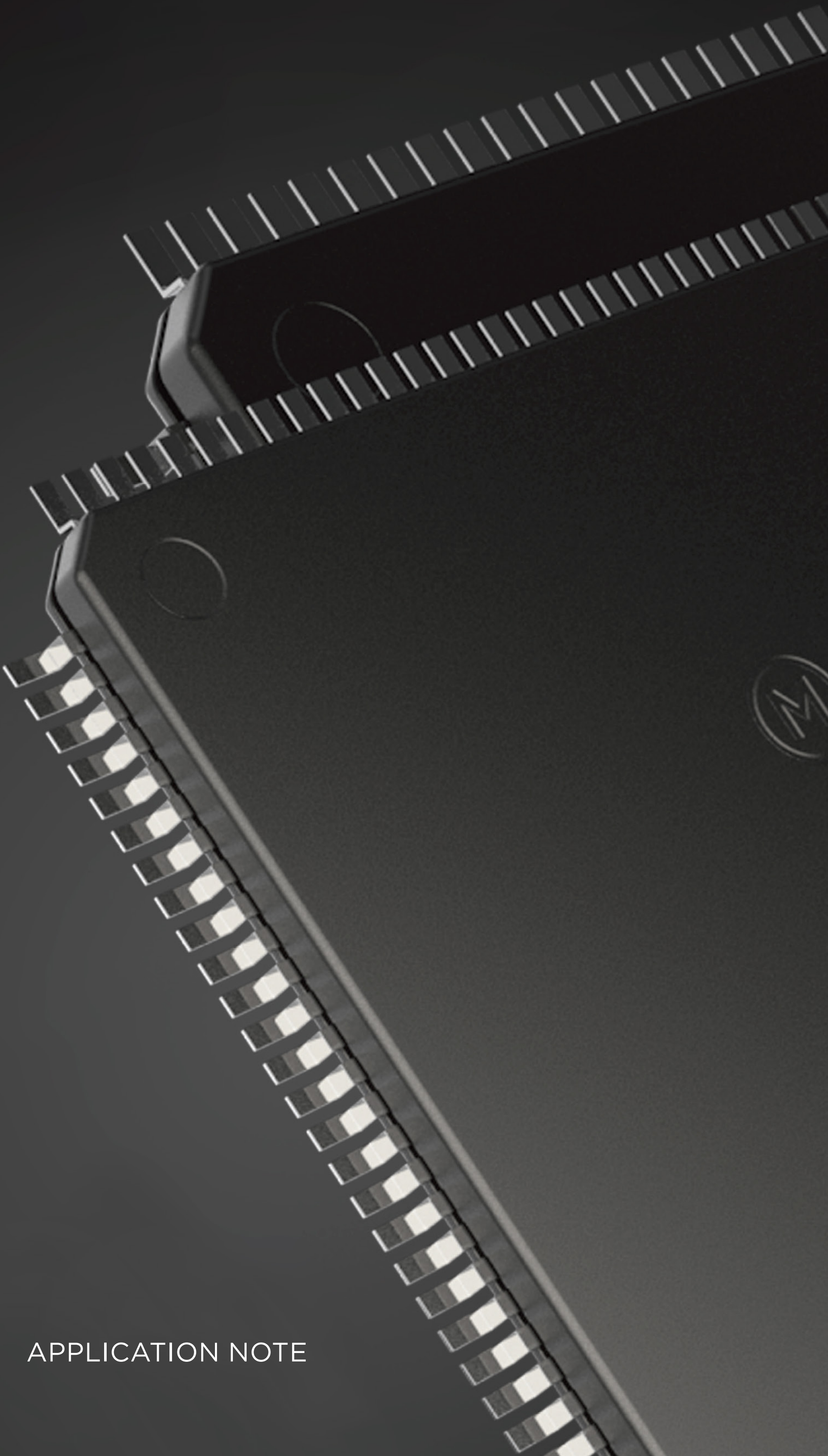
TASKING®

WINDOWS DOCKER IMAGE



APPLICATION NOTE

WINDOWS DOCKER IMAGE

INTRODUCTION

In this Application Note, we will explain how to Build and run **Windows Docker Image** for a TASKING product on Local Disk and Remote Repository. We will take the product "TASKING VX-toolset for TriCore v6.3r1" with a floating license, hosted via Remote TASKING License Server, as an example. Node-locked licenses are not recommended to be used for a Docker image.

WHAT IS A DOCKER IMAGE?

A Docker image is a read-only template that contains a set of instructions for creating a container that can run on a Docker platform. It provides a convenient way to package up applications and preconfigured server environments, which you can use for your own private use or share publicly with other Docker users. For more information about Docker, please refer to https://www.docker.com.

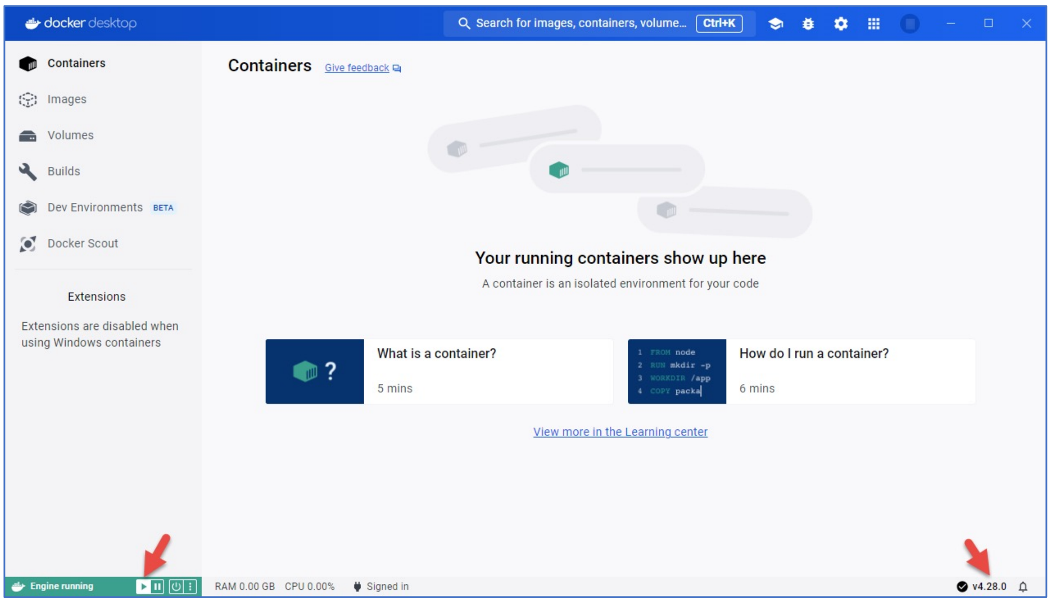
CREATE TASKING DOCKER IMAGE ON LOCAL DISK

Setting Up Docker Desktop application

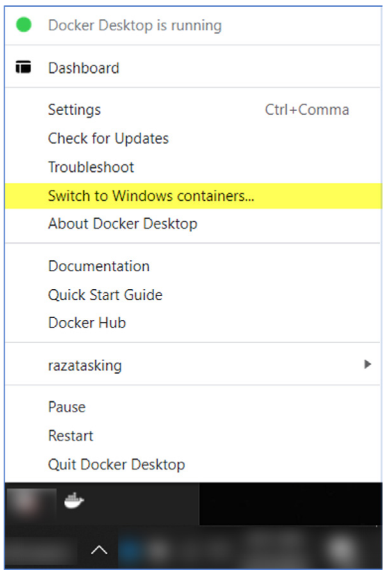
1. Download the latest Dockers Desktop App and install it from:
   https://www.docker.com/products/docker-desktop

For this Application Note Docker Desktop 4.28.0 will be used.

