see the following window:

The screenshot shows the Docker Desktop application window. The title bar says 'Docker Desktop' and 'Upgrade plan'. The main content area has a large heading 'Get started with Docker in a few easy steps!'. Below the heading is a clock icon and the text 'ESTIMATED TIME: 2 minutes'. There is a prominent blue 'Start' button. Below that is a link 'Skip tutorial'. At the bottom, it says 'We send usage statistics. Check your [privacy settings](#).' The window has standard Windows window controls (minimize, maximize, close) and a 'Sign in' button in the top right corner.

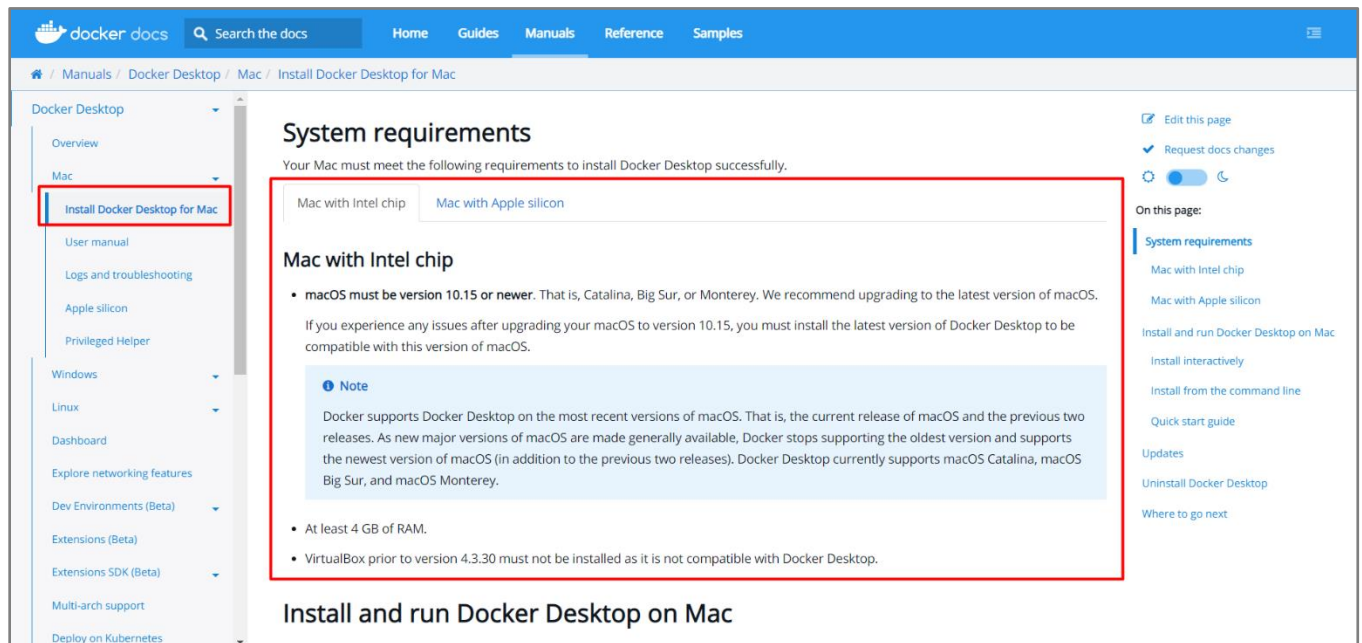
Congratulations! You are now successfully running Docker Desktop on Windows.

# Download Docker Desktop on Mac

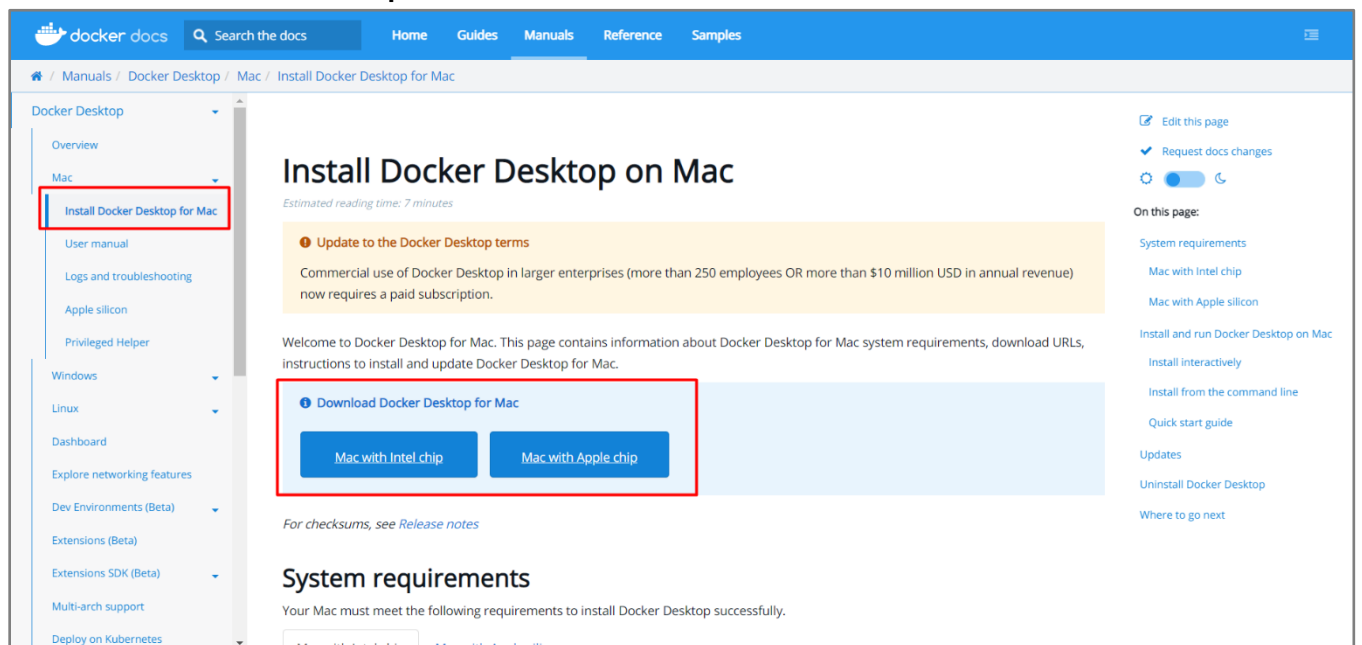
First, you will need to open the Docker documentation <https://docs.docker.com/desktop/>.

