codewithmukesh

what and why? 🤔

DOCKER

FOR .NET DEVELOPERS

What is docker?

Why do you need it?

& some basic docker commands

mukesh murugan
@iammukeshm

codewithmukesh

Imagine..

Imagine that you are all set with your new **.NET Application** and ready to deploy it to higher environments!

Everything is working as expected on my development machine! Let's deploy this thing!

mukesh murugan
@iammukeshm

codewithmukesh

But then you realize that there are several issues when deployed to other environments, or worst case, in production!

We have all faced this at some point, right?

mukesh murugan
@iammukeshm

codewithmukesh

Then happens the famous..

" But it works on my machine!"

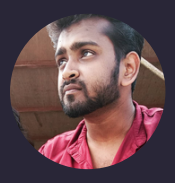
mukesh murugan

@iammukeshm

codewithmukesh

What could be the potential reasons?

- Packages Dependencies
- Framework Issues
- Misconfigured Environment Variables
- OS Level Mismatches
- Inconsistent System Configurations
- Service Interactions
- You might have forgotten to set a feature flag!
- And many other things that you never expected!



mukesh murugan
@iammukeshm

codewithmukesh

The Solution?

We need a standardized way to package our .NET application along with everything it needs to function, in any environment.

Docker is a tool that allows you to build, test and deploy applications quickly, using containers and images.

mukesh murugan
@iammukeshm

codewithmukesh

Docker..

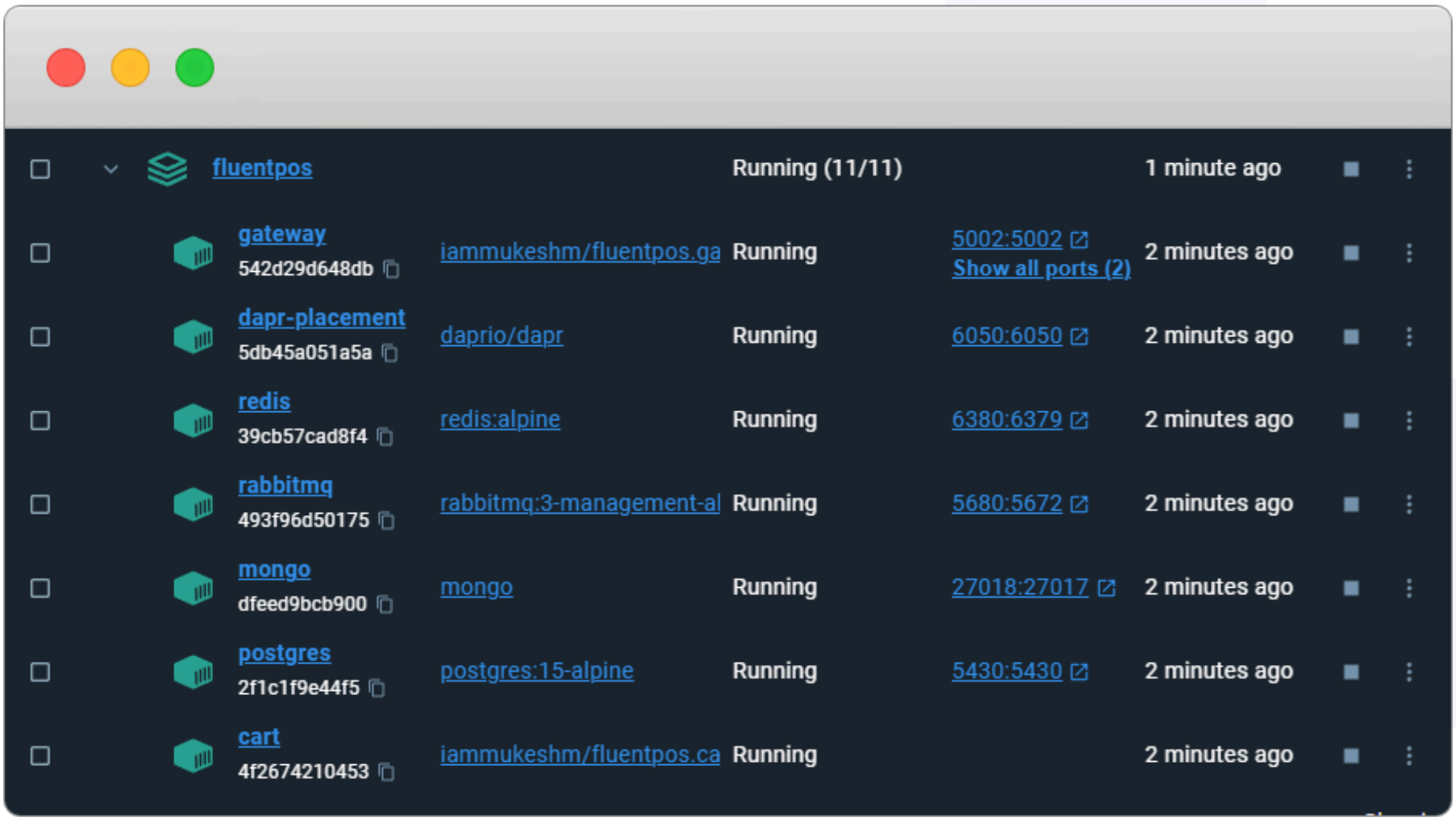
Docker essentially can package your application and all its related dependencies into Docker Images. These images can be pushed to a central repository, for example, Docker Hub or Amazon Elastic Container Registry, and can be run or any server!