2. Launch the Docker Desktop app. Once Docker Engine had started successful it will look like this (Docker Icon turn to Green):


docker desktop
Search for images, containers, volume...
Ctrl+K
Containers
Images
Volumes
Builds
Dev Environments BETA
Docker Scout
Extensions
Extensions are disabled when using Windows containers
Containers
Give feedback
Your running containers show up here
A container is an isolated environment for your code
What is a container?
5 mins
How do I run a container?
6 mins
View more in the Learning center
Engine running
RAM 0.00 GB CPU 0.00%
Signed in
v4.28.0

3. Make sure to “Switch to Window Containers…” via right clicking on the Docker icon present in “Show Hidden Icons” from Windows task bar:



If you are creating an image for Linux OS leave it to Linux Container (this is the default).

Prepare Docker file

For this App Note, let’s assume that TASKING VX-toolset TriCore v6.3r1 is placed at the following installation directory:

```
C:\Program Files\TASKING\TriCore v6.3r1
```

1. Create a new text file(.txt) file at the installation directory of v6.3r1.
   For now, we will refer this .txt file as **“Dockerfile.txt”**.

2. Place the following text in Dockerfile:

```
FROM mcr.microsoft.com/windows/servercore:ltsc2019

RUN echo „Making a New Directory Called Tricore‟

RUN mkdir Tricore

RUN echo „Now Adding Folders‟

ADD . /Tricore

# Setting the Required Environment Variable

ENV TSK_OPTIONS_FILE_SW160800v6_3r1 "C:/Tricore/etc/licopt.txt"

# Sets a command that will run forever to keep container running

CMD ["powershell", "While(1) {}"]
```

3. Save and Dockerfile.txt and close it.

4. Rename the **Dockerfile.txt** to **Dockerfile**
   (Remove the extension .txt as the Docker engine does not accept the .txt extension)

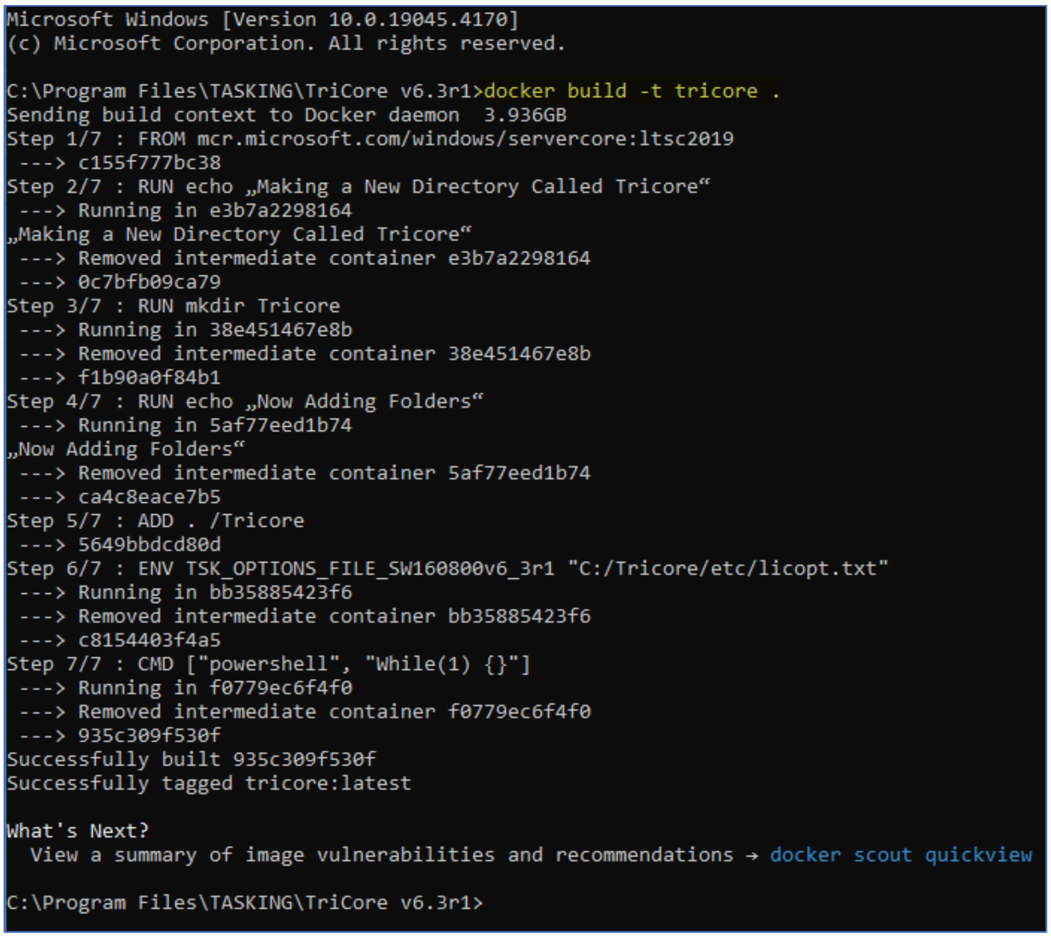
Please note that if you are using another version of Tricore VX toolset e.g. v6.2r2 or lower, please adopt the above highlighted environmental variable (TSK_OPTIONS_FILE_SW160800v6_3r1) accordingly.

Creating Docker Image Locally

1. Open the Command Prompt(cmd.exe) at Tricore Installation Directory and write the following command:

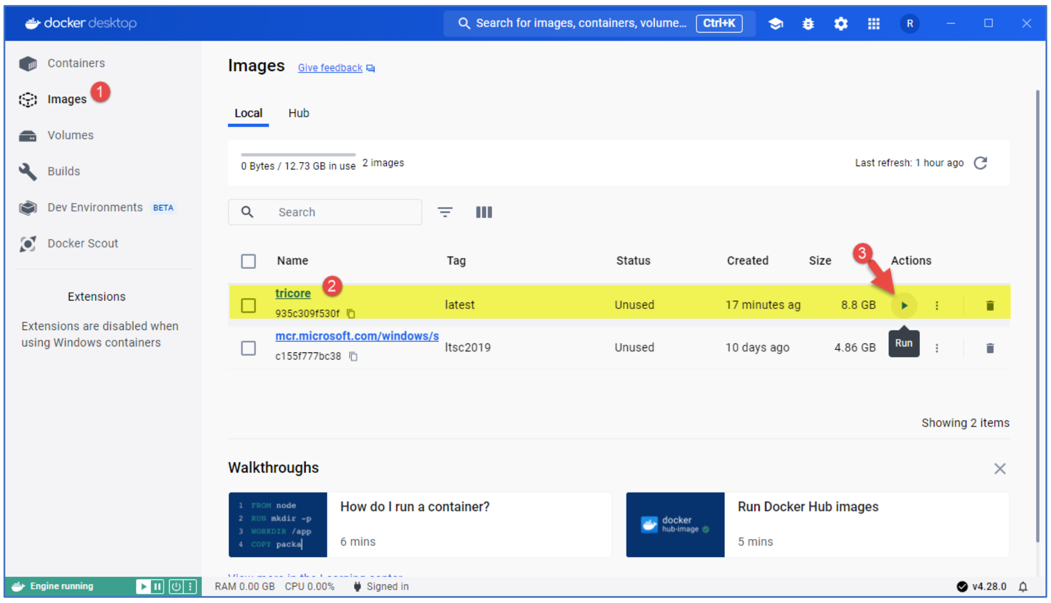
   ```
   docker build -t tricore .
   ```

