

Containerize a .NET 8 Web API application with Docker and upload/download the image to/from Docker Hub

0. Prerequisites

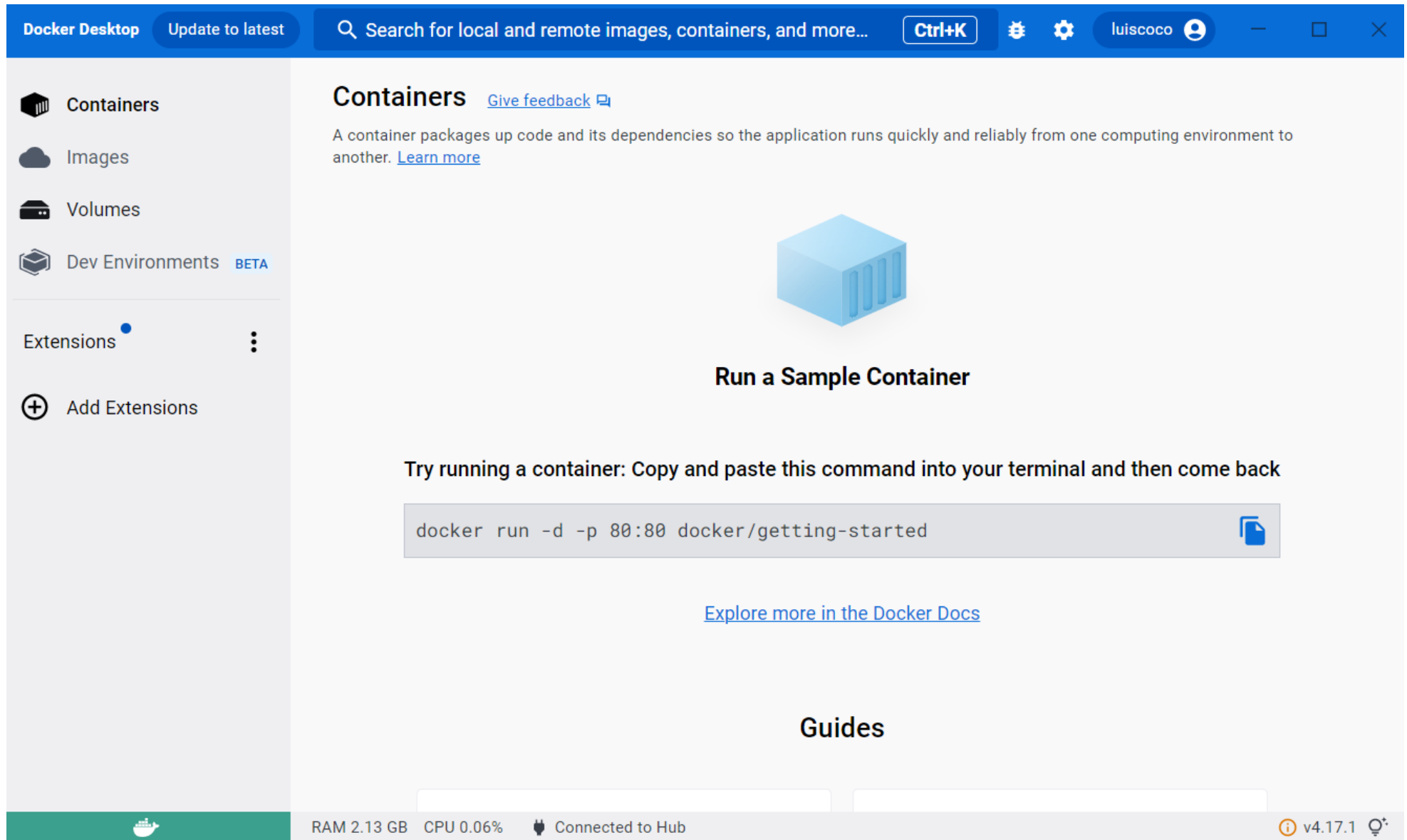
Install Docker Desktop

Create a new account in Docker Hub

Install Visual Studio 2022 Community edition

1. Create a .NET 8 Web API in Visual Studio 2022 community

IMPORTANT! Install and Run Docker Desktop before starting creating the application in Visual Studio 2022



The screenshot shows the Docker Desktop application window. The title bar includes 'Docker Desktop', an 'Update to latest' button, a search bar with the text 'Search for local and remote images, containers, and more...', a 'Ctrl+K' button, and user profile information for 'luiscoco'. The left sidebar contains navigation options: 'Containers' (selected), 'Images', 'Volumes', 'Dev Environments' (marked BETA), 'Extensions' (with a blue dot and a menu icon), and 'Add Extensions'. The main content area is titled 'Containers' with a 'Give feedback' link. It contains a description of containers, a 'Run a Sample Container' section with a terminal command, and a 'Guides' section. The bottom status bar shows system metrics (RAM 2.13 GB, CPU 0.06%), connection status (Connected to Hub), and version information (v4.17.1).