Choose **"Mac"** from the menu on the left side of the screen. Before installing make sure that your machine **matches the needed requirements** to use Docker Desktop.

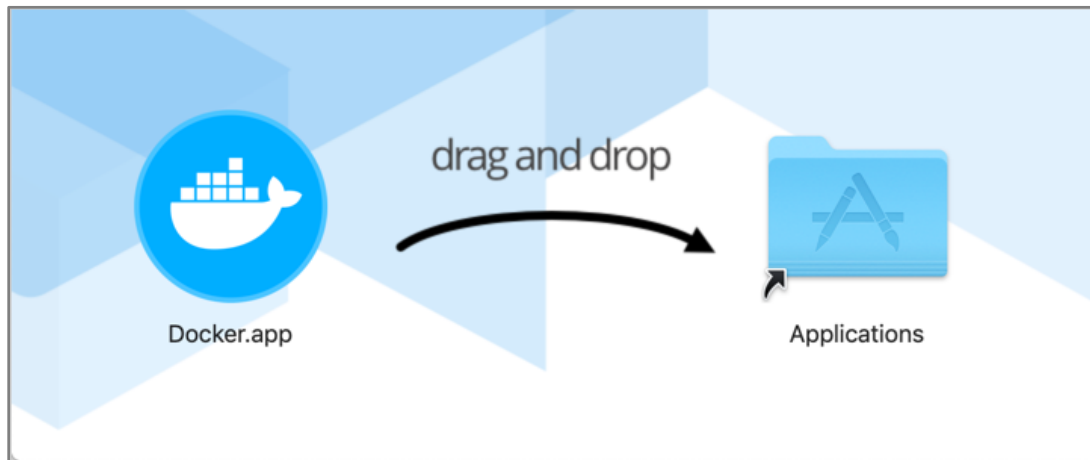
For example, if you use MacOS 10.15+ you could **directly** download and install Docker Desktop:



## Download the Docker Desktop Installer:

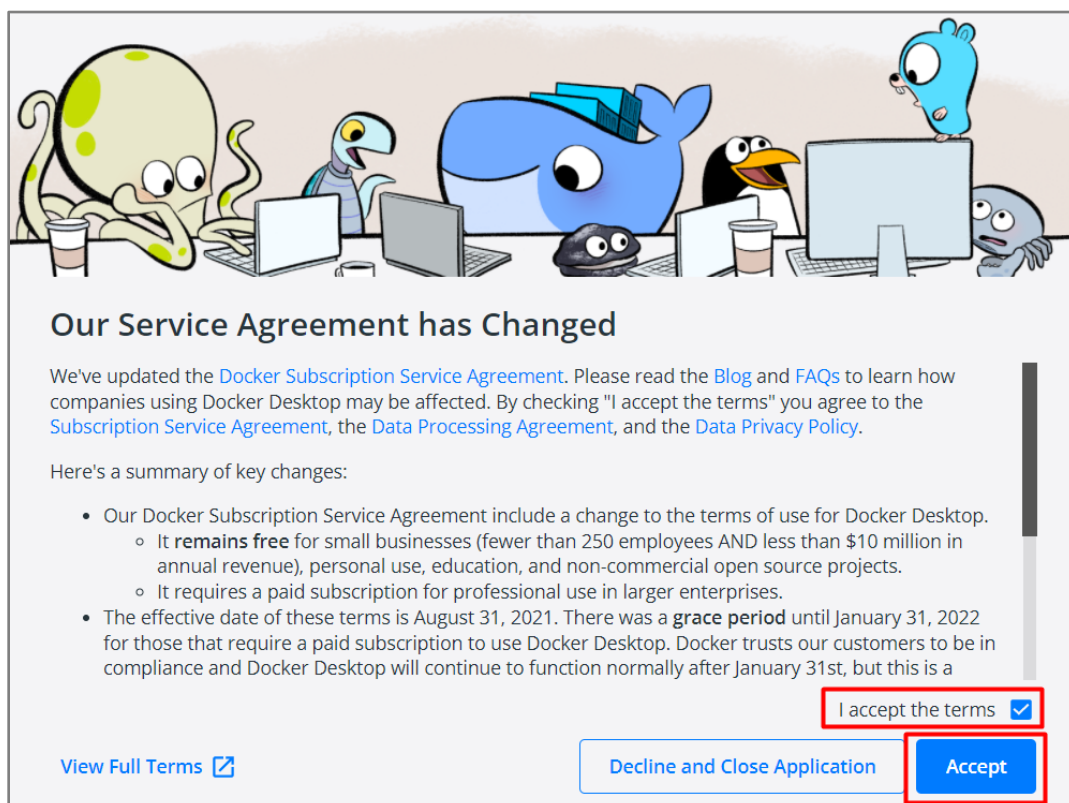


**Double-click Docker.dmg** to open the installer, then **drag the Docker icon to the Applications folder**:

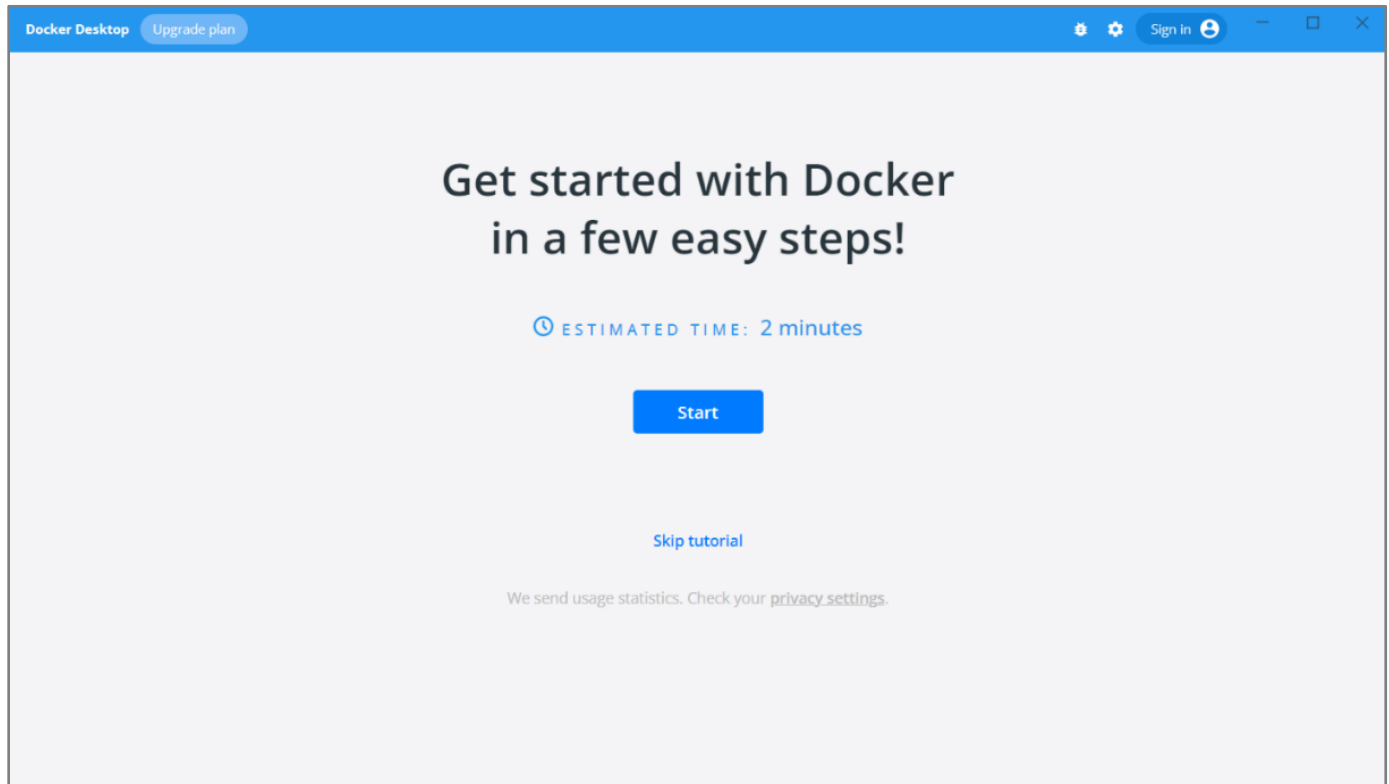


**Double-click Docker.app** in the Applications folder to start Docker.

First, it will display the **Docker Subscription Service Agreement** window. Read the terms and click the checkbox to indicate that you accept the updated terms and then click **Accept** to continue:



If you installed it **correctly**, you should see the following window:



Congratulations! You are now successfully running Docker Desktop on Mac.