   i.e.



```
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Program Files\TASKING\TriCore v6.3r1>docker build -t tricore .
Sending build context to Docker daemon  3.936GB
Step 1/7 : FROM mcr.microsoft.com/windows/servercore:ltsc2019
 ---> c155f777bc38
Step 2/7 : RUN echo „Making a New Directory Called Tricore“
 ---> Running in e3b7a2298164
„Making a New Directory Called Tricore“
 ---> Removed intermediate container e3b7a2298164
 ---> 0c7bfb09ca79
Step 3/7 : RUN mkdir Tricore
 ---> Running in 38e451467e8b
 ---> Removed intermediate container 38e451467e8b
 ---> f1b90a0f84b1
Step 4/7 : RUN echo „Now Adding Folders“
 ---> Running in 5af77eed1b74
„Now Adding Folders“
 ---> Removed intermediate container 5af77eed1b74
 ---> ca4c8eace7b5
Step 5/7 : ADD . /Tricore
 ---> 5649bbdcd80d
Step 6/7 : ENV TSK_OPTIONS_FILE_SW160800v6_3r1 "C:/Tricore/etc/licopt.txt"
 ---> Running in bb35885423f6
 ---> Removed intermediate container bb35885423f6
 ---> c8154403f4a5
Step 7/7 : CMD ["powershell", "While(1) {}"]
 ---> Running in f0779ec6f4f0
 ---> Removed intermediate container f0779ec6f4f0
 ---> 935c309f530f
Successfully built 935c309f530f
Successfully tagged tricore:latest

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Program Files\TASKING\TriCore v6.3r1>
```

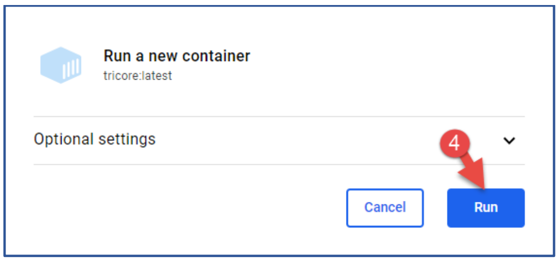
2. This will in return create an image named as “tricore” with the tag “latest” (default) like (as indiceted in the screenshot below):



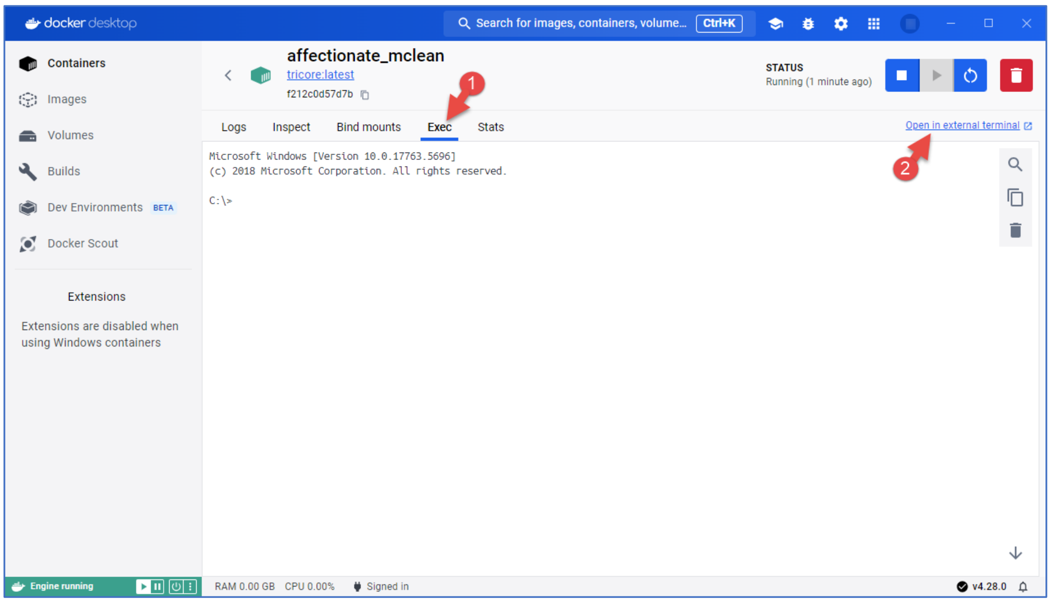
Name	Tag	Status	Created	Size	Actions
tricore 935c309f530f	latest	Unused	17 minutes ag	8.8 GB	
mcr.microsoft.com/windows/s c155f777bc38	ltsc2019	Unused	10 days ago	4.86 GB	

Running the Docker Container

1. Click on the **"Run"** button (as indicated above at 3) in-order to run the image. This will open a pop-up window as shown below, click on **"Run"** (as indicated at 4)

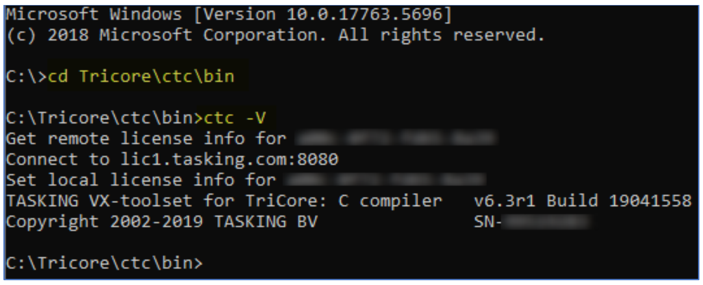


2. Once the Docker image is running, click on the Execute (as indicated below at 1) and open the command line interface via clicking at Open in external terminal (as indicated below at 2)



```
Microsoft Windows [Version 10.0.17763.5696]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>
```

3. Once the command line interface is open, change the working directory to `\Tricore\ctc\bin` and run the "`ctc -V`" command as indicated below:



```
Microsoft Windows [Version 10.0.17763.5696]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>cd Tricore\ctc\bin

C:\Tricore\ctc\bin>ctc -V
Get remote license info for [redacted]
Connect to lic1.tasking.com:8080
Set local license info for [redacted]
TASKING VX-toolset for TriCore: C compiler   v6.3r1 Build 19041558
Copyright 2002-2019 TASKING BV               SN-[redacted]

C:\Tricore\ctc\bin>
```

Remark: During the preparation of Dockerfile, we had introduced a layer to create a new folder named "Tricore" in the Docker container (`RUN mkdir Tricore`). After that we have copied (`ADD . /Tricore`) the full content of the root installation folder (`C:\Program Files\TASKING\TriCore v6.3r1`) to this newly created folder Tricore (`C:\Tricore`).