Docker Desktop Update to latest Search for local and remote images, containers, and more... Ctrl+K luiscoco

Containers [Give feedback](#)

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. [Learn more](#)

Run a Sample Container

Try running a container: Copy and paste this command into your terminal and then come back

```
docker run -d -p 80:80 docker/getting-started
```

[Explore more in the Docker Docs](#)

Guides

RAM 2.13 GB CPU 0.06% Connected to Hub v4.17.1

Run Visual Studio 2022 Community Edition and create a new project

Select the .NET Web API template





Create a new project

Search for templates (Alt+S) 🔍

All languages ▾ All platforms ▾ All project types ▾

C# Linux macOS Windows Cloud Service Web

Recent project templates

-  ASP.NET Core Web API C#
A project template for creating a RESTful Web API using ASP.NET Core controllers or minimal APIs, with optional support for OpenAPI and authentication.
C# Linux macOS Windows API Cloud Service Web
Web API
-  ASP.NET Core Web API (native AOT) C#
A project template for creating a RESTful Web API using ASP.NET Core minimal APIs published as native AOT.
C# Linux macOS Windows API Cloud Service Web
Web API
-  Class Library C#
A project for creating a class library that targets .NET or .NET Standard
C# Android Linux macOS Windows Library
-  ASP.NET Core Empty C#
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.

Back Next

Set the project name and location

Configure your new project

ASP.NET Core Web API C# Linux macOS Windows API Cloud Service Web Web API

Project name

WebAPIdotNET8

Location

C:\dotNet 8 Web API\

Solution name ⓘ

WebAPIdotNET8

☒ Place solution and project in the same directory

Project will be created in "C:\dotNet 8 Web API\WebAPIdotNET8\"

Back Next

Select the .NET 8 framework, also select **Enable Docker** for creating a dockerfile automatically

Click on the dockerfile to see the content

This is the dockefile source code

#See <https://aka.ms/customizecontainer> to learn how to customize your debug container and how Visual Studio uses this Dockerfile

```
FROM mcr.microsoft.com/dotnet/aspnet:8.0 AS base
USER app
WORKDIR /app
EXPOSE 8080
EXPOSE 8081

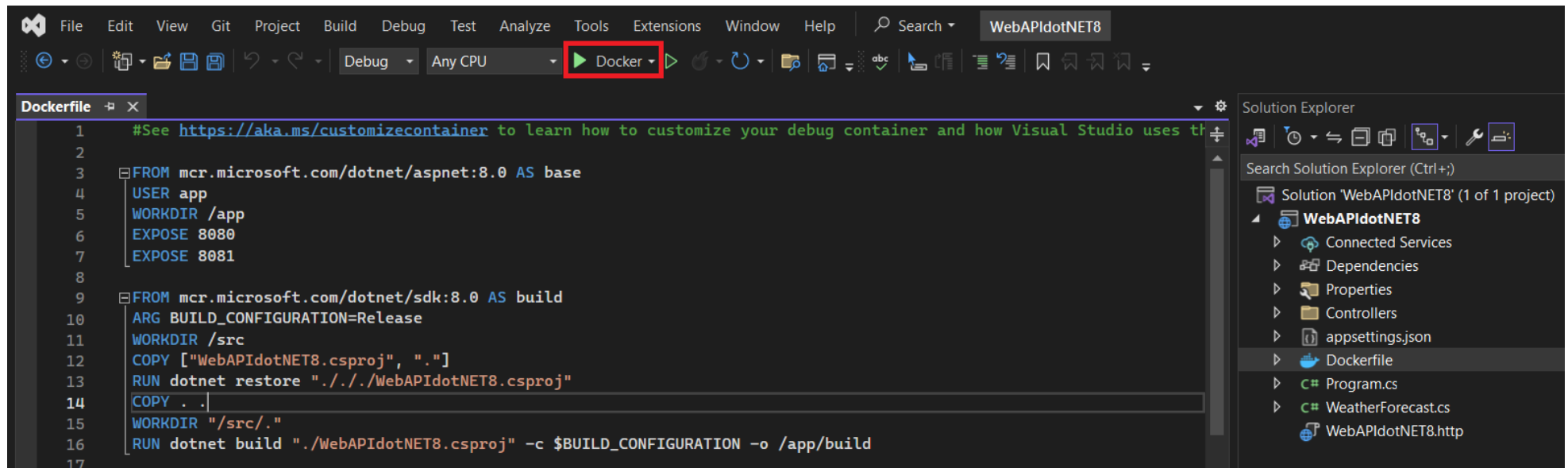
FROM mcr.microsoft.com/dotnet/sdk:8.0 AS build
ARG BUILD_CONFIGURATION=Release
WORKDIR /src
COPY ["WebAPIdotNET8.csproj", "."]
RUN dotnet restore "../WebAPIdotNET8.csproj"
COPY . .
WORKDIR "/src/."
RUN dotnet build "../WebAPIdotNET8.csproj" -c $BUILD_CONFIGURATION -o /app/build

FROM build AS publish
ARG BUILD_CONFIGURATION=Release
RUN dotnet publish "../WebAPIdotNET8.csproj" -c $BUILD_CONFIGURATION -o /app/publish /p:UseAppHost=false

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "WebAPIdotNET8.dll"]
```

2. Create and run the Docker image

We can automatically create and run the application Docker Image pressing the **Docker** button in Visual Studio 2022