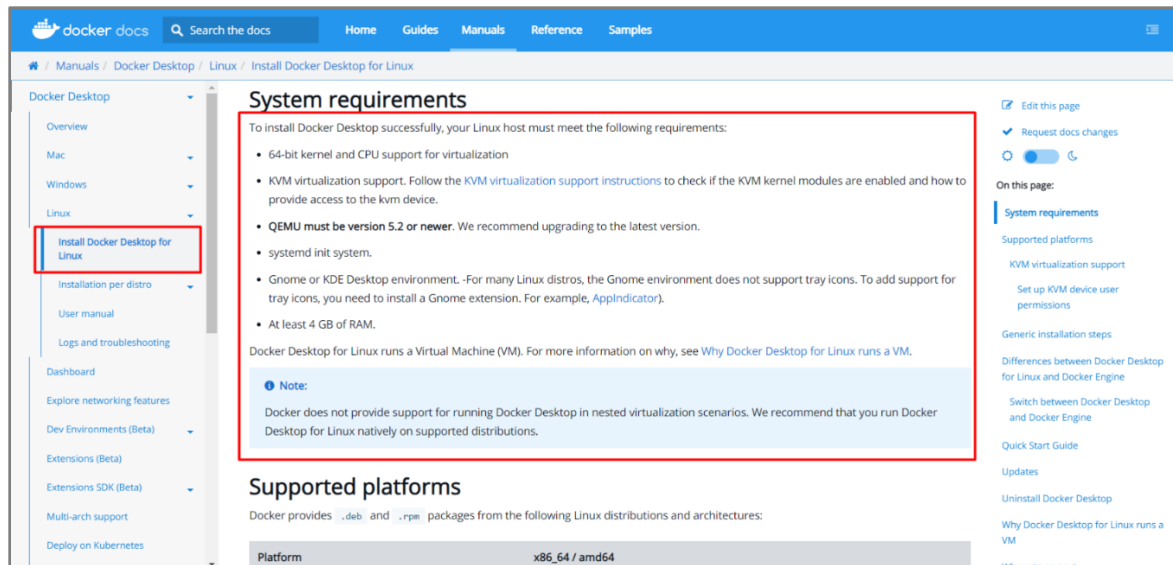
## Download Docker Desktop on Linux

First, you will need to open the Docker documentation <https://docs.docker.com/desktop/>.

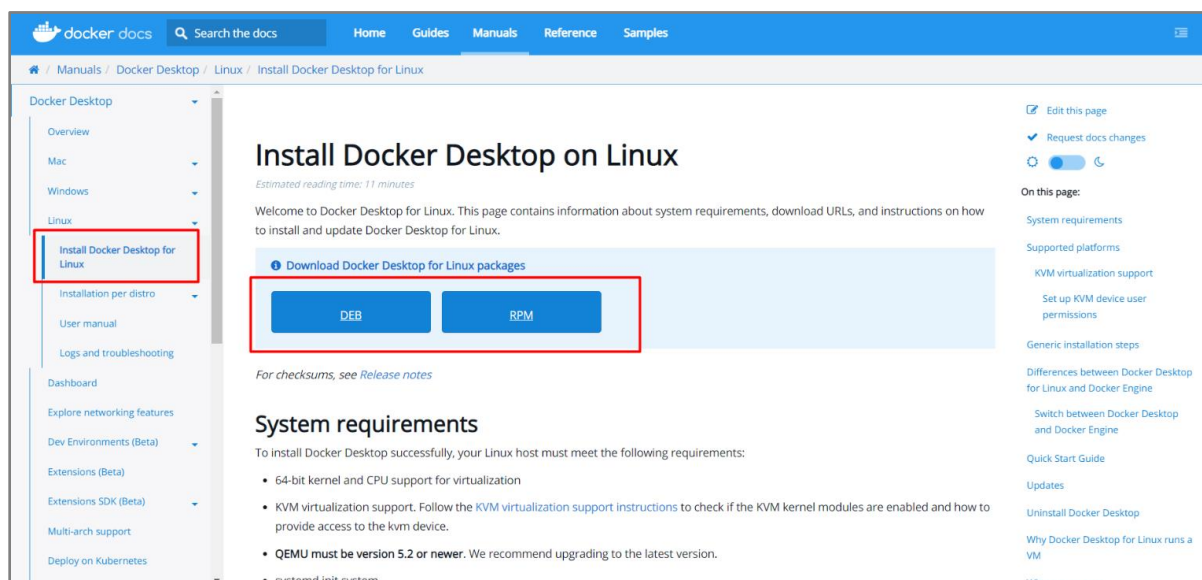
Choose "**Linux**" from the menu on the left side of the screen. Before installing make sure that your machine **matches the needed requirements** to use Docker Desktop.