4. This will show the current version number of Tricore being used in the Docker image which proofs that the Tricore image has been created successfully.

CREATE A TASKING DOCKER IMAGE FOR DOCKER HUB REPOSITORIES

Building an Image for Docker Repositories

1. In order to push an image to docker repository on Docker Hub, first sign up for the Docker Hub community and login at Docker Desktop App with your respective credentials. Create a repository in Docker Hub. In this App Note we are going to refer username as **taskingdocker** and repository as **tricore_2024**. Once the repository has been created, go to the local installation directory of TASKING VX-toolset TriCore v6.3r1. For this App Note we consider that as:

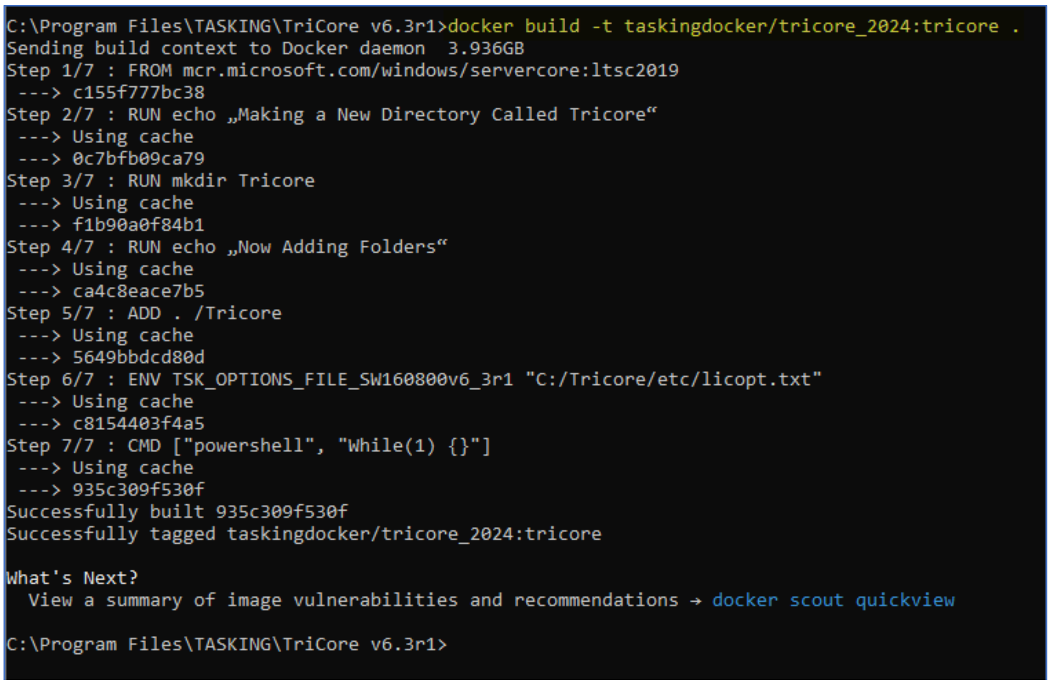
   ```
   C:\Program Files\TASKING\TriCore v6.3r1
   ```

2. Add the Docker file as mentioned in previous Chapter (please refer to Prepare Docker file).
   Open the command prompt(cmd.exe) here and use the following command to build an image for repository.

   ```
   docker build -t [username]/[repository_name]:image_tag .
   ```

   like:

   ```
   docker build -t taskingdocker/tricore_2024:tricore .
   ```



```
C:\Program Files\TASKING\TriCore v6.3r1>docker build -t taskingdocker/tricore_2024:tricore .
Sending build context to Docker daemon  3.936GB
Step 1/7 : FROM mcr.microsoft.com/windows/servercore:ltsc2019
 ---> c155f777bc38
Step 2/7 : RUN echo „Making a New Directory Called Tricore“
 ---> Using cache
 ---> 0c7bfb09ca79
Step 3/7 : RUN mkdir Tricore
 ---> Using cache
 ---> f1b90a0f84b1
Step 4/7 : RUN echo „Now Adding Folders“
 ---> Using cache
 ---> ca4c8eace7b5
Step 5/7 : ADD . /Tricore
 ---> Using cache
 ---> 5649bbdcd80d
Step 6/7 : ENV TSK_OPTIONS_FILE_SW160800v6_3r1 "C:/Tricore/etc/licopt.txt"
 ---> Using cache
 ---> c8154403f4a5
Step 7/7 : CMD ["powershell", "While(1) {}"]
 ---> Using cache
 ---> 935c309f530f
Successfully built 935c309f530f
Successfully tagged taskingdocker/tricore_2024:tricore

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

C:\Program Files\TASKING\TriCore v6.3r1>
```

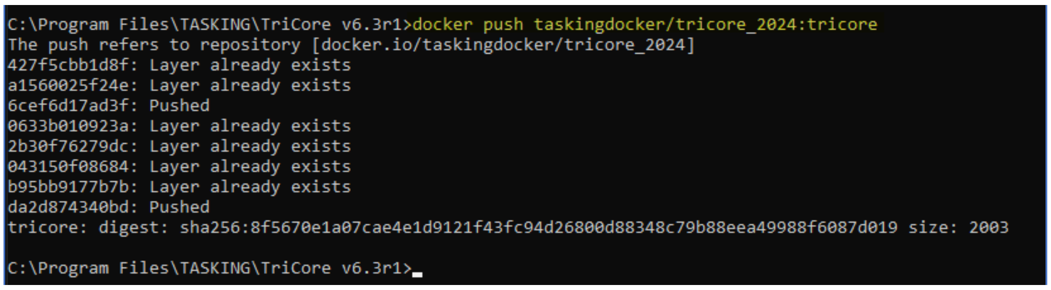
Pushing an Image to Docker Repositories

1. Once the image had been successfully built for the repository, it can be pushed to repository with the following command

   ```
   docker push [username]/[repository_name]:tagname
   ```

   like:

   ```
   docker push taskingdocker/tricore_2024:tricore
   ```



```
C:\Program Files\TASKING\TriCore v6.3r1>docker push taskingdocker/tricore_2024:tricore
The push refers to repository [docker.io/taskingdocker/tricore_2024]
427f5cbb1d8f: Layer already exists
a1560025f24e: Layer already exists
6cef6d17ad3f: Pushed
0633b010923a: Layer already exists
2b30f76279dc: Layer already exists
043150f08684: Layer already exists
b95bb9177b7b: Layer already exists
da2d874340bd: Pushed
tricore: digest: sha256:8f5670e1a07cae4e1d9121f43fc94d26800d88348c79b88eea49988f6087d019 size: 2003

C:\Program Files\TASKING\TriCore v6.3r1>
```

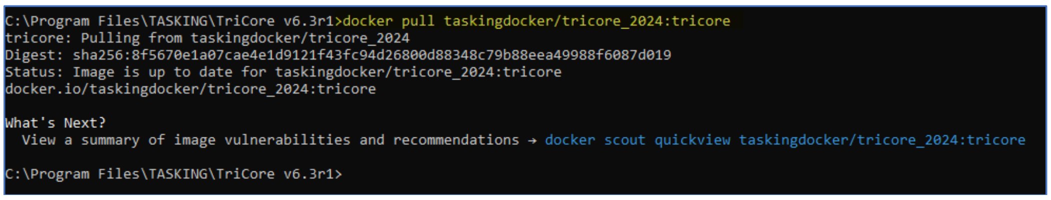
Pulling an Image from Docker Repositories