See the docker image in Docker Desktop

See the docker running container in Docker Desktop

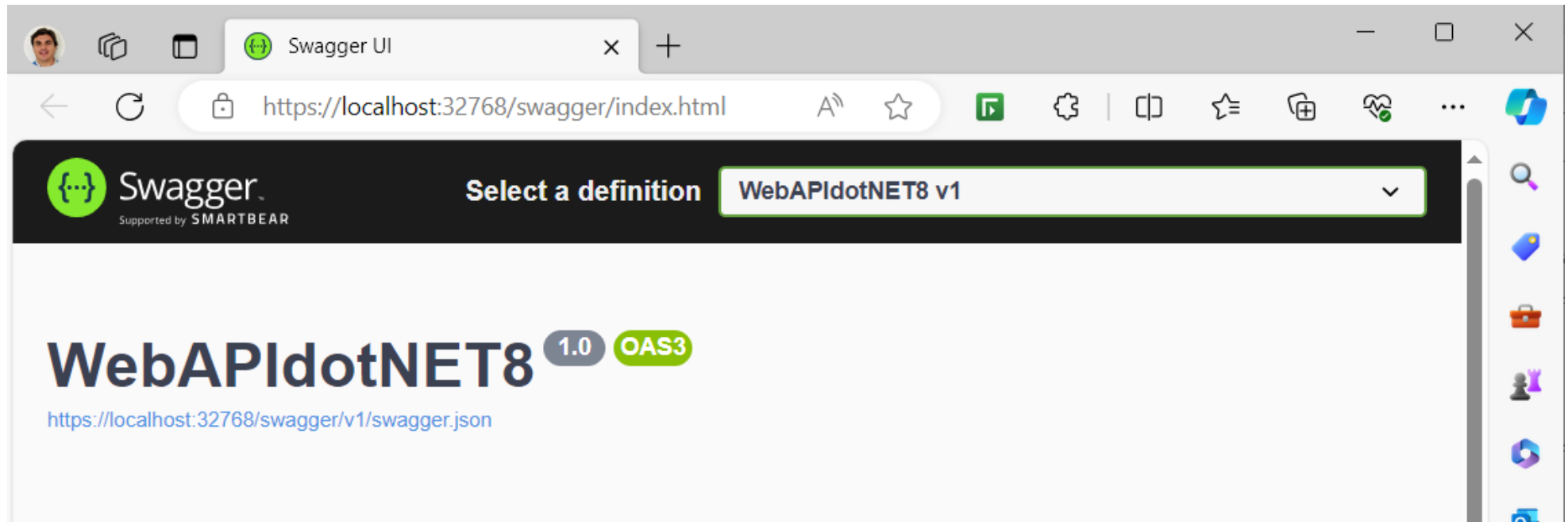
Also we can see the image with the command prompt command:

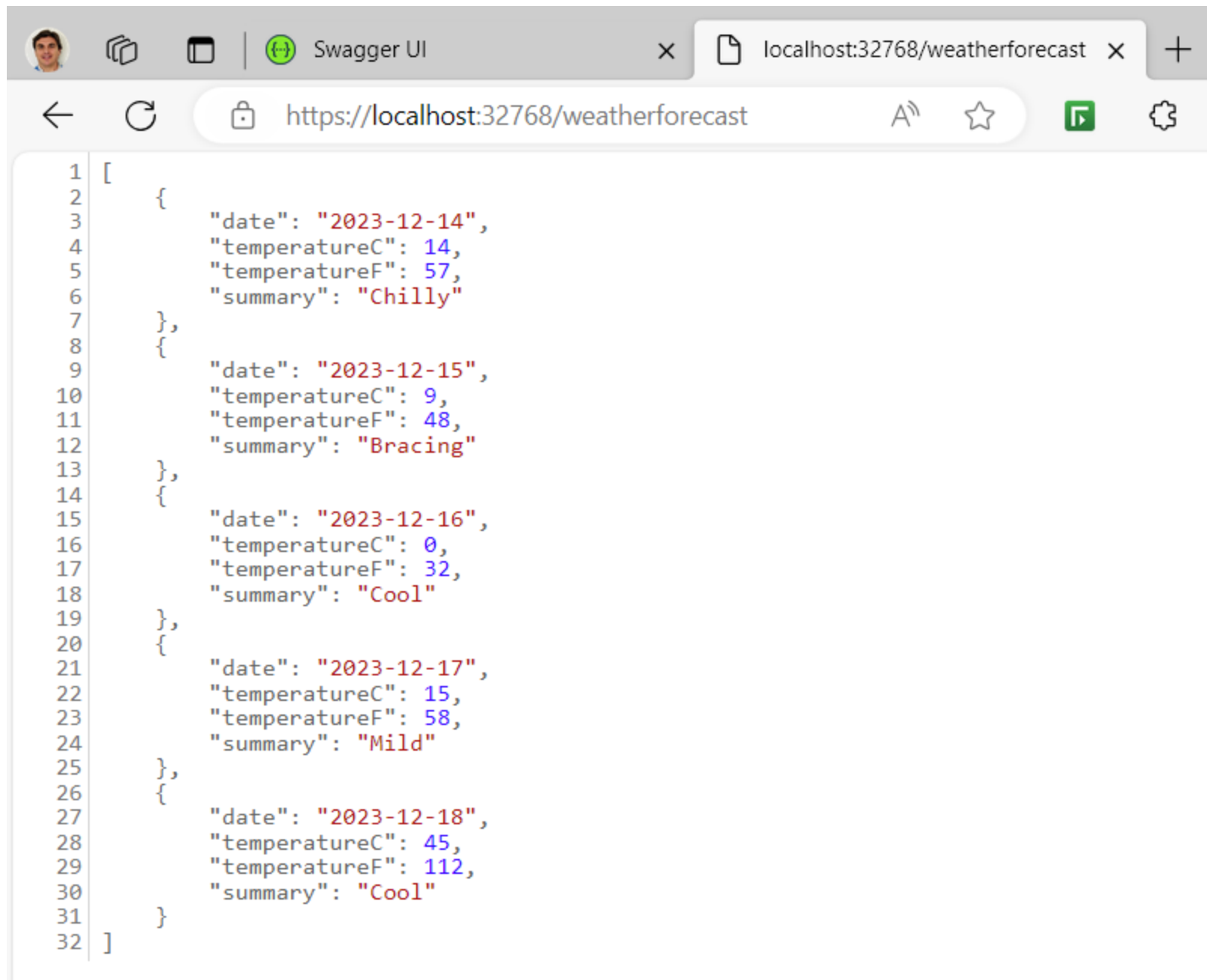
```
docker image
```

And we can see the running containers with the command:

```
docker ps
```

Finally see the Web API running container endpoints





```
1 [
2   {
3     "date": "2023-12-14",
4     "temperatureC": 14,
5     "temperatureF": 57,
6     "summary": "Chilly"
7   },
8   {
9     "date": "2023-12-15",
10    "temperatureC": 9,
11    "temperatureF": 48,
12    "summary": "Bracing"
13  },
14  {
15    "date": "2023-12-16",
16    "temperatureC": 0,
17    "temperatureF": 32,
18    "summary": "Cool"
19  },
20  {
21    "date": "2023-12-17",
22    "temperatureC": 15,
23    "temperatureF": 58,
24    "summary": "Mild"
25  },
26  {
27    "date": "2023-12-18",
28    "temperatureC": 45,
29    "temperatureF": 112,
30    "summary": "Cool"
31  }
32 ]
```

IMPORTANT NOTE:

Another option for creating automatically a dockerfile is to add "docker support" in our application

Then we select the docker virtual machine operating system, in our case we select "**Linux**"

3. Upload/download the Docker image to/from the Docker Hub

Login in Docker Hub and create a new account (Sign Up)

The screenshot shows a web browser window with multiple tabs open. The active tab is 'hub.docker.com'. The browser's address bar shows the URL 'hub.docker.com'. The Docker Hub website is displayed, featuring a blue header with the Docker Hub logo, navigation links for 'Explore' and 'Pricing', a search bar, and a 'Sign In' button. A 'Sign up' button is highlighted with a red rectangle. Below the header, the main content area has a large blue background with the text 'Build and Ship any Application Anywhere'. A white modal box is overlaid on the right side of the page, titled 'Create your account'. The modal text states 'Signing up for Docker is fast and free.' and provides three options: 'Continue with Google', 'Continue with GitHub', and 'Continue with Email'. At the bottom of the modal, there is a link that says 'Already have an account? Sign in'.

Testing "Shifts Left" with Docker's Acquisition of AtomicJar. [Learn more](#)

dockerhub Explore Pricing

Search Docker Hub ctrl+K ?

Sign In Sign up

Build and Ship any Application Anywhere

Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.

Create your account

Signing up for Docker is fast and free.

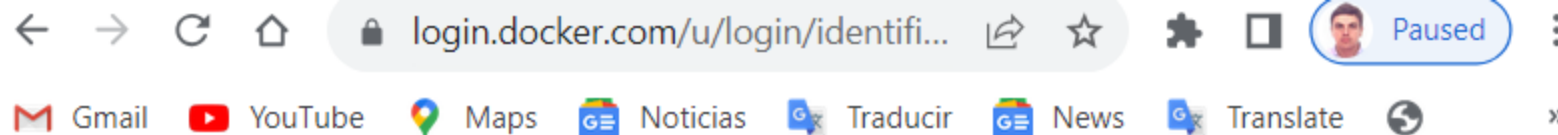
[Continue with Google](#)

[Continue with GitHub](#)

[Continue with Email](#)

[Already have an account? Sign in](#)

After creating an account we Sing in



Don't have an account? [Sign Up](#)