For example, your Linux distribution should have 64-bit kernel and CPU support for virtualization, KVM virtualization support, and QEMU must be version 5.2 or newer:





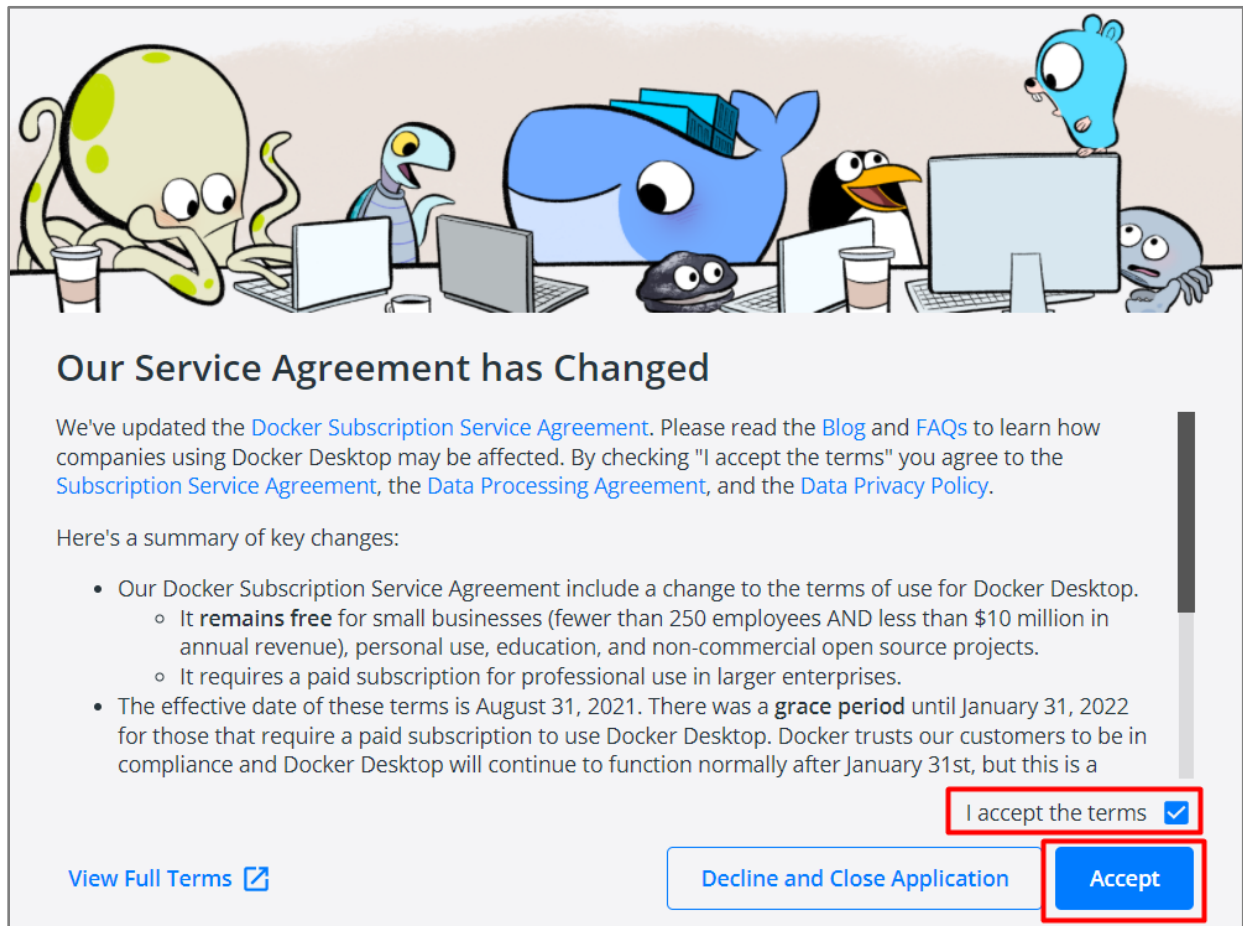
Download the correct package **for your Linux distribution** and **install** it with the **corresponding package manager**:



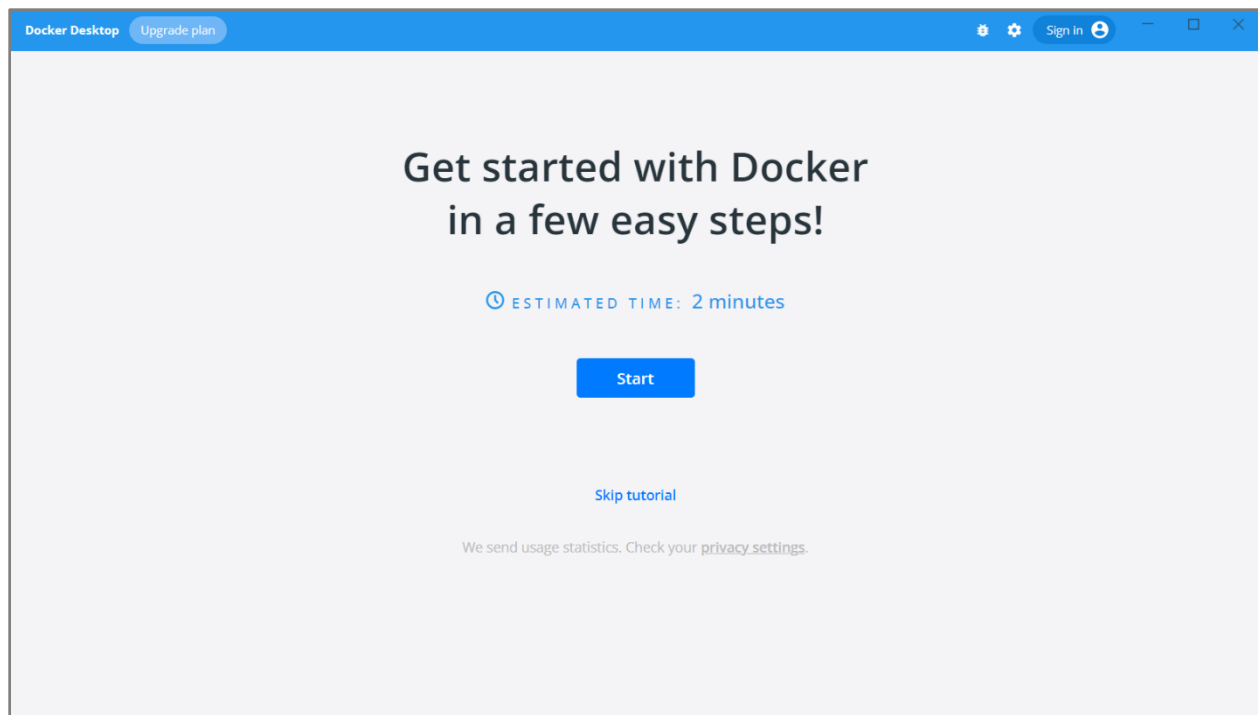
- More on how to install Docker Desktop on **Debian**:  
<https://docs.docker.com/desktop/linux/install/debian/>
- More on how to install Docker Desktop on **Fedora**:  
<https://docs.docker.com/desktop/linux/install/fedora/>
- More on how to install Docker Desktop on **Ubuntu**:  
<https://docs.docker.com/desktop/linux/install/ubuntu/>
- More on how to install Docker Desktop on **Arch**:  
<https://docs.docker.com/desktop/linux/install/archlinux/>

**Open your Applications menu in Gnome/KDE Desktop and search for Docker Desktop. Then, select Docker Desktop to start Docker.**

First, it will display the **Docker Subscription Service Agreement** window. Read the terms and click the checkbox to indicate that you accept the updated terms and then click **Accept** to continue:



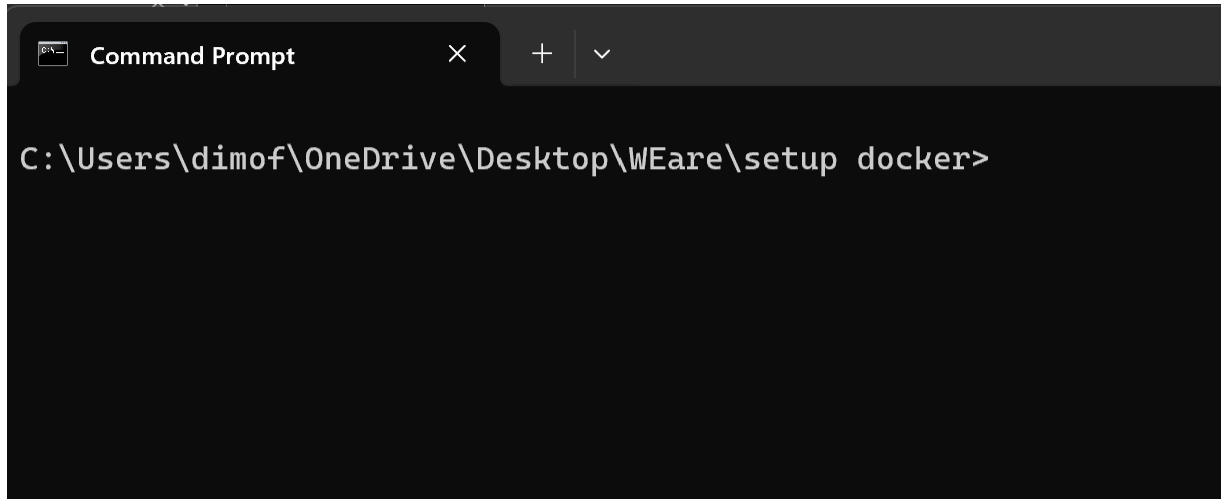
If you installed it **correctly**, you should see the following window:



Congratulations! You are now successfully running Docker Desktop on Linux.