mukesh murugan
@iammukeshm

codewithmukesh

Each image represents a blueprint for an application. These Images can be downloaded to any location and you can have them run as containers, which are isolated processes running on your machine.

Here is how a bunch of microservices run on Docker!



mukesh murugan
@iammukeshm

codewithmukesh

Dockerfile

Here is a sample Dockerfile, which acts as instructions for Docker to build an image from a .NET application.
It downloads the required .NET SDK (for build), copies all the project content to a working directory, and publishes the required DLLs in Release mode. You can also set the Environment variables for your application here.

You would have to have docker-desktop running on your machine, and this dockerfile would sit right beside your .csproj file.

```
FROM mcr.microsoft.com/dotnet/sdk:6.0 as build-env
WORKDIR /src
COPY *.csproj .
RUN dotnet restore
COPY . .
RUN dotnet publish -c Release -o /publish

FROM mcr.microsoft.com/dotnet/aspnet:6.0 as runtime
WORKDIR /publish
COPY --from=build-env /publish .
ENV ASPNETCORE_URLS=http://+:5000
EXPOSE 5000
ENTRYPOINT ["dotnet", "HelloDocker.dll"]
```

mukesh murugan
@iammukeshm

codewithmukesh

Basic Commands

To get started, ensure that you have docker-desktop downloaded and running on your machine. Here are some important docker cli commands that you must know.

- **docker version** : gets the current version of the docker instance.
- **docker ps** : lists all the running containers.
- **docker images** : lists all available images locally on your machine.
- **docker run <image:tag>** : spins up a new container using an image. For example, 'docker run redis:latest'
- **docker pull <image:tag>** : pulls an image with a specific tag onto your local machine. For example, 'docker pull redis:latest'
- **docker stop <container name / id>** : stops a particular container.
- **docker rm <container name / id>** : removed the container from your instance.
- **docker rmi <image>** : can remove the copy of the image from your local.
- **docker exec <container name>** : helps you run a command inside your container. I use 'docker exec -it redis bash' a lot.
- **docker build -t <image_name> <dockerfile_path>** : using the instructions from the dockerfile, build a new image with a custom name and tag. Remember the dockerfile we created earlier? You can build an image with it --> 'docker build -t HelloWorld:1.0 .' The dot (.) at the end is pretty important, since it specifies the location of your dockerfile. In my case, I was running the command from the directory where the dockerfile exists.

mukesh murugan
@iammukeshm

codewithmukesh

Join the .NET Zero to Hero Series!

One Awesome Email Every Week. Over 10+ interesting topics covered till now, and so much more to come.

Join by Subscribing.

Link in the Description!

mukesh murugan
@iammukeshm

codewithmukesh

WAS THIS HELPFUL?

Share with a friend who needs it!

mukesh murugan
@iammukeshm

Follow me for more!