Sign in

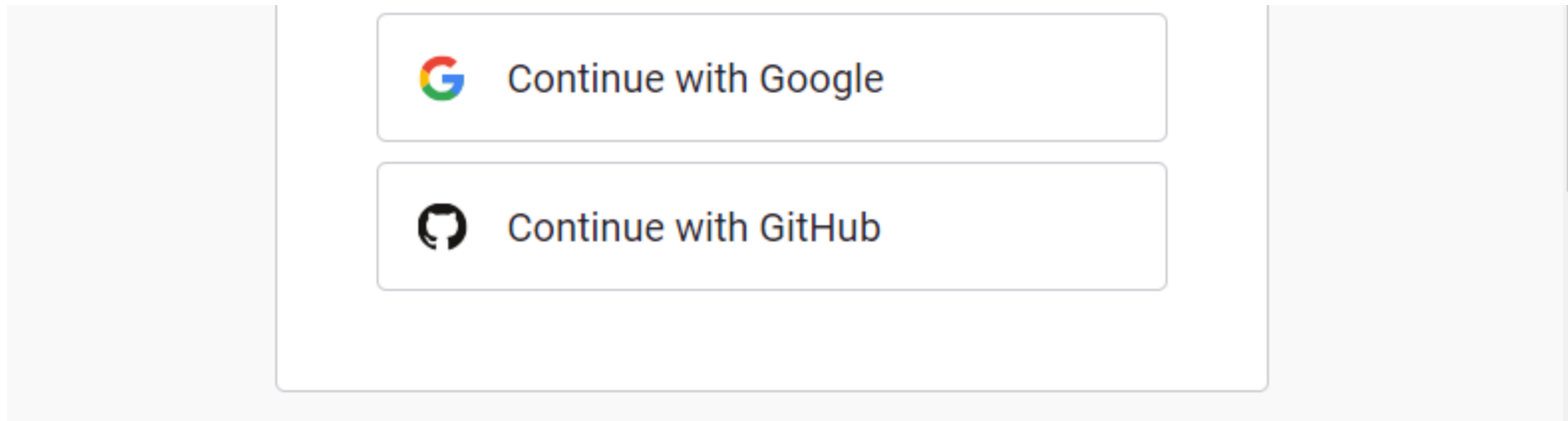
Sign in to Docker to continue to Docker Hub.

Username or email address

|

Continue

OR



Press the "Create Repository" button

hub.docker.com/repositories/luisccoco

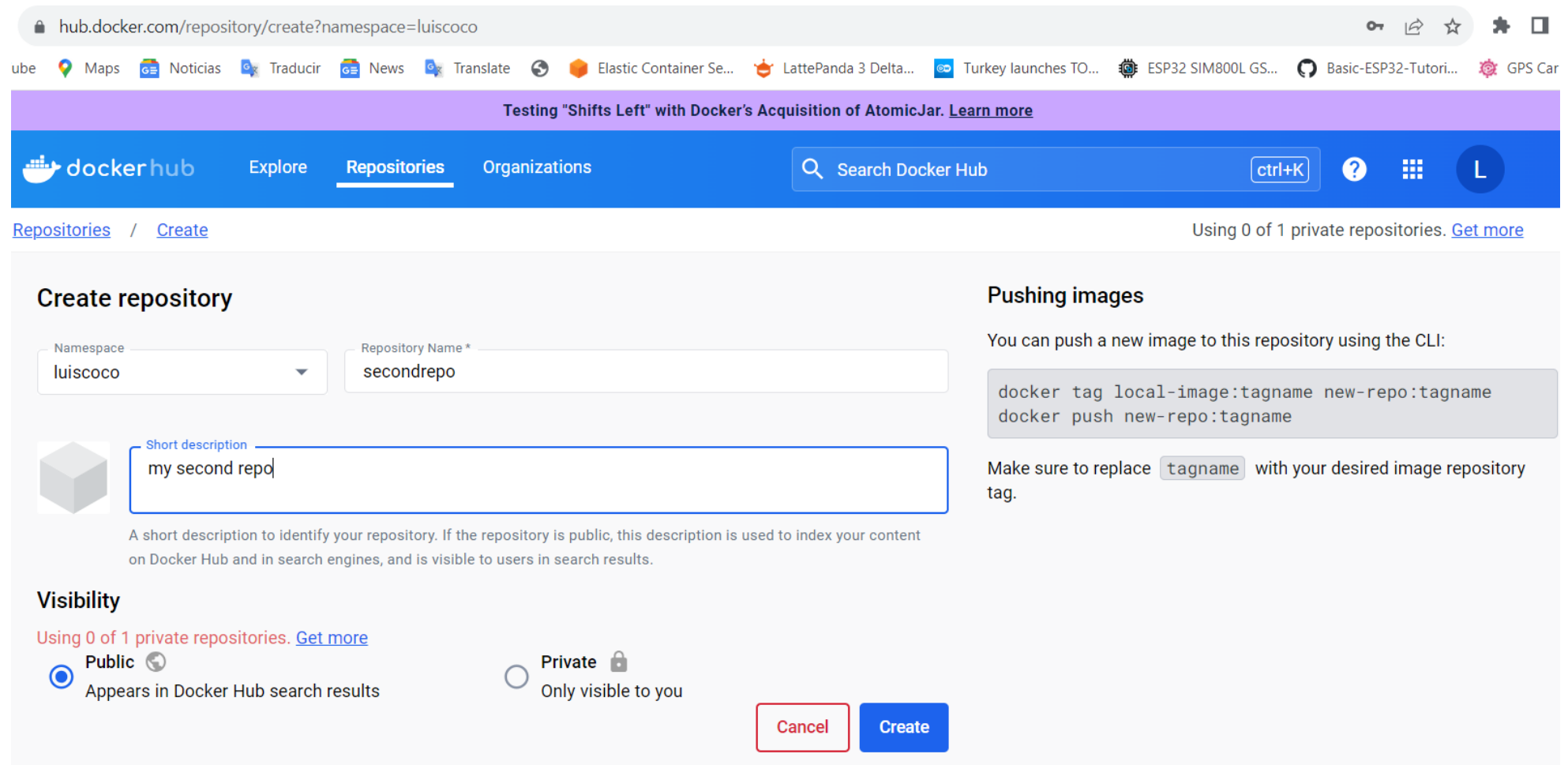
Testing "Shifts Left" with Docker's Acquisition of AtomicJar. [Learn more](#)