1. Once the image had been successfully pushed in the repository, it can be pulled from the repository with the following command

```
docker pull [username]/[repository_name]:tagname
```

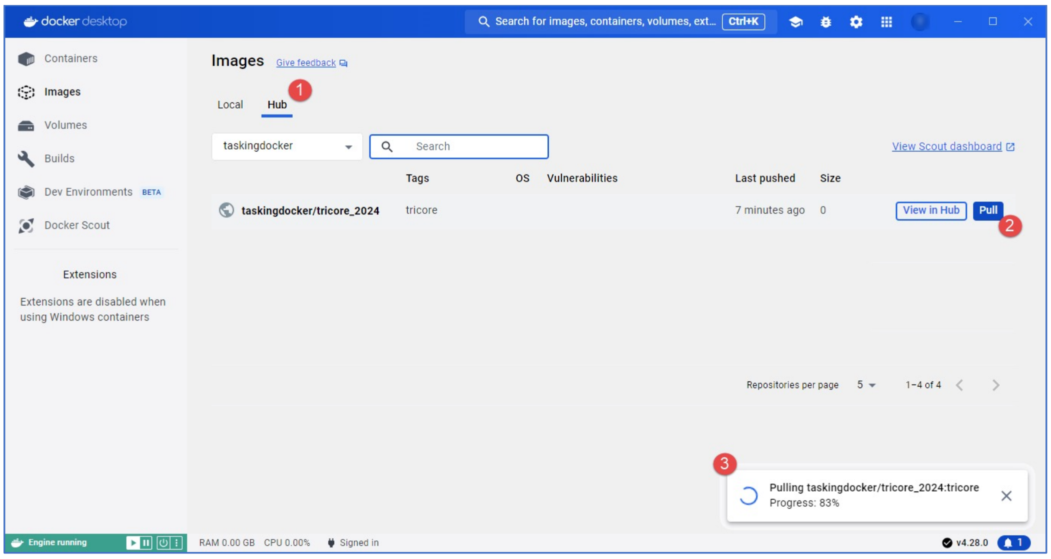
like:

```
docker pull taskingdocker/tricore_2024:tricore
```



```
C:\Program Files\TASKING\TriCore v6.3r1>docker pull taskingdocker/tricore_2024:tricore
tricore: Pulling from taskingdocker/tricore_2024
Digest: sha256:8f5670e1a07cae4e1d9121f43fc94d26800d88348c79b88eea49988f6087d019
Status: Image is up to date for taskingdocker/tricore_2024:tricore
docker.io/taskingdocker/tricore_2024:tricore

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview taskingdocker/tricore_2024:tricore

C:\Program Files\TASKING\TriCore v6.3r1>
```

You can also pull the image via using Docker Desktop App via following the steps below:



	Tags	OS	Vulnerabilities	Last pushed	Size
taskingdocker/tricore_2024	tricore			7 minutes ago	0

Pulling taskingdocker/tricore_2024:tricore
Progress: 83%

Note: This process also works for a local TLM server based floating license. Then the licopt.txt file will include an entry for the local TLM server and this entry is also generic. So it can be used with multiple docker containers.

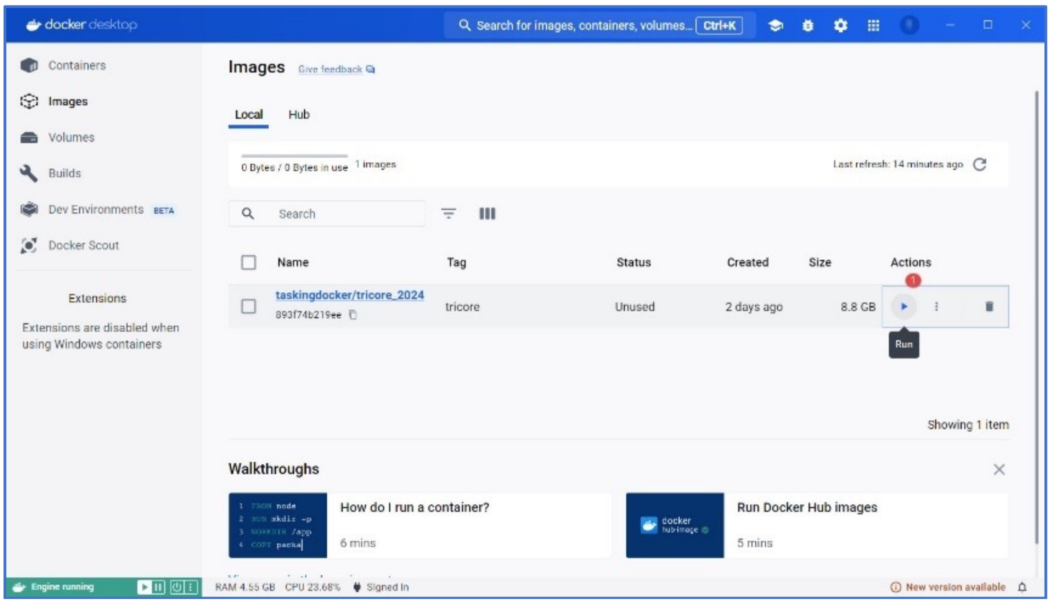
How to Compile Source Code within a Container in Docker Desktop App

Let say we have source file name `file_1.c` present on our host at location:

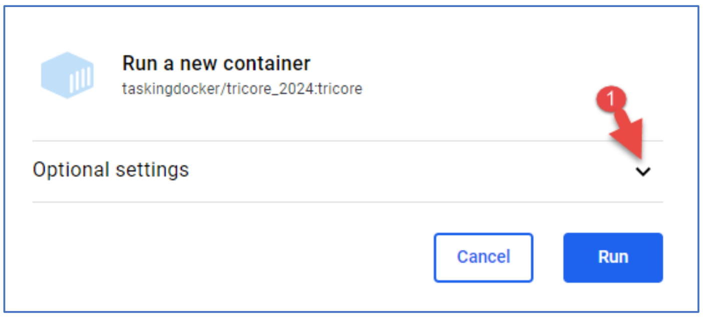
```
C:\Users\Username\Downloads\Docker_Data
```

which we want to compile within the Docker container. This can be achieved via mounting this directory into the docker container. Docker run command offer “-v” or “—volume” option to mount a directory from the Docker host into a container. This allows to share data between the host and the container.