# Run WEare app with Docker Compose

You will need to download the .ZIP file containing the APP files. Open the folder using your favorite Terminal [Command Prompt in the example below] and navigate to “setup docker” folder:

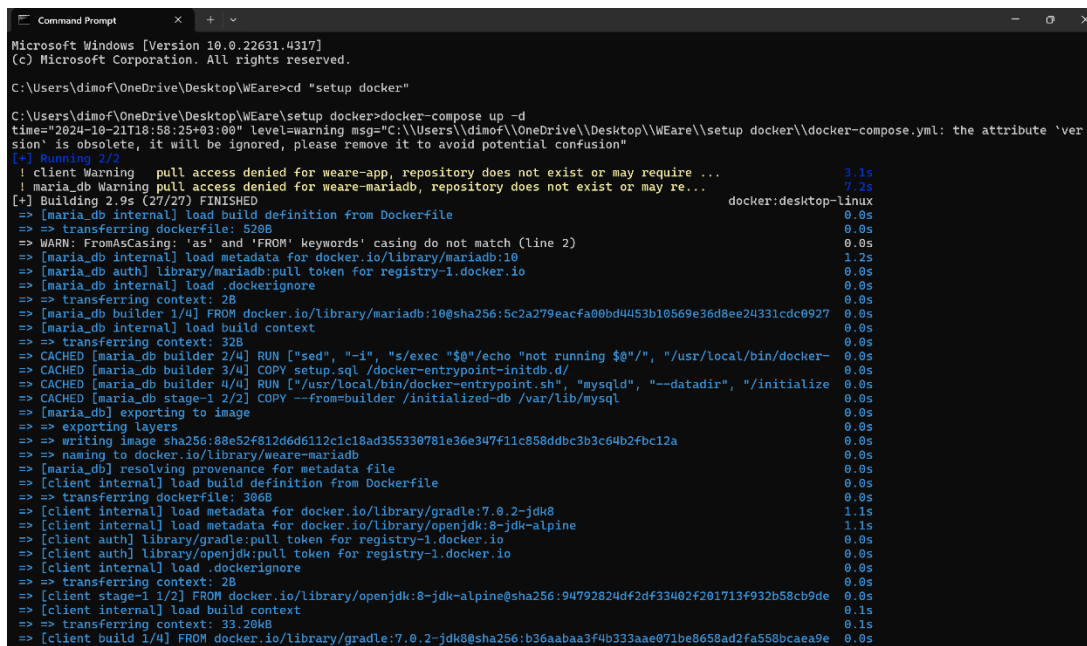


```
Command Prompt
C:\Users\dimof\OneDrive\Desktop\WEare\setup docker>
```

Once you are navigated to respective folder execute the following command in the terminal:

**docker-compose up -d**

Once done the docker commands will start to build and pull respective images. You should see something like this:



```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.











C:\Users\dimof\OneDrive\Desktop\WEare>cd "setup docker"

C:\Users\dimof\OneDrive\Desktop\WEare\setup docker>docker-compose up -d
time="2024-10-21T18:58:25+03:00" level=warning msg="C:\Users\dimof\OneDrive\Desktop\WEare\setup docker\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 2/2
! client Warning  pull access denied for weare-app, repository does not exist or may require ...    3.1s
! maria_db Warning pull access denied for weare-mariadb, repository does not exist or may require ...    7.2s
[+] Building 2.9s (27/27) FINISHED
=> [maria_db internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 520B                                                       0.0s
=> WARN: FromAsCasing: 'as' and 'FROM' keywords' casing do not match (line 2)             0.0s
=> [maria_db internal] load metadata for docker.io/library/mariadb:10                      1.2s
=> [maria_db auth] library/mariadb:pull token for registry-1.docker.io                    0.0s
=> [maria_db internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                             0.0s
=> [maria_db builder 1/4] FROM docker.io/library/mariadb:10@sha256:5c2a279eacfa08bd4453b10569e36d8ee24331cdc6927 0.0s
=> [maria_db internal] load build context                                                  0.0s
=> => transferring context: 32B                                                             0.0s
=> CACHED [maria_db builder 2/4] RUN ["sed", "-i", "s/exec \"$0\"/echo \"not running $0\"/", "/usr/local/bin/docker- 0.0s
=> CACHED [maria_db builder 3/4] COPY setup.sql /docker-entrypoint-initdb.d/              0.0s
=> CACHED [maria_db builder 4/4] RUN ["/usr/local/bin/docker-entrypoint.sh", "mysqld", "--datadir", "/initialize 0.0s
=> CACHED [maria_db stage-1 2/2] COPY --from=builder /initialized-db /var/lib/mysql         0.0s
=> [maria_db] exporting to image                                                          0.0s
=> => exporting layers                                                                    0.0s
=> => writing image sha256:88a52f812d6d6112c1c18ad355330781e36e347f11c858ddbc3b3c6b2fbc12a    0.0s
=> => naming to docker.io/library/weare-mariadb                                          0.0s
=> [maria_db] resolving provenance for metadata file                                     0.0s
=> [client internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 306B                                                       0.0s
=> [client internal] load metadata for docker.io/library/gradle:7.0.2-jdk8               1.1s
=> [client internal] load metadata for docker.io/library/openjdk:8-jdk-alpine            1.1s
=> [client auth] library/gradle:pull token for registry-1.docker.io                      0.0s
=> [client auth] library/openjdk:pull token for registry-1.docker.io                     0.0s
=> [client internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                             0.0s
=> [client stage-1 1/2] FROM docker.io/library/openjdk:8-jdk-alpine@sha256:94792824df2df33402f201713f932b58cb9de 0.0s
=> [client internal] load build context                                                  0.1s
=> => transferring context: 33.20kB                                                       0.1s
=> [client build 1/4] FROM docker.io/library/gradle:7.0.2-jdk8@sha256:b36aabaa3f4b333aae071be8658ad2fa558bcaea9e 0.0s
```