docker hub Explore **Repositories** Organizations Search Docker Hub ctrl+K ? [User Avatar]

luisccoco Search by repository name All Content **Create repository**

luisccoco / webapi Contains: Image Last pushed: 4 hours ago	Inactive	0	3	Public
luisccoco / shoppingapi Contains: Image Last pushed: a year ago	Inactive	0	40	Public
luisccoco / shoppingclient Contains: Image Last pushed: a year ago	Inactive	0	7	Public

We set the new repo name, we select public or private repo and we include a repo description.



hub.docker.com/repository/create?namespace=luiscoco

Testing "Shifts Left" with Docker's Acquisition of AtomicJar. [Learn more](#)

dockerhub Explore **Repositories** Organizations

Search Docker Hub ctrl+K ? ⌵ L

[Repositories](#) / [Create](#) Using 0 of 1 private repositories. [Get more](#)

Create repository


Namespace: luiscoco Repository Name *: secondrepo


Short description: my second repo

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

Visibility

Using 0 of 1 private repositories. [Get more](#)

☒ **Public**  Appears in Docker Hub search results

☐ **Private**  Only visible to you

Cancel Create

Pushing images

You can push a new image to this repository using the CLI:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

Make sure to replace tagname with your desired image repository tag.

We can Push images to this repo with the following commands:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

We can see the repositories list

4. Create the docker image and upload to the new repo in Docker Hub

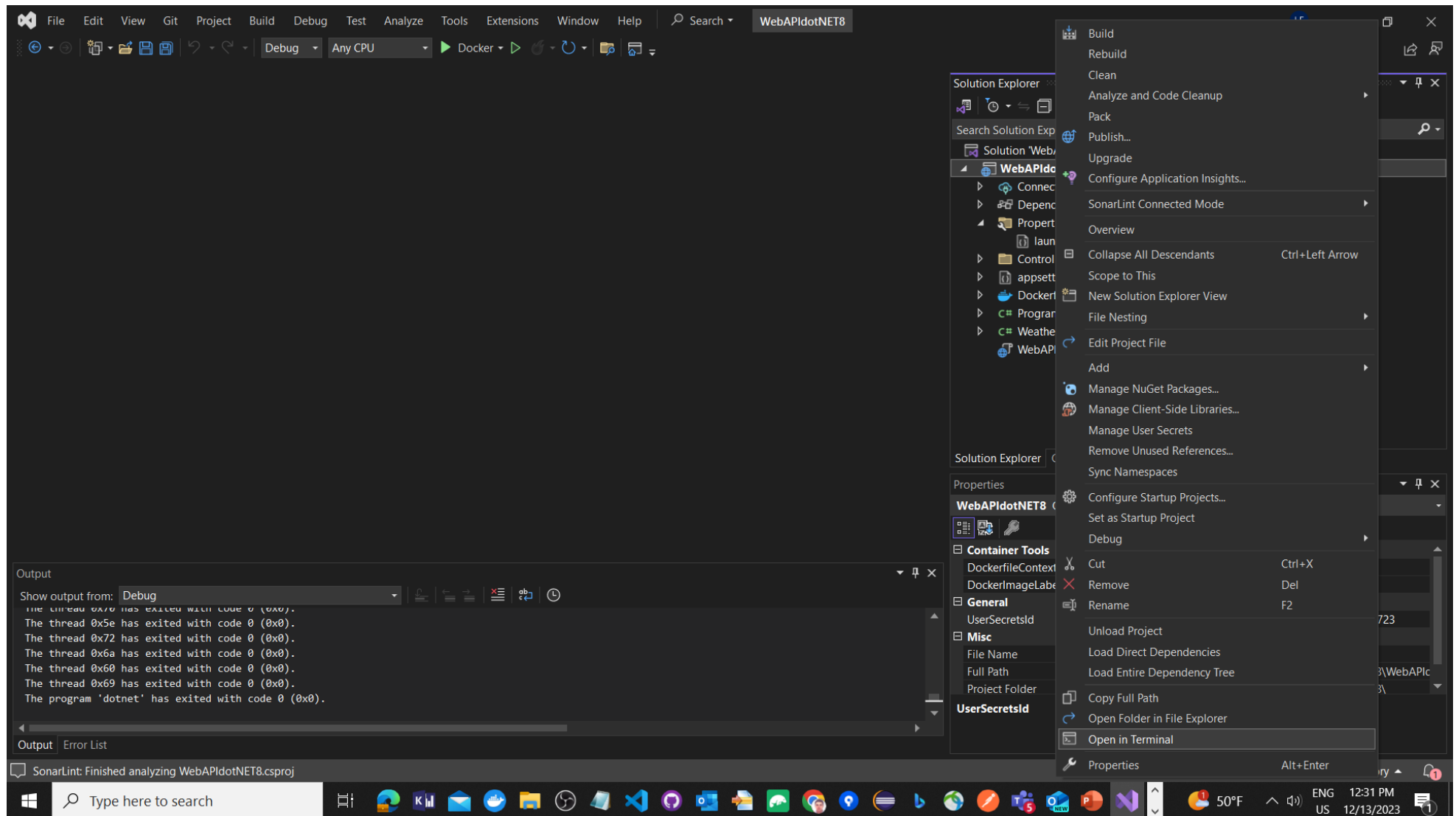
This is the general syntax for creating a new image that we would like to upload to the Docker Hub repo.

Pay attention we have to include the docker hub repo name then the docker image name and finally the docker image tag.

```
docker build -t dockerreponame/dockerimagename:tag .
```

For a real example, we right click on the project name and we select the menu option "**Open in Terminal**" then we run the following command:

```
docker build -t luisccoco/webapidotnet8:latest .
```

The screenshot shows the Visual Studio IDE with the 'WebAPIdotNET8' project open. The Developer PowerShell terminal at the bottom displays the output of the 'docker build' command. The command 'docker build -t luiscoco/webapidotnet8:latest .' is highlighted with a red box. The output shows the build process, including loading the Dockerfile, transferring the context, and building the image. The Solution Explorer on the right shows the project structure, including files like 'launchSettings.json', 'Controllers', 'appsettings.json', 'Dockerfile', 'Program.cs', 'WeatherForecast.cs', and 'WebAPIdotNET8.http'.

```

PS C:\dotNet 8 Web API\WebAPIdotNET8> docker build -t luiscoco/webapidotnet8:latest .
[+] Building 10.4s (12/17)
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 882B                                              0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 464B                                                 0.0s
=> [internal] load metadata for mcr.microsoft.com/dotnet/sdk:8.0                0.4s
=> [internal] load metadata for mcr.microsoft.com/dotnet/aspnet:8.0             0.3s
=> [build 1/7] FROM mcr.microsoft.com/dotnet/sdk:8.0@sha256:bb65e39b662be0265f780afae9cdbfcaa315ef63edb245ad9fb2aa1aabca0b6b 0.0s
=> [base 1/2] FROM mcr.microsoft.com/dotnet/aspnet:8.0@sha256:f4cb736b1220b1013e7a0f77e49487c5cc1fe643bf41a750306c4a1caa7faa97 0.0s
=> [internal] load build context                                                0.1s
=> => transferring context: 6.14kB                                              0.0s
=> CACHED [build 2/7] WORKDIR /src                                              0.0s
=> [build 3/7] COPY [WebAPIdotNET8.csproj, .]                                  0.0s
  
```

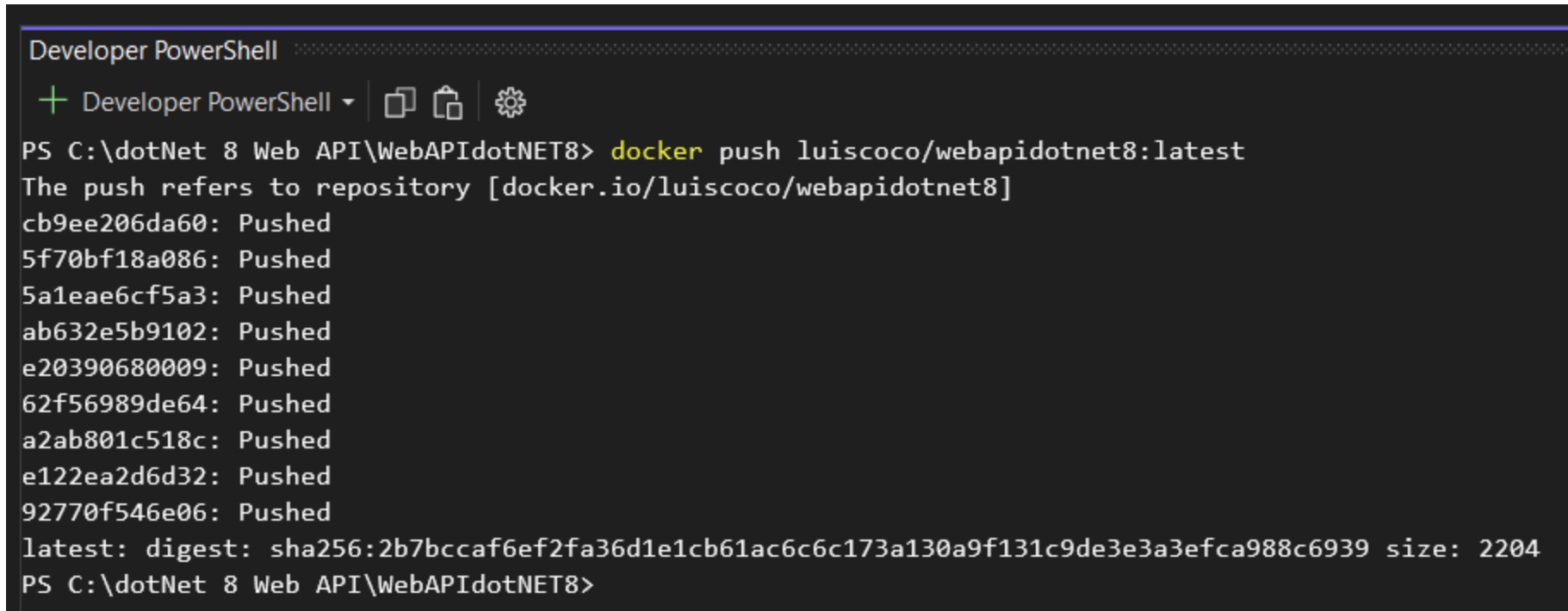
We check we create the docker image and we run the docker container

Then we use the docker push command to upload the image to the Docker Hub repository:

```
docker push luiscoco/webapidotnet8:latest
```

Also we can pull the docker image from the Docker Hub repository with the following command:

```
docker pull luiscoco/webapidotnet8
```



```
Developer PowerShell
+ Developer PowerShell
PS C:\dotNet 8 Web API\WebAPIdotNET8> docker push luiscoco/webapidotnet8:latest
The push refers to repository [docker.io/luiscoco/webapidotnet8]
cb9ee206da60: Pushed
5f70bf18a086: Pushed
5a1eae6cf5a3: Pushed
ab632e5b9102: Pushed
e20390680009: Pushed
62f56989de64: Pushed
a2ab801c518c: Pushed
e122ea2d6d32: Pushed
92770f546e06: Pushed
latest: digest: sha256:2b7bccaf6ef2fa36d1e1cb61ac6c6c173a130a9f131c9de3e3a3efca988c6939 size: 2204
PS C:\dotNet 8 Web API\WebAPIdotNET8>
```

See the new docker image in the repo

hub.docker.com/repository/docker/luiscoco/webapidotnet8/general

Tube Maps Noticias Traducir News Translate Elastic Container Se... LattePanda 3 Delta... Turkey launches TO... ESP32 SIM800L GS... Basic-ESP32-Tutori... GPS Ca

Testing "Shifts Left" with Docker's Acquisition of AtomicJar. [Learn more](#)

dockerhub Explore Repositories Organizations Search Docker Hub ? L

luiscoco / [Repositories](#) / [webapidotnet8](#) / [General](#) Using 0 of 1 private repositories. [Get more](#)

General Tags Builds Collaborators Webhooks Settings

Add a short description for this repository Update
The short description is used to index your content on Docker Hub and in search engines. It's visible to users in search results.

luiscoco / webapidotnet8

Description
This repository does not have a description

Last pushed: a minute ago

Docker commands Public View
To push a new tag to this repository:

```
docker push luiscoco/webapidotnet8:tagname
```

5. We pull the docker image from Docker hub and we run in Docker Desktop

We pull the image with the command:

```
docker pull luiscoco/webapidotnet8
```

Then we open Docker Desktop and we confirm the docker image was downloaded but it is not yet running in a container

To run the docker image in a container, we press the run button (see the following image), or we type this command

<input type="checkbox"/>	Name	Tag	Status	Created	Size	Actions
<input type="checkbox"/>	luiscoco/webapidotnet8 e8790c3ac9b4	latest	Unused	9 minutes ago	220.59 MB	

Or we can type the following command, taking into account we map the launchSetting.json http port with the 8080 port in dockerfile

```
docker run -d -p 5269:8080 luiscoco/webapidotnet8:latest
```

And also we can see the docker logs, see this picture

```
Developer PowerShell
+ Developer PowerShell |   
PS C:\dotNet 8 Web API\WebAPIdotNET8> docker run -d -p 5269:8080 luiscoco/webapidotnet8:latest
1910b41deb65d7f032a08a27396a914b36f77db98756f83e6aa8f67bceb8931e
PS C:\dotNet 8 Web API\WebAPIdotNET8> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
1910b41deb65   luiscoco/webapidotnet8:latest       "dotnet WebAPIdotNET..." 3 seconds ago  Up 3 seconds  8081/tcp, 0.0.0.0:5269->8080/tcp    festive_khorana
PS C:\dotNet 8 Web API\WebAPIdotNET8> docker logs festive_khorana
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://[::]:8080
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Production
info: Microsoft.Hosting.Lifetime[0]
      Content root path: /app
PS C:\dotNet 8 Web API\WebAPIdotNET8>
```

Now in Docker Desktop we can see the docker container is running

We can also navigate to the Web API endpoints:

<http://localhost:5269/weatherforecast>



```
1 [
2   {
3     "date": "2023-12-14",
4     "temperatureC": 5,
5     "temperatureF": 40,
6     "summary": "Freezing"
7   },
8   {
9     "date": "2023-12-15",
10    "temperatureC": 30,
11    "temperatureF": 85,
12    "summary": "Hot"
13  },
14  {
15    "date": "2023-12-16",
16    "temperatureC": 26,
17    "temperatureF": 78,
18    "summary": "Freezing"
19  },
20  {
21    "date": "2023-12-17",
22    "temperatureC": -8,
23    "temperatureF": 18,
24    "summary": "Sweltering"
25  },
26  {
27    "date": "2023-12-18",
28    "temperatureC": 53,
29    "temperatureF": 127,
30    "summary": "Bracing"
31  }
32 ]
```