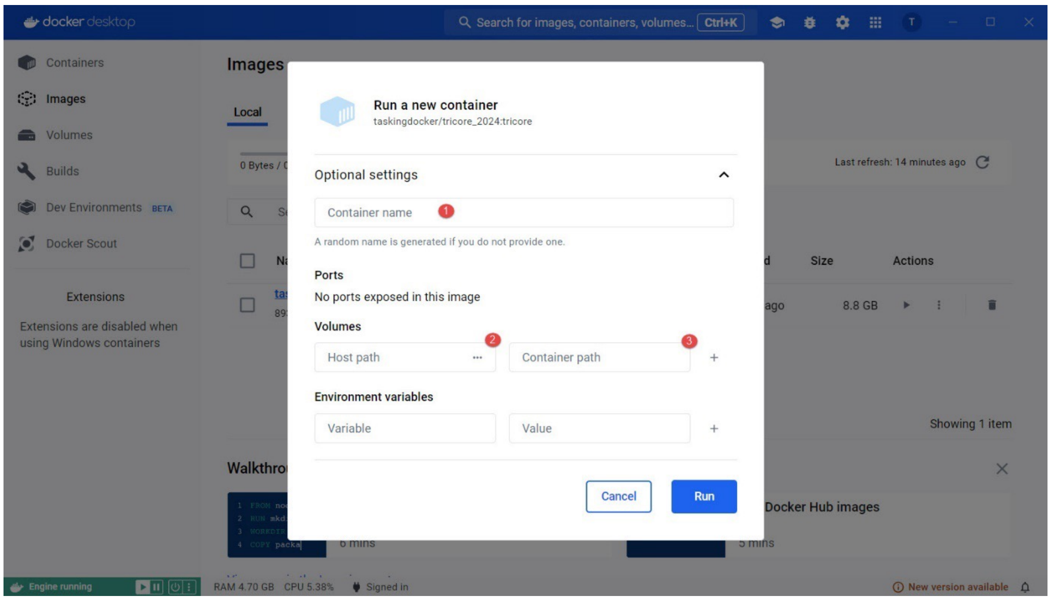
Once the Tricore Image has been created successfully, click on the "Run" (as indicated below at 1) within Docker Desktop.



This will pop-up a small options window as indicated below:



Click on the "Optional settings" drop-down menu, this will open

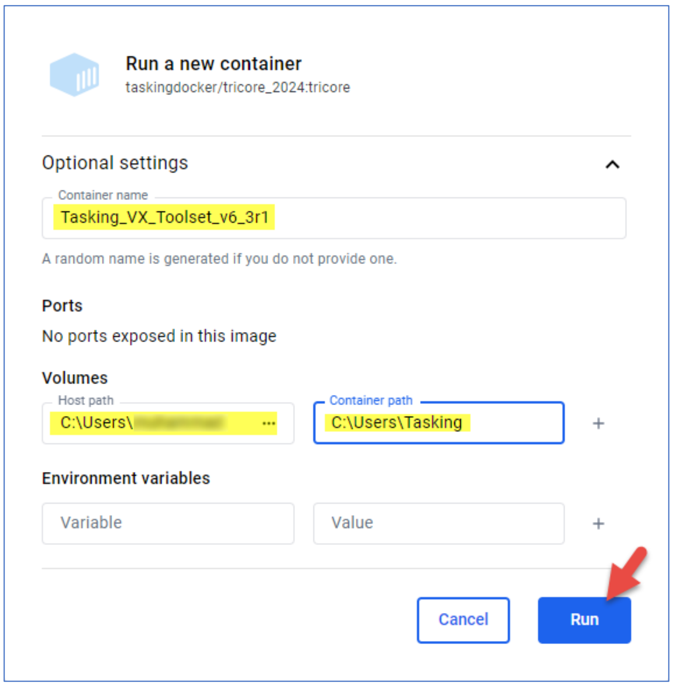


Fill in the respective entries indicated above. Description of each entry is as below:

1	Container Name	The container name in Docker is a user-assigned identifier for a specific running container instance.
2	Host Path	This is the path to a directory or file on the host system, where your source files are located. It refers to a location in the host's filesystem.
3	Container Path	This is the path to a directory or file inside the container. It refers to a location in the container's filesystem.

Note: Because the compiler generates output files, please select the host/container paths where you have read and write access.

In our use case the following settings are applied:



Run a new container
taskingdocker/tricore_2024:tricore

Optional settings

Container name: Tasking_VX_Toolset_v6_3r1

A random name is generated if you do not provide one.

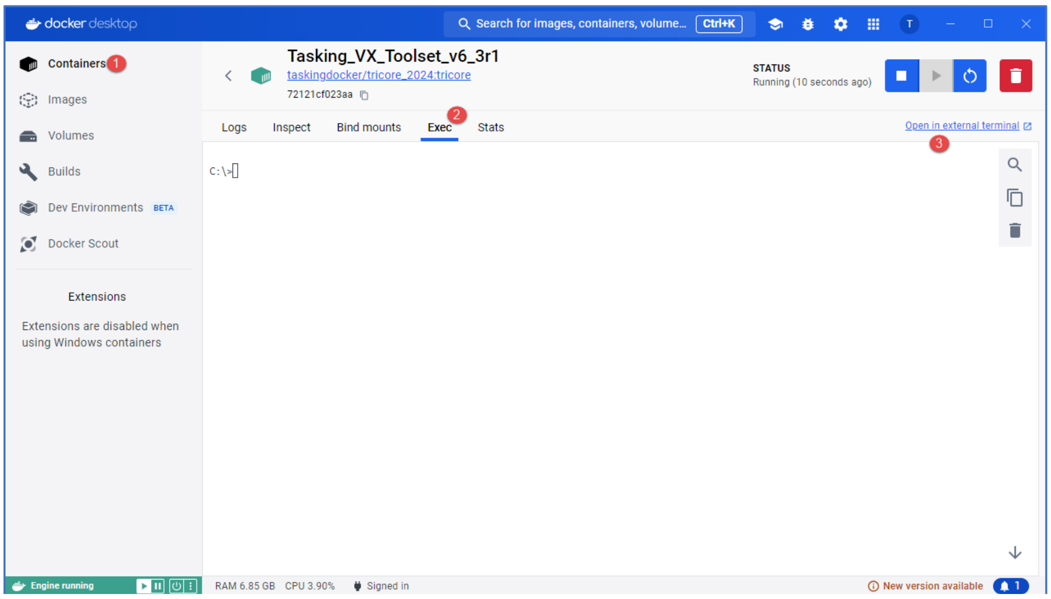
Ports
No ports exposed in this image

Volumes
Host path: C:\Users\[redacted] ···
Container path: C:\Users\Tasking

Environment variables
Variable
Value

Container Name	`Tasking_VX_Toolset_v6_3r1`
Host Path	`C:\Users\Username\Downloads\Docker_Data`
Container Path	`C:\Users\Tasking`

After setting the respective values click "Run". Once the container is running, launch the external terminal as shown below:



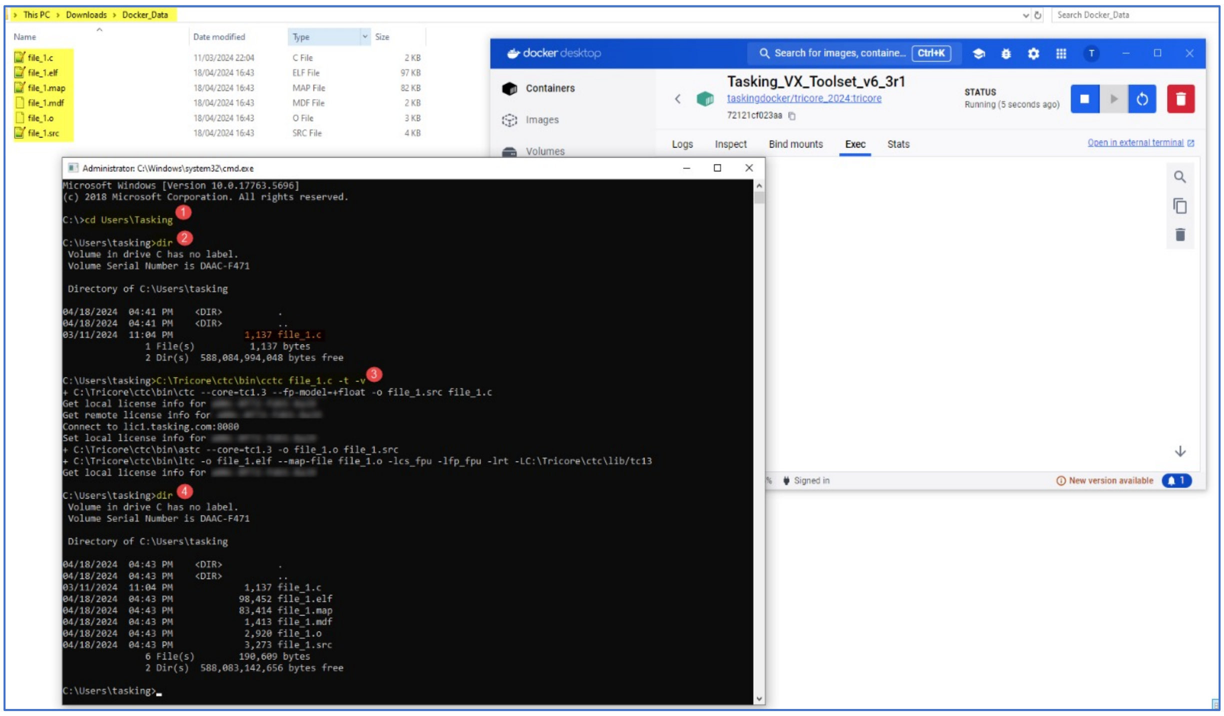
Once the external cmd.exe terminal is opened from within the container, switch to the Container path directory which you had set earlier i.e. in this use case:

```
cd Users\Tasking
```

Call the `dir` command to see the list of files present in this directory. And then call the control program to compile the source file file_1.c via :

```
C:\Tricore\ctc\bin\cctc file_1.c -t -v
```

After successful compilation execute the `dir` command to see the list of files.



Name	Date modified	Type	Size
file_1.c	11/03/2024 22:04	C File	2 KB
file_1.elf	18/04/2024 16:43	ELF File	97 KB
file_1.map	18/04/2024 16:43	MAP File	82 KB
file_1.mdf	18/04/2024 16:43	MDF File	2 KB
file_1.o	18/04/2024 16:43	O File	3 KB
file_1.src	18/04/2024 16:43	SRC File	4 KB

```
Microsoft Windows [Version 10.0.17763.5696]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>cd Users\Tasking

C:\Users\tasking>dir
 Volume in drive C has no label.
 Volume Serial Number is DAAC-F471

 Directory of C:\Users\tasking

04/18/2024  04:41 PM    <DIR>          .
04/18/2024  04:41 PM    <DIR>          ..
03/11/2024  11:04 PM             1,137 file_1.c
               1 File(s)          1,137 bytes
               2 Dir(s)  588,084,994,048 bytes free

C:\Users\tasking>C:\Tricore\ctc\bin\cctc file_1.c -t -v
+ C:\Tricore\ctc\bin\ctc --core=tc1.3 --fp-model=+float -o file_1.src file_1.c
Get local license info for [redacted]
Get remote license info for [redacted]
Connect to lic1.tasking.com:8080
Set local license info for [redacted]
+ C:\Tricore\ctc\bin\astc --core=tc1.3 -o file_1.o file_1.src
+ C:\Tricore\ctc\bin\ltc -o file_1.elf --map-file file_1.o -lcs_fpu -lfp_fpu -lrt -LC:\Tricore\ctc\lib/tc13
Get local license info for [redacted]

C:\Users\tasking>dir
 Volume in drive C has no label.
 Volume Serial Number is DAAC-F471

 Directory of C:\Users\tasking

04/18/2024  04:43 PM    <DIR>          .
04/18/2024  04:43 PM    <DIR>          ..
03/11/2024  11:04 PM             1,137 file_1.c
04/18/2024  04:43 PM            98,452 file_1.elf
04/18/2024  04:43 PM            83,414 file_1.map
04/18/2024  04:43 PM             1,413 file_1.mdf
04/18/2024  04:43 PM             2,920 file_1.o
04/18/2024  04:43 PM             3,273 file_1.src
               6 File(s)        190,609 bytes
               2 Dir(s)  588,083,142,656 bytes free

C:\Users\tasking>
```

You will observe that the compiler generated output present in container filesystem, will be visible on your host filesystem too.

How to Compile Source Code within a Container in Windows cmd.exe

If you do not want to use Docker Desktop App, you can use the Windows cmd.exe from your host PC too.
Launch two instances of cmd.exe

In the first 1st Instance of cmd.exe run the image i.e.:

```
docker run -v /host/directory:/container/directory image_name
```

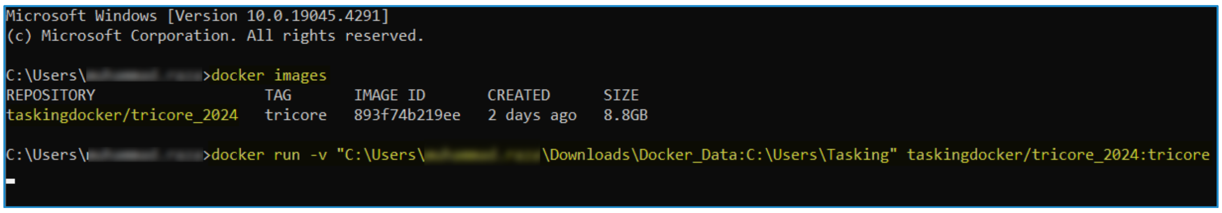
Or in case of Docker hub image use this:

```
docker run -v /host/directory:/container/directory repository_name[:tag]
```

e.g.

`docker images` \\This will give a list of all images available

```
docker run -v "C:\Users\Username\Downloads\Docker_Data:C:\Users\Tasking"
taskingdocker/tricore_2024:tricore
```



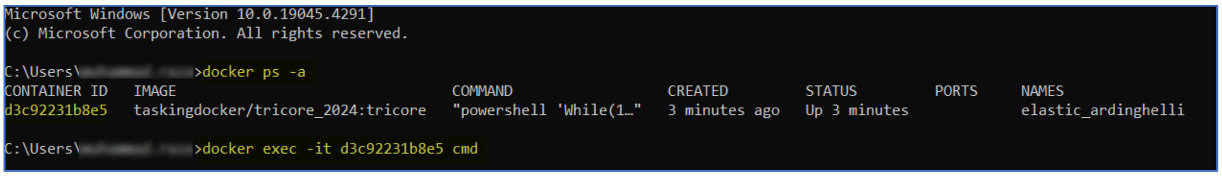
```
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\Users\[redacted]>docker images
REPOSITORY                   TAG       IMAGE ID       CREATED      SIZE
taskingdocker/tricore_2024   tricore   893f74b219ee   2 days ago   8.8GB

C:\Users\[redacted]>docker run -v "C:\Users\[redacted]\Downloads\Docker_Data:C:\Users\Tasking" taskingdocker/tricore_2024:tricore
```

In the 2nd instance of cmd.exe write this:

`docker ps -a` \\This will provide all the list of running container

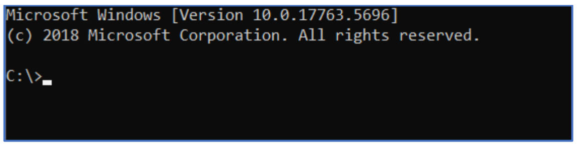
`docker exec -it cotainerID cmd` \\\\ This will execute the container in interactive mode of container cmd.exe

e.g. :



```
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\Users\[redacted]>docker ps -a
CONTAINER ID   IMAGE                                COMMAND                  CREATED         STATUS         PORTS     NAMES
d3c92231b8e5   taskingdocker/tricore_2024:tricore   "powershell 'While(1…"   3 minutes ago   Up 3 minutes             elastic_ardinghelli

C:\Users\[redacted]>docker exec -it d3c92231b8e5 cmd
```

Once you entered the above command, a command prompt (cmd.exe) window will be opened in the Docker container. i.e.



```
Microsoft Windows [Version 10.0.17763.5696]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\>
```

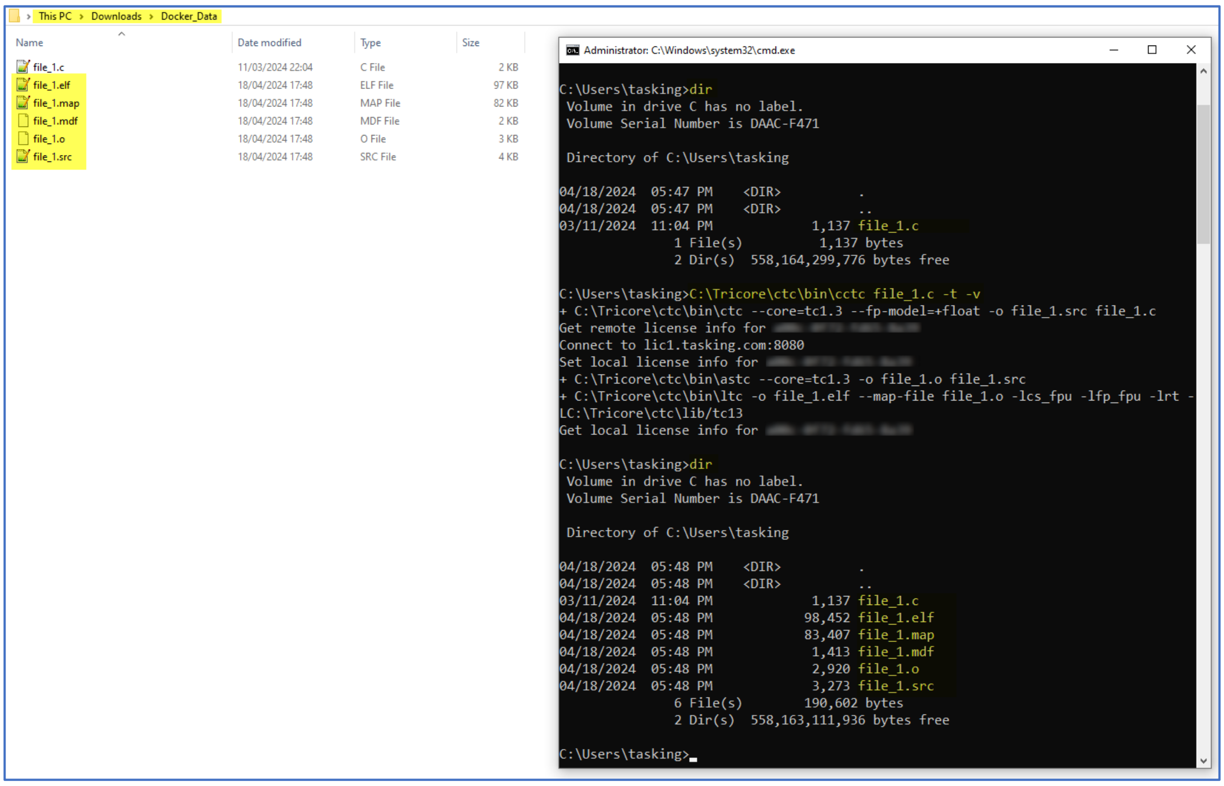
Switch to the Container path directory which you had set earlier i.e. in this use case:

```
cd Users\Tasking
```

Call the `dir` command to see the list of files present in this directory. And then call the control program to compile the source file file_1.c via:

```
C:\Tricore\ctc\bin\cctc file_1.c -t -v
```

After successful compilation execute the `dir` command to see the list of files.



This PC › Downloads › Docker_Data

Name	Date modified	Type	Size
file_1.c	11/03/2024 22:04	C File	2 KB
file_1.elf	18/04/2024 17:48	ELF File	97 KB
file_1.map	18/04/2024 17:48	MAP File	82 KB
file_1.mdf	18/04/2024 17:48	MDF File	2 KB
file_1.o	18/04/2024 17:48	O File	3 KB
file_1.src	18/04/2024 17:48	SRC File	4 KB

```
C:\Users\tasking>dir
 Volume in drive C has no label.
 Volume Serial Number is DAAC-F471

 Directory of C:\Users\tasking

04/18/2024  05:47 PM    <DIR>          .
04/18/2024  05:47 PM    <DIR>          ..
03/11/2024  11:04 PM             1,137 file_1.c
               1 File(s)          1,137 bytes
               2 Dir(s)  558,164,299,776 bytes free

C:\Users\tasking>C:\Tricore\ctc\bin\cctc file_1.c -t -v
+ C:\Tricore\ctc\bin\ctc --core=tc1.3 --fp-model=+float -o file_1.src file_1.c
Get remote license info for [redacted]
Connect to lic1.tasking.com:8080
Set local license info for [redacted]
+ C:\Tricore\ctc\bin\astc --core=tc1.3 -o file_1.o file_1.src
+ C:\Tricore\ctc\bin\ltc -o file_1.elf --map-file file_1.o -lcs_fpu -lfp_fpu -lrt -
LC:\Tricore\ctc\lib/tc13
Get local license info for [redacted]

C:\Users\tasking>dir
 Volume in drive C has no label.
 Volume Serial Number is DAAC-F471

 Directory of C:\Users\tasking

04/18/2024  05:48 PM    <DIR>          .
04/18/2024  05:48 PM    <DIR>          ..
03/11/2024  11:04 PM             1,137 file_1.c
04/18/2024  05:48 PM            98,452 file_1.elf
04/18/2024  05:48 PM            83,407 file_1.map
04/18/2024  05:48 PM             1,413 file_1.mdf
04/18/2024  05:48 PM             2,920 file_1.o
04/18/2024  05:48 PM             3,273 file_1.src
               6 File(s)        190,602 bytes
               2 Dir(s)  558,163,111,936 bytes free

C:\Users\tasking>
```

You will observe that the compiler generated output present in container filesystem, will be visible in your host filesystem too.