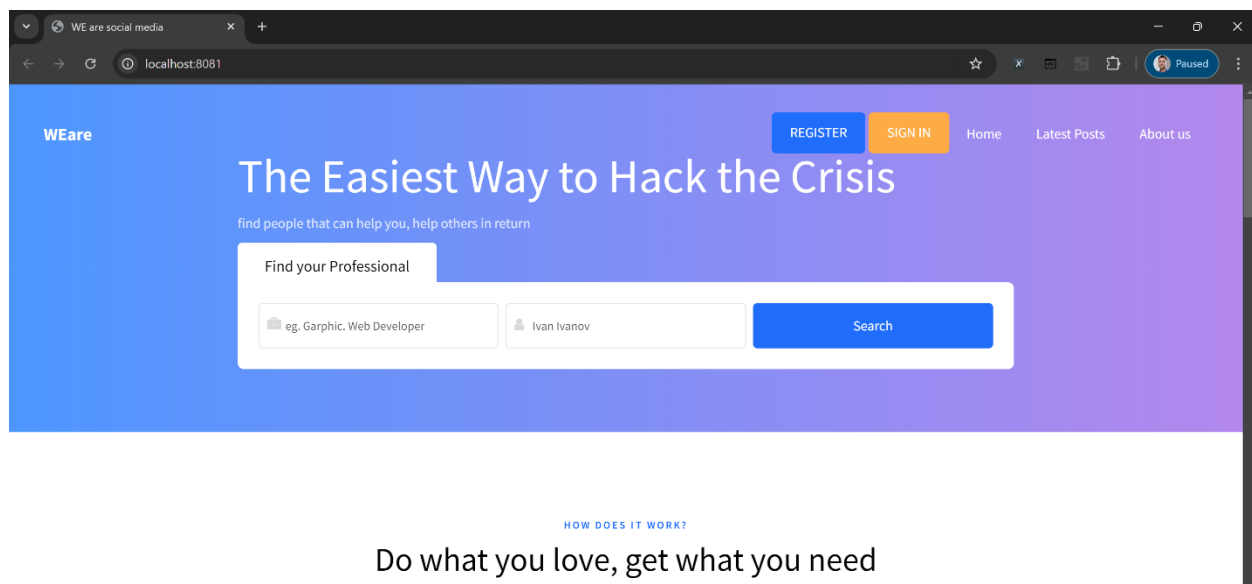
Once completed your terminal should display the following message:

```
=> => Writing image sha256:3c0b00c90b4d31c2011b01207c1b03007ea12a2a0c90d0340b00745100404 0.05s
=> => naming to docker.io/library/weare-app 0.05s
=> [client] resolving provenance for metadata file 0.05s
[+] Running 3/3
  ✓ Network project-weare Created 0.1s
  ✓ Container weare-mariadb Started 1.8s
  ✓ Container weare-app Started 1.4s
C:\Users\dimof\OneDrive\Desktop\WEare\setup docker>
```

If you open your Docker Desktop and navigate to Containers tab the following containers should be listed:

Container CPU usage ⓘ		Container memory usage ⓘ		Show charts			
0.00% / 1600% (16 CPUs available)		0B / 6.54GB					
<input type="text" value="Search"/>		<input type="checkbox"/> Only show running containers					
<input type="checkbox"/>	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	 setupdocker		Running (2/2)	0%		3 minutes ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 weare-mariadb afaf1a62a63e 	weare-mariadb:<none>	Running	0%	3307:3306 	3 minutes ago	<input type="checkbox"/> ⋮ 
<input type="checkbox"/>	 weare-app 0c266d6d7840 	weare-app:<none>	Running	0%	8081:8081 	3 minutes ago	<input type="checkbox"/> ⋮ 

You should be able to navigate to <http://localhost:8081> and see WEare app:



In case any of the ports used [8081 for the application and 3306 for the database] are not free, please feel free to change them in the **docker-compose.yml** file with any free ports. Here is an example on how we are changing the app to work on port **8085** and the database on port **3305**.

```
version: "3.8"
services:
  client:
    build: ../app
    image: weare-app
    container_name: weare-app
    ports:
      - "8085:8081"
    depends_on:
      - maria_db
    networks:
      - app-network

  maria_db:
    build: ../database
    image: weare-mariadb
    container_name: weare-mariadb
    ports:
      - 3305:3306
    networks:
      - app-network

networks:
  app-network:
    name: project-weare
```

Once changed execute again the same command in your terminal from “**setup docker**” folder:

**docker-compose up -d**

The app should start but this time on port 8085:

