

Docker

Java Foundation

Presented by

Spadavecchia Marisa

IBM Client Innovation Center - Italy

17/11/2023

IBM Client Innovation Center
Italy

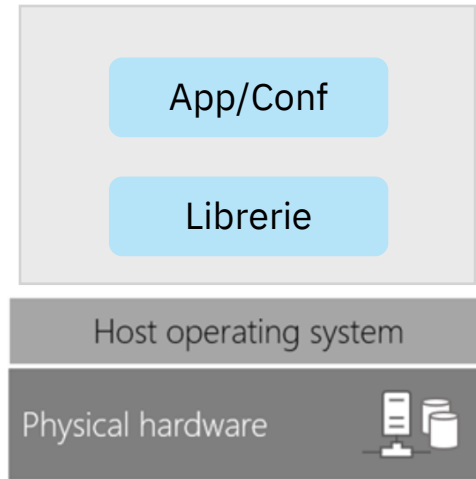
Agenda

- 1 Applicazioni tradizionali
- 2 Macchina Virtuale
- 3 Container
- 4 Punti di forza dei container
- 5 Container vs Virtual Machine

Agenda

-  6 Cos'è Docker
-  7 Strumenti e termini di Docker
-  8 Comandi base
-  9 Installazione di Docker Desktop

Applicazioni tradizionali



Le **applicazioni** software **tradizionali** sono tipicamente costituite da un insieme di librerie e file di configurazione in esecuzione su un ambiente di runtime.

Sono distribuite su un sistema operativo (Windows, Linux, Mac) che consente alle applicazioni di accedere ad un insieme di servizi in esecuzione, come un database o un server HTTP.

Punto debole: l'applicazione software è legata all'ambiente di runtime e qualsiasi aggiornamento o patch applicata al sistema operativo di base potrebbe interrompere l'applicazione.

Esempio1 : un aggiornamento del sistema operativo potrebbe includere aggiornamenti incompatibili con l'applicazione in esecuzione

Esempio 2: se due applicazioni condividono lo stesso insieme di librerie, può accadere che un aggiornamento che corregge le librerie della prima applicazione si ripercuote sulla seconda.

Ne consegue che qualsiasi **manutenzione dell'ambiente** potrebbe richiedere una serie completa di test per garantire che gli aggiornamenti del sistema operativo non influiscano anche sull'applicazione.

Gli aggiornamenti inoltre potrebbero determinare una **indisponibilità** dell'applicazione.

Virtual Machine

Una **Virtual Machine (Guest)** è un software che grazie ad un processo di virtualizzazione crea un ambiente virtuale che emula un ambiente fisico (PC, Server, Client) . E' una **istanza virtualizzata di un computer** ed è in grado di eseguire tutte le stesse funzioni di una macchina fisica, tra cui l'esecuzione delle applicazioni e dei sistemi operativi.

Le macchine virtuali vengono eseguite su una **macchina fisica** e accedono alle risorse (Ram, CPU, spazio disco) tramite un software chiamato **hypervisor**.

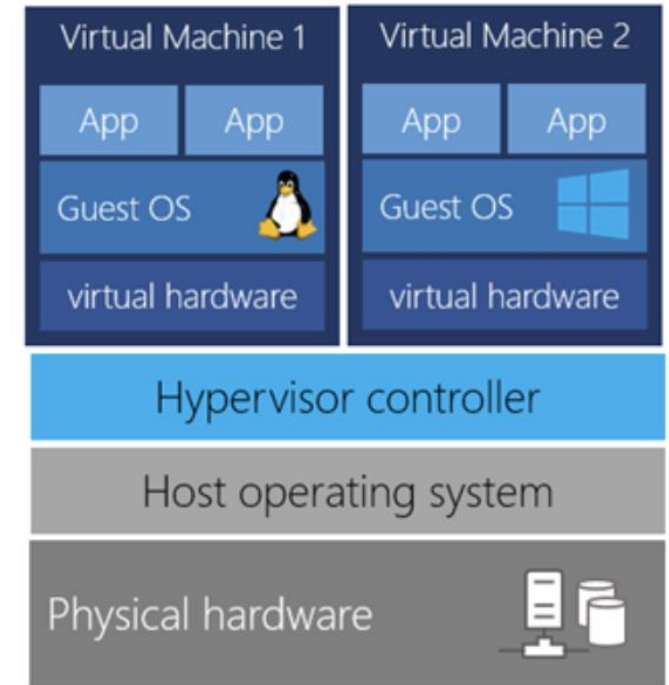
È possibile ospitare più VM su una singola macchina fisica.
In questo modo, le risorse di calcolo (computing, archiviazione, rete) sono flessibili e possono essere distribuite tra le VM in base alle esigenze, aumentando l'efficienza complessiva.

Punti di forza

- una macchina virtuale è **isolata** dal resto del sistema perchè il software eseguito al suo interno non può danneggiare il computer host, nè quello delle altre machine virtuali
- gli aggiornamenti su una VM non influenzano quelli di un'altra VM
- possono essere usate come ambienti di test e per svolgere operazioni rischiose

Punti deboli:

- ogni macchina virtuale necessita del proprio sistema operativo (servono licenze) e una copia virtuale di tutto l'hardware
- come una macchina fisica, richiede molte risorse quali cpu, ram e spazio disco
- tempi di riavvio non trascurabili



Container



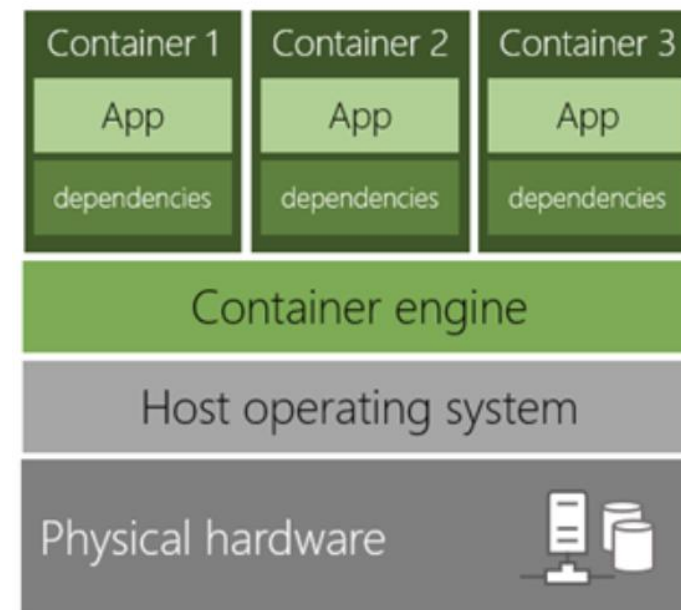
I **container** sono delle unità eseguibili di software in cui viene impacchettato il codice applicativo, insieme alle sue librerie e dipendenze, in modo da poter essere eseguito ovunque.

I container non hanno un proprio Sistema Operativo: si trovano su un server fisico e **condividono il kernel dell'OS** dell'host in sola lettura e solo per le informazioni che servono.

Il Kernel è l'interfaccia primaria tra l'hardware di un sistema e i suoi processi, ai quali consente di comunicare gestendo le risorse nel modo più efficiente possibile.

Le funzioni del SO vengono utilizzate sia per **isolare i processi** che per controllare la quantità di **CPU, memoria e disco** cui tali processi hanno accesso.

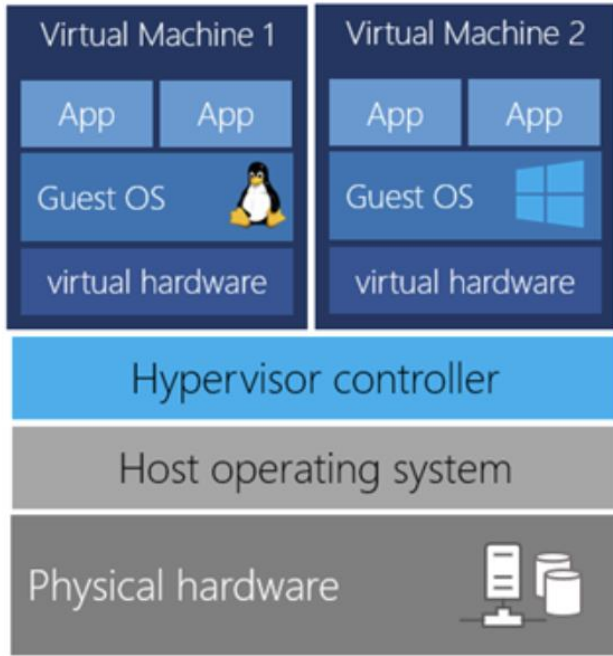
I container non **virtualizzano** l'hardware sottostante, ma solo il **sistema operativo** (SO), il che permette di far funzionare un'applicazione in modo corretto anche quando viene spostata tra VM diverse.



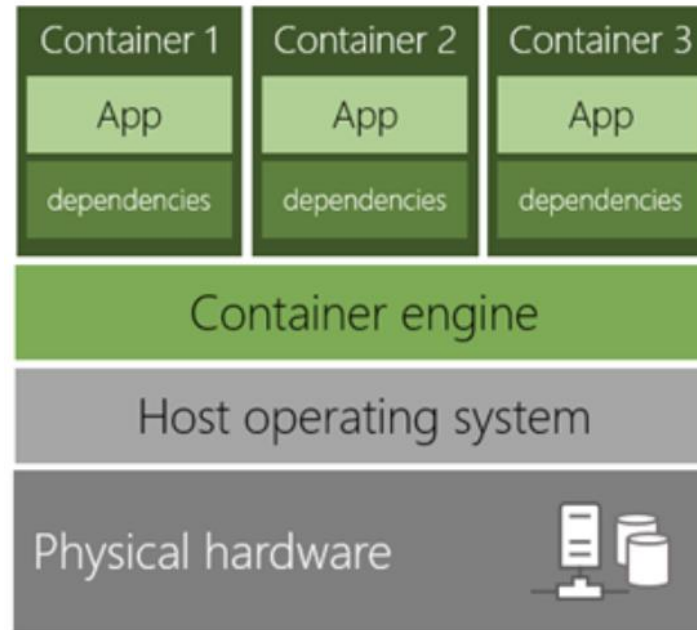
Punti di forza dei container

- **Leggeri** : i container condividono il kernel del sistema operativo della macchina, eliminando la necessità di un'istanza del sistema operativo completa per ogni applicazione e rendendo i file container piccoli e poco gravosi sulle risorse
- **Veloci**: la loro dimensione è più piccola, soprattutto rispetto alle VM, e possono essere avviati rapidamente
- **Isolati**: le modifiche apportate al sistema operativo host o ad altre applicazioni non influiscono sul container. Poiché le librerie necessarie a un contenitore sono autonome, l'applicazione può essere eseguita senza interruzioni
- **Portabili e indipendenti dalla piattaforma**: i container portano tutte le loro dipendenze con loro, il che significa che il software può essere scritto una volta e quindi eseguito senza che occorra riconfigurarli sui diversi ambienti
- **Supportano lo sviluppo e le architetture moderne**: i container sono piccoli e leggeri, il che li rende particolarmente adatti per le architetture a [microservizi](#) in cui le applicazioni sono costituite da molti servizi più piccoli, accoppiati in modo lasco e implementabili in modo indipendente.

Virtual Machine VS Container



Virtual Machine



Container

Cos'è Docker?

Docker è una piattaforma software **open-source** di containerizzazione.

Consente agli sviluppatori di:

- impacchettare le applicazioni in container
- eseguire, aggiornare e arrestare i container utilizzando dei semplici comandi con la massima rapidità

Strumenti e termini di Docker



- **Immagine Docker:** Le *immagini Docker* contengono il codice sorgente dell'applicazione eseguibile nonché tutti gli strumenti, le librerie e le dipendenze di cui il codice dell'applicazione ha bisogno per l'esecuzione come container. Sono una **fotografia immutabile** del file system di un S.O. tipicamente Linux
- **Dockerfile:** è un file di testo contenente le istruzioni su come creare l'immagine; tali istruzioni saranno eseguite da Docker per creare le immagini
- **Container Docker:** I container Docker sono le istanze in esecuzione delle immagini Docker. Mentre le immagini Docker sono file di sola lettura, il container è un **file system temporaneo** derivato dall'immagine di Docker sul quale è possibile effettuare delle modifiche. Gli utenti possono interagirvi e gli amministratori possono modificare le impostazioni utilizzando i comandi docker.

Strumenti e termini di Docker



- **Docker registry:** Un registro Docker è un sistema di storage e distribuzione open-source delle immagini Docker organizzato in [repository](#). Ogni repository consente di tracciare tutte le [versioni delle immagini](#), utilizzando l'assegnazione di tag per l'identificazione. Tale operazione viene eseguita utilizzando [git](#), uno strumento di controllo delle versioni.
- Tramite i comandi [pull](#) e [push](#) si possono scaricare le immagini in locale oppure caricarle sul registry;
- **Docker Hub:** è l'istanza pubblica del Docker registry, è "la più grande libreria e community al mondo di immagini container". Contiene più di 100.000 immagini.

Le immagini nel registro includono il nome dell'organizzazione o dell'utente, ad es.

[docker pull my-registry/foo/bar:2.1](#)

Effettuerà la pull dell'immagine foo/bar avente il tag 2.1 dal registro my-registry

- **Daemon Docker:** è un servizio che viene eseguito sul sistema operativo, come Microsoft Windows, Apple MacOS o iOS. Questo servizio crea e gestisce le immagini Docker

Comandi base

- Login to DockerHub: [docker login](#)
- Cercare una image nel repository pubblico di Docker: [docker search](#)
 - es: `docker search mysql`
- Scaricare in locale una image: [docker pull](#)
 - es: `docker pull mysql`
- Copiare una image locale sul registro: [docker push](#)
 - es. `docker push my-username/my-image`

Comandi base

Creare un container ed eseguirlo: `docker run <opzioni> <immagine> <argomenti>`

Rinominare un container: `docker rename <container> <nuovoNome>`

Avviare il container: `docker start <container>`

Arrestare il container: `docker stop <container>`

Forzare l'arresto del container: `docker kill <container>`

Eliminare il container: `docker rm <container>`

Visualizzare i container in esecuzione: `docker ps`

Visualizzare i container anche in non esecuzione: `docker ps -a`

Rimuovere i container non utilizzati: `docker container prune`

Visualizzare i processi in esecuzione di un container: `docker top <container>`

Visualizzare le stats di utilizzo: `docker stats <container>`

Comandi base

Visualizzare le differenze dei file system tra il container e la sua immagine: `docker diff <container>`

Visualizzare la configurazione sottoforma di JSON: `docker inspect <container>`

Creare un'immagine del file system del container: `docker commit <container> <name>`

Visualizzare la lista delle immagini: `docker image list` oppure `docker images`

Rimuovere un'immagine: `docker image rm <image>`

Rimuovere le immagini non utilizzate: `docker image prune`

Copiare un file dal container : `docker cp <container>:<pathContainer> <pathHost>`

Copiare un file dall'host al container: `docker cp <pathHost> <container>:<pathContainer>`

Esportare contenuto container come file .tar: `docker export <container> -o <nomeFile>`

Eseguire un comando all'interno di un container: `docker exec <container> <argomenti>`

Installazione Docker Desktop

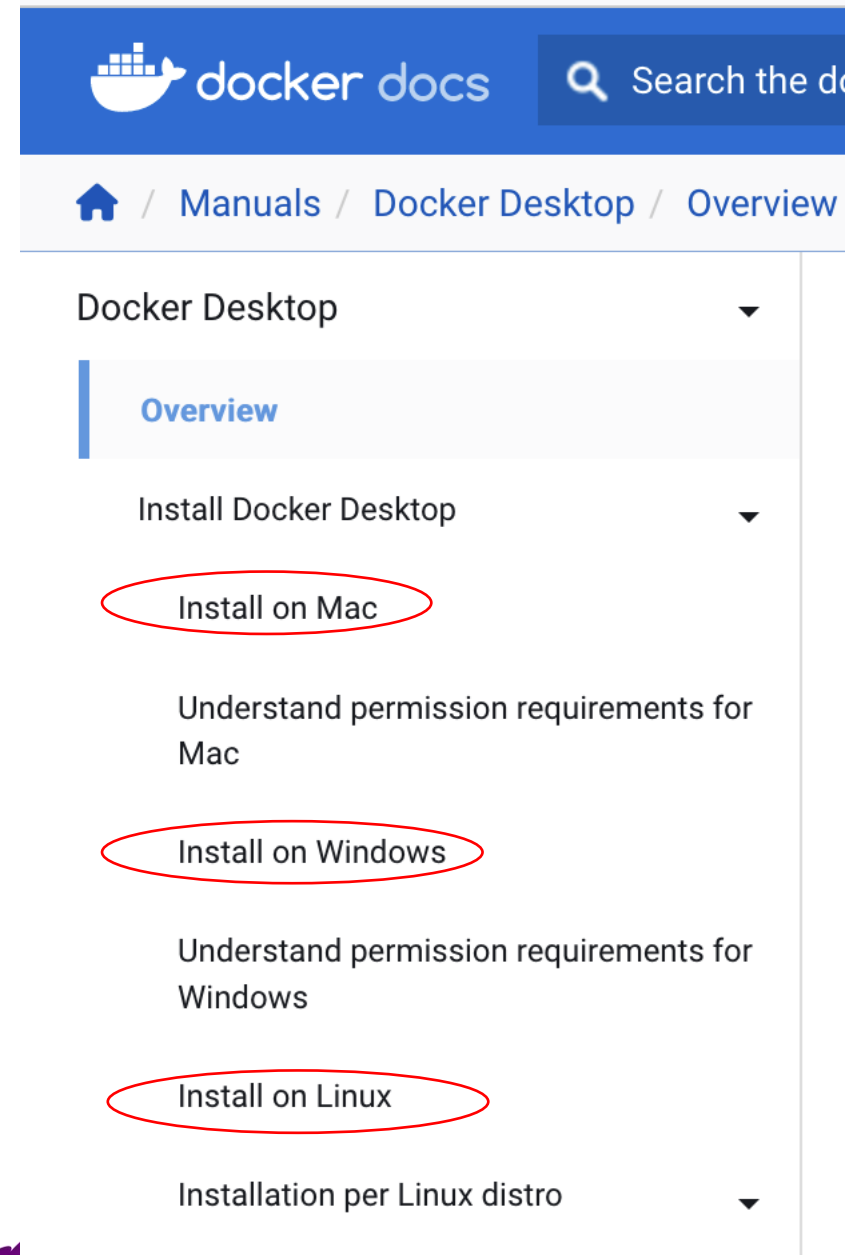
Docker Desktop è una applicazione, disponibile per Mac, Linux, o Windows che abilita alla costruzione e condivisione di applicazioni containerizzate.

Fornisce una **interfaccia grafica** che consente in maniera semplice la gestione dei container e delle immagini direttamente dal proprio computer.

Costante l'accesso ad una vasta quantità di immagini certificate su **DockerHub**

Tutte le istruzioni su come installare docker desktop sono reperibili sul sito ufficiale:

<https://docs.docker.com/desktop/>



Installazione Docker Desktop: Windows

System requirements

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

WSL 2 backend [Hyper-V backend and Windows containers](#)

Seguire le istruzioni di installazione per WSL 2

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](#).

WSL 2 and Windows Home

1. Virtual Machine Platform
2. [Windows Subsystem for Linux](#)
3. [Virtualization enabled in the BIOS](#)
4. Hypervisor enabled at Windows startup

Prima di installare Docker Desktop devono essere verificati i requisiti di sistema e verificata l'abilitazione della **virtualizzazione** nel BIOS così come descritto al link seguente

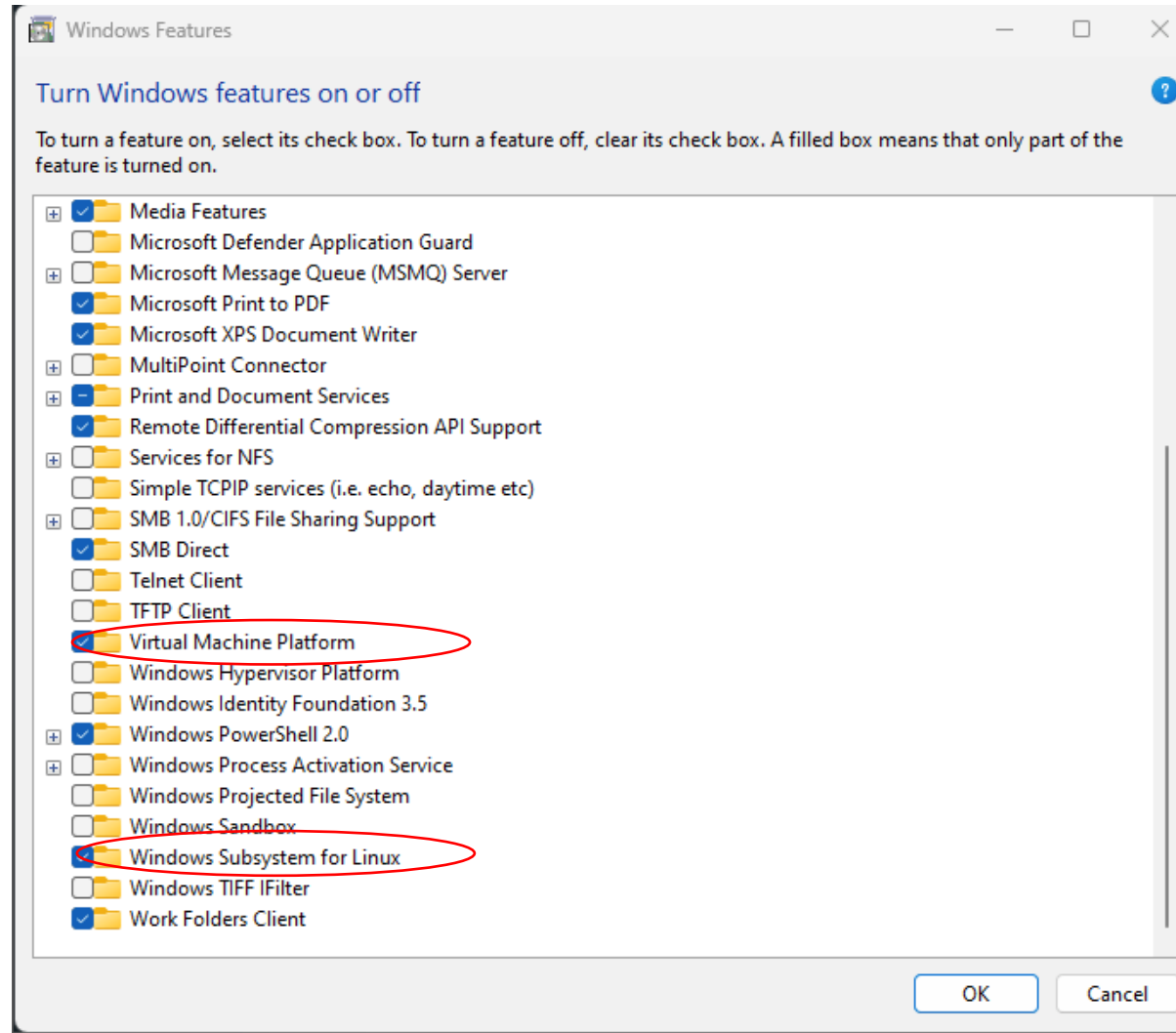
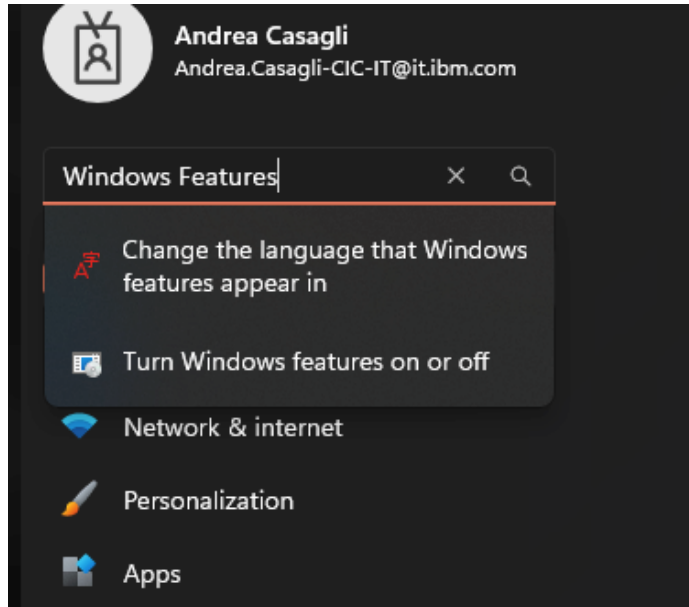
<https://docs.docker.com/desktop/troubleshoot/topics/#virtualization>

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

<https://bce.berkeley.edu/enabling-virtualization-in-your-pc-bios.html>

Installazione Docker Desktop: Windows

Verificare le impostazioni generali della virtualizzazione in **Windows Features**



Installazione Docker Desktop: Windows

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

Prima di installare Docker Desktop effettuare i seguenti passi:

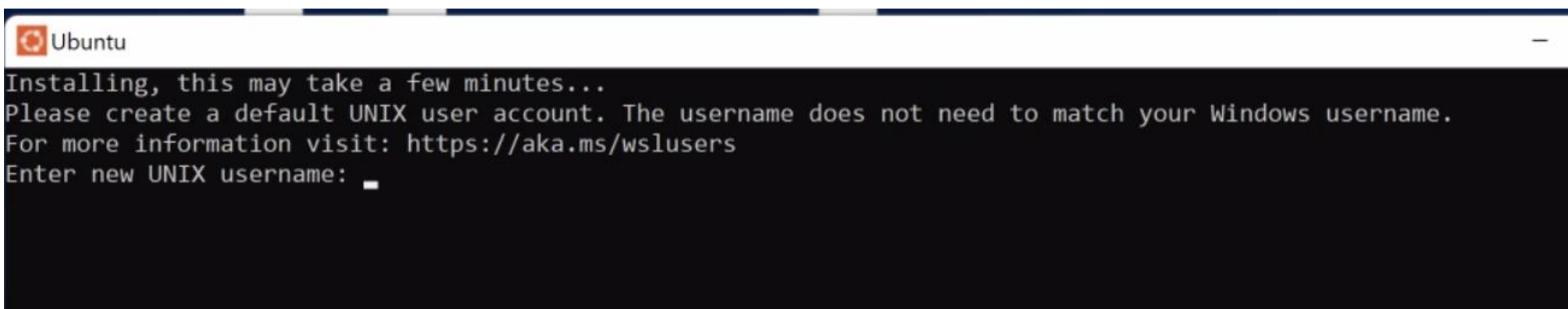
1. Verificare se è già presente una installazione di **WSL** (Windows Subsystem for Linux). Per farlo, avviare una istanza di Power Shell come amministratore e lanciare il comando `wsl -l -v`
2. Se WSL non è installato, lanciare il comando `wsl --install` oppure `wsl -install -d Ubuntu`; al termine dell'installazione riavviare il computer; questo comando installerà una distribuzione del Sistema operativo Linux (di default Ubuntu). Se è installato vedere la slide 23
3. Creare una **utenza da amministratore** per il Sistema Ubuntu: Subito dopo il riavvio, comparirà il terminale di Ubuntu in cui si dovrà inserire User e password

Per i dettagli su come creare sull'utenza andare sul sito <https://learn.microsoft.com/en-us/windows/wsl/setup/environment#set-up-your-linux-username-and-password>)

Schermate visibili durante l'installazione 1/2

```
PS C:\Windows\system32> wsl --install
Installazione in corso: Piattaforma macchina virtuale
Piattaforma macchina virtuale è stato installato.
Installazione in corso: Sottosistema Windows per Linux
Sottosistema Windows per Linux è stato installato.
Installazione in corso: Ubuntu
Ubuntu è stato installato.
L'operazione richiesta è stata eseguita. Le modifiche avranno effetto al riavvio del sistema.
PS C:\Windows\system32>
```

Esito del comando `wsl --install`

A screenshot of a terminal window titled 'Ubuntu'. The text inside the terminal reads: 'Installing, this may take a few minutes...', 'Please create a default UNIX user account. The username does not need to match your Windows username.', 'For more information visit: https://aka.ms/wslusers', and 'Enter new UNIX username:'. There is a cursor at the end of the last line.

```
Ubuntu
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: _
```

Terminale del sistema operativo Ubuntu in cui viene richiesta la [creazione dell'utenza Linux](#)

Schermate visibili duante l'installazione 2/2

```
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: evestito
New password:
Retype new password:
passwd: password updated successfully
Installation successful!
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.79.1-microsoft-standard-WSL2 x86_64)

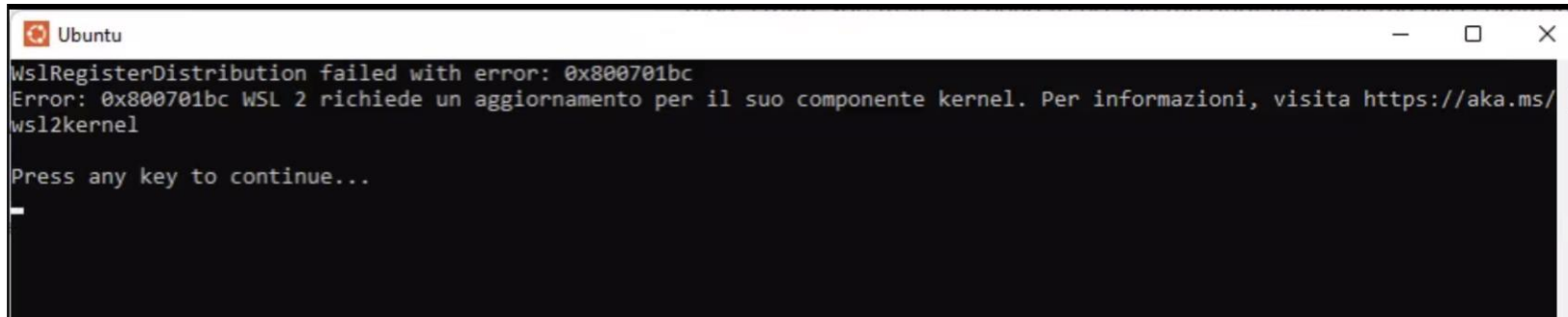
 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This message is shown once a day. To disable it please create the
/home/evestito/.hushlogin file.
evestito@IBM-PW03193V:~$
```

Terminale del sistema operativo Ubuntu dopo la [creazione dell'utenza](#)

Schermate visibili durante l'installazione

In presenza di questo errore



```
Ubuntu
WslRegisterDistribution failed with error: 0x800701bc
Error: 0x800701bc WSL 2 richiede un aggiornamento per il suo componente kernel. Per informazioni, visita https://aka.ms/wsl2kernel
Press any key to continue...
```

Effettuare l'aggiornamento scaricabile dal seguente link

<https://learn.microsoft.com/en-us/windows/wsl/install-manual#step-4---download-the-linux-kernel-update-package>

Verifica dell'installazione di WSL

4. Da powershell lanciare il comando **wsl -l -v** e accertarsi che la versione installata sia la 2

```
PS C:\Users\065643758> wsl -l -v
NAME      STATE      VERSION
* Ubuntu   Running    2
PS C:\Users\065643758>
```

Se già abilitata una versione di WSL diversa da 2, è possibile fare un upgrade di versione Installando l' aggiornamento

<https://learn.microsoft.com/en-us/windows/wsl/install-manual#step-4---download-the-linux-kernel-update-package>

E poi usando il comando per impostare la versione 2

wsl --set-version <distro name> 2 rimpiazzando <distro name> con il nome della versione di Linux

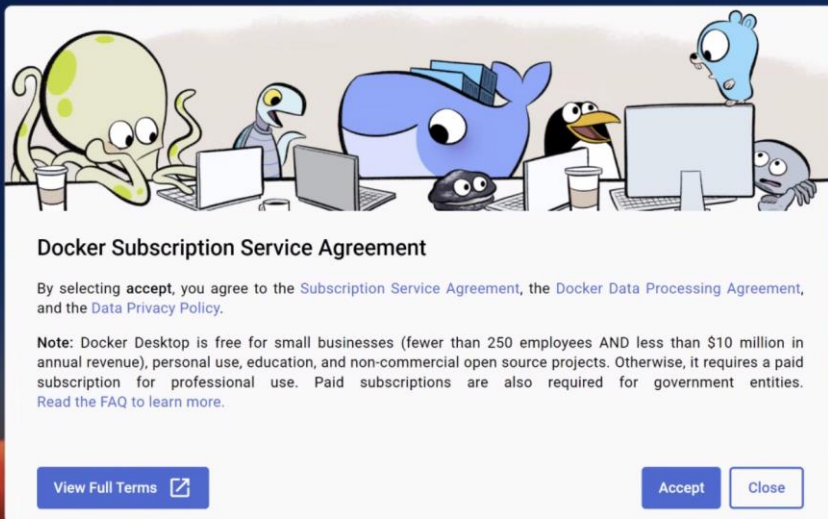
Installazione Docker Desktop: Windows

5. Scaricare docker desktop e avviare l'installazione.

Installing Docker Desktop 4.16.2 (95914)

Configuration

- ☒ Use WSL 2 instead of Hyper-V (recommended)
- ☒ Add shortcut to desktop



Installing Docker Desktop 4.16.2 (95914)

Docker Desktop 4.16.2

Installation succeeded

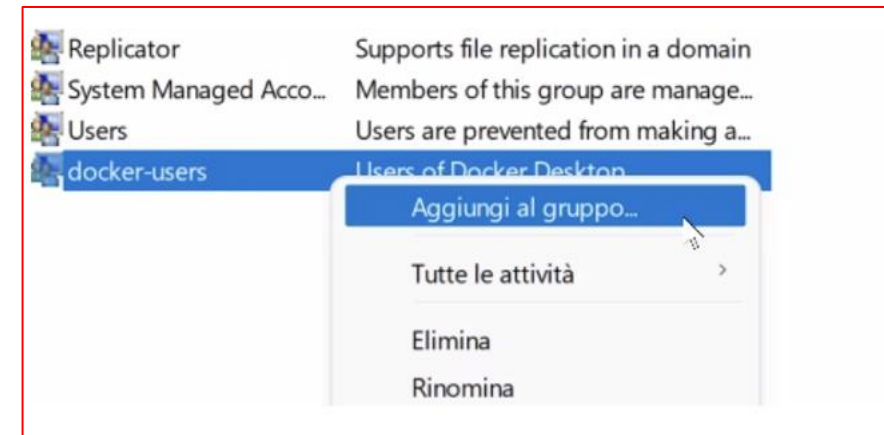
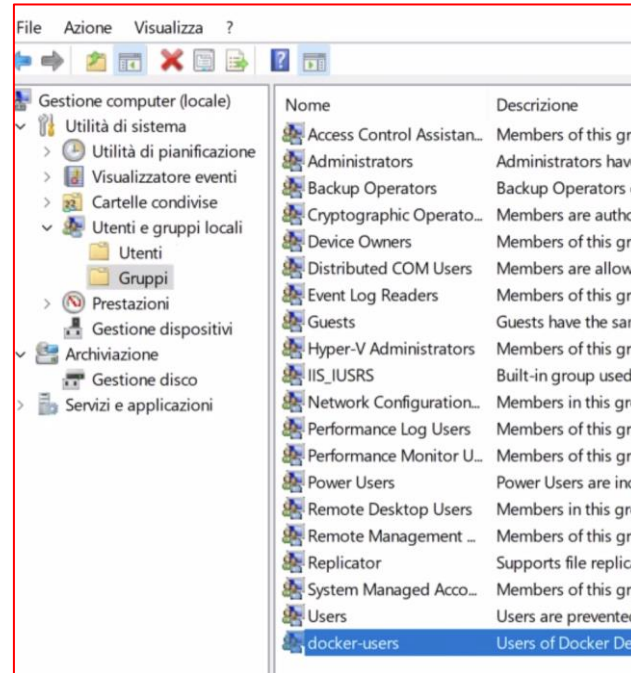
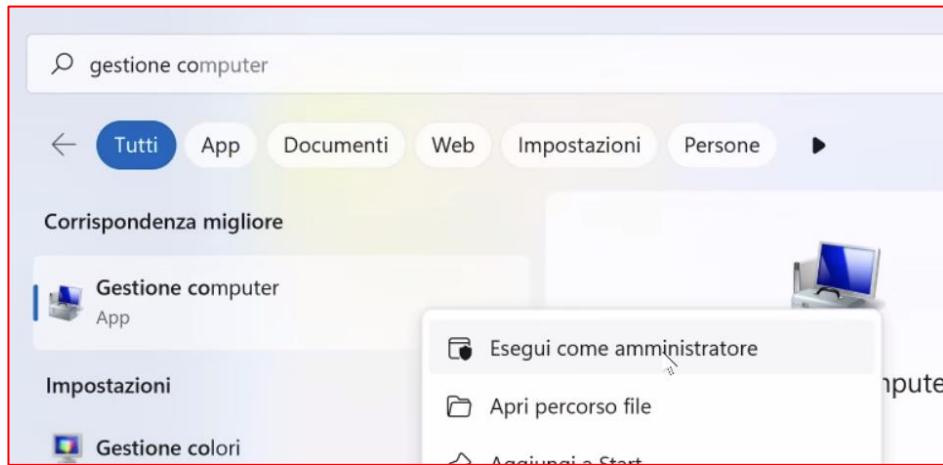
You must restart Windows to complete installation.

Close and restart


6. Dopo il riavvio, accettare il Service Agreement

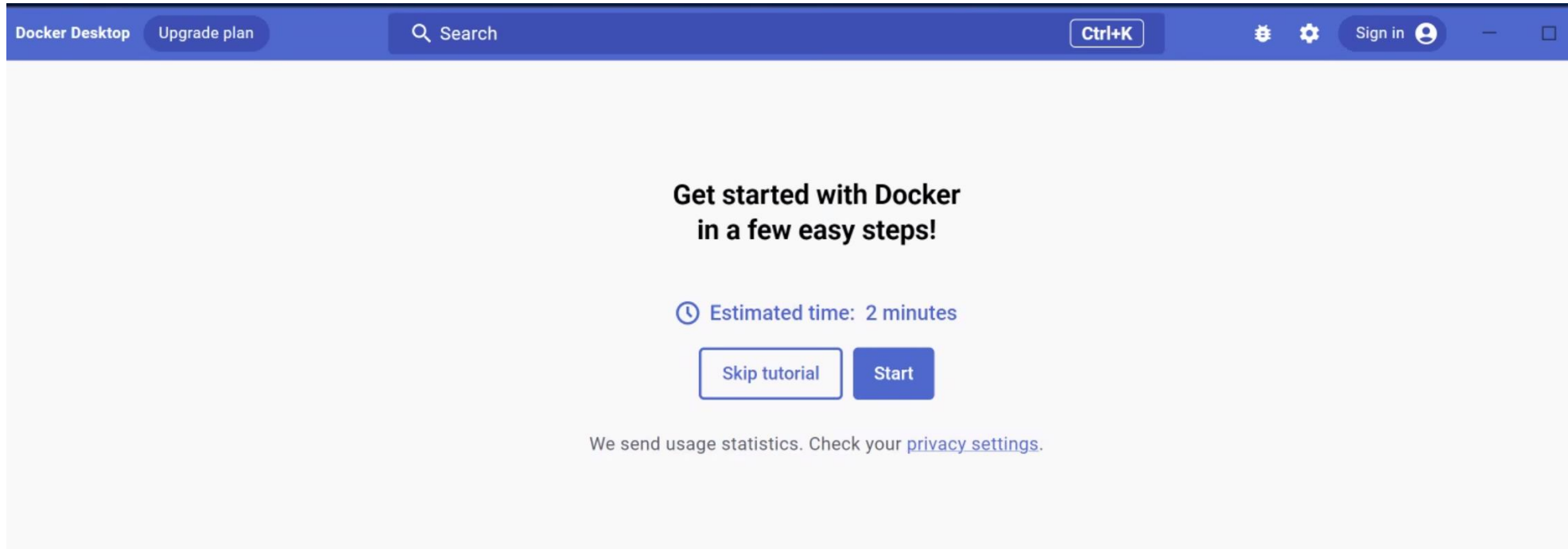
Installazione Docker Desktop: Windows

7. Aggiungere l'utente Windows al gruppo docker-user, solo se diverso dall'utente admin



Installazione Docker Desktop: Windows

Per avviare il Quick Start Guide, selezionare il menu Docker  e selezionare **Quick Start Guide**



Il Quick Start Guide include un semplice esercizio per per costruire una immagine Docker, eseguirla come container, effettuare il push e salvare l'immagine nel Docker Hub.

Installazione Docker Desktop: macOS - Intel

System requirements

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

Mac with Intel chip

Mac with Apple silicon

Mac with Intel chip

- macOS must be version 11 or newer. That is Big Sur (11), Monterey (12), or Ventura (13). We recommend upgrading to the latest version of macOS.

Note

Scaricare Docker.dmg ed avviare l'installazione: <https://docs.docker.com/desktop/mac/install/>

Installazione Docker Desktop: macOS –Apple silicon

Rosetta è un'app che permette a un Mac con processore Apple di usare app create per i Mac con processore Intel

System requirements

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

Mac with Intel chip

Mac with Apple silicon

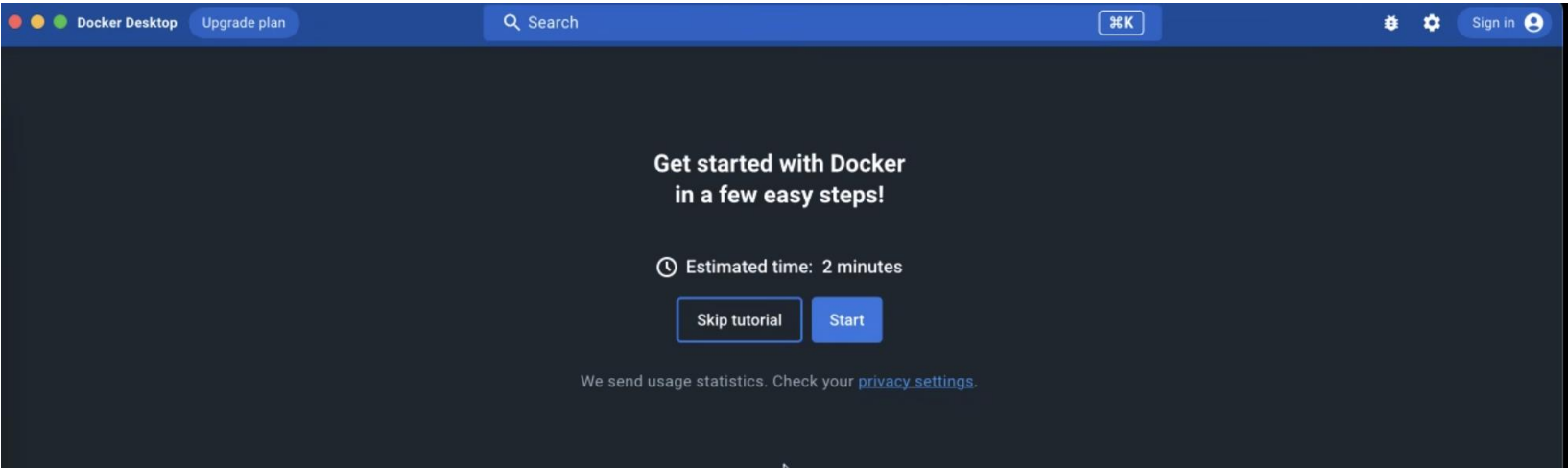
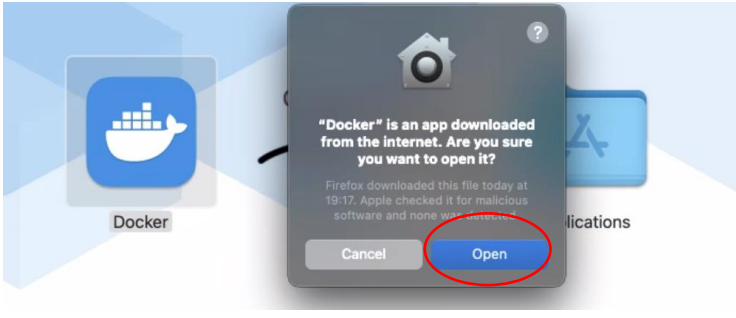
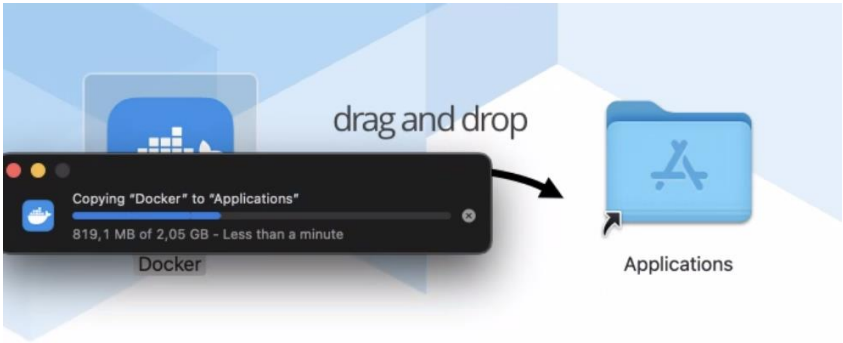
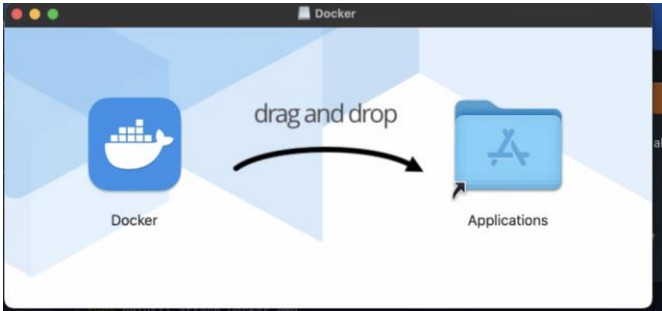
Mac with Apple silicon

- Beginning with Docker Desktop 4.3.0, we have removed the hard requirement to install **Rosetta 2**. There are a few optional command line tools that still require Rosetta 2 when using Darwin/AMD64. See the [Known issues section](#). However, to get the best experience, we recommend that you install Rosetta 2. To install Rosetta 2 manually from the command line, run the following command:

```
$ softwareupdate --install-rosetta
```

- ❖ eseguire il comando: **softwareupdate --install-rosetta**
- ❖ Scaricare Docker.dmg ed avviare l'installazione: <https://docs.docker.com/desktop/mac/install/>

Installazione Docker Desktop - macOS



Docker

Installazione Docker Desktop: Linux

Seguire la guida: <https://docs.docker.com/desktop/linux/install/ubuntu/>

Docker – Containerizzare la Todo List application

- Clonare il repository getting-started con il comando
git clone <https://github.com/docker/getting-started.git>
 - Verificare il contenuto nella directory **getting-started/app** (src, spec)
 - Creare il Dockerfile nella directory **app** copiando il seguente contenuto

```
# syntax=docker/dockerfile:1
FROM node:18-alpine
WORKDIR /app
COPY . .
RUN yarn install --production
CMD ["node", "src/index.js"]
EXPOSE 3000
```
 - Posizionarsi nella cartella app e costruire l'immagine lanciando il comando
`docker build -t getting-started .`
 - Avviare il container
`docker run -dp 3000:3000 getting-started`
- Testare il container puntando a <http://localhost:3000>

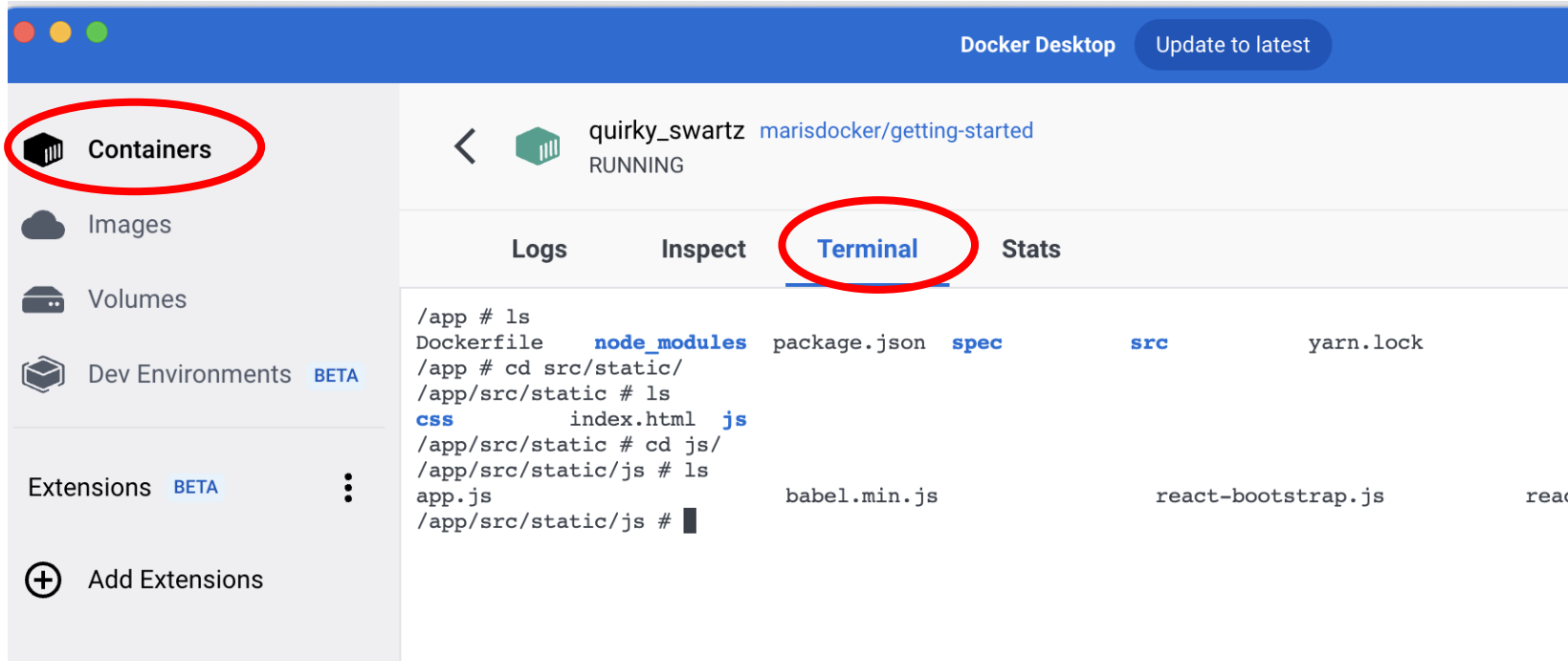
Docker – Modifichiamo l'immagine della Todo List application

- Aggiornare il file `src/static/js/app.js`
... - `<p className="text-center">No items yet! Add one above!</p>`
+ `<p className="text-center">You have no todo items yet! Add one above!</p>` ...
- Creare la versione aggiornata dell'immagine usando il comando
`docker build -t getting-started .`
- Creare un nuovo container usando l'immagine aggiornata (prima stoppare e rimuovere il precedente container)
`docker run -dp 3000:3000 getting-started`
- Testare nuovamente il container puntando a <http://localhost:3000>

Modificare il file js del container e verificare la modifica dall'url

Stoppare e rimuovere il container per verificare che la modifica effettuata nel container non è più disponibile

Docker – Modifichiamo il container



- dal terminale di Docker, entrare in editing nel file js dell'applicazione
- modificare nuovamente il messaggio in app.js
- verificare la modifica dall'url
- stoppare e rimuovere il container
- ricreare il container verificare che la modifica appena effettuata non è più visibile

Docker – Condividiamo la Todo List application su DockerHub

- Creare su Docker Hub un repository di nome `getting-started`
- Lloggarsi a docker con il comando `docker login -u YOUR-USER-NAME`
- Taggare l'immagine `docker tag getting-started YOUR-USER-NAME/getting-started`
- `docker images` per verificare
- Effettuare il push sul registry di DockerHub
`docker push YOUR-USER-NAME/getting-started`
- Ricreare il container
`docker run -dp 3000:3000 YOUR-USER-NAME/getting-started`