Introdução a Docker

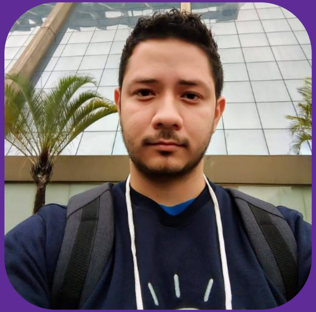
By Rodrigo C. Moraes

https://github.com/rodrigocmoraes

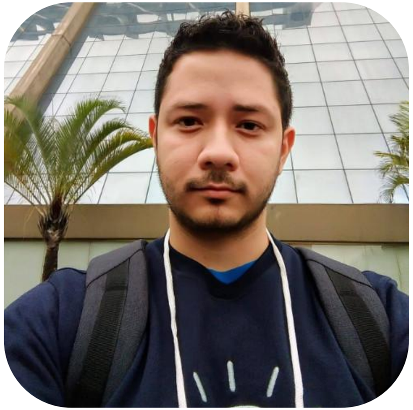
rdcmdev@gmail.com

kaggle https://www.kaggle.com/rdcmdev

@rodrigocmoraes



Quem sou eu?



UEA UNIVERSIDADE DO ESTADO DO AMAZONAS

itriad. Instituto Triad de Pesquisa e Desenvolvimento

Engenheiro de Machine Learning

Graduando em Engenharia de Computação

Ex Maratonista de Programação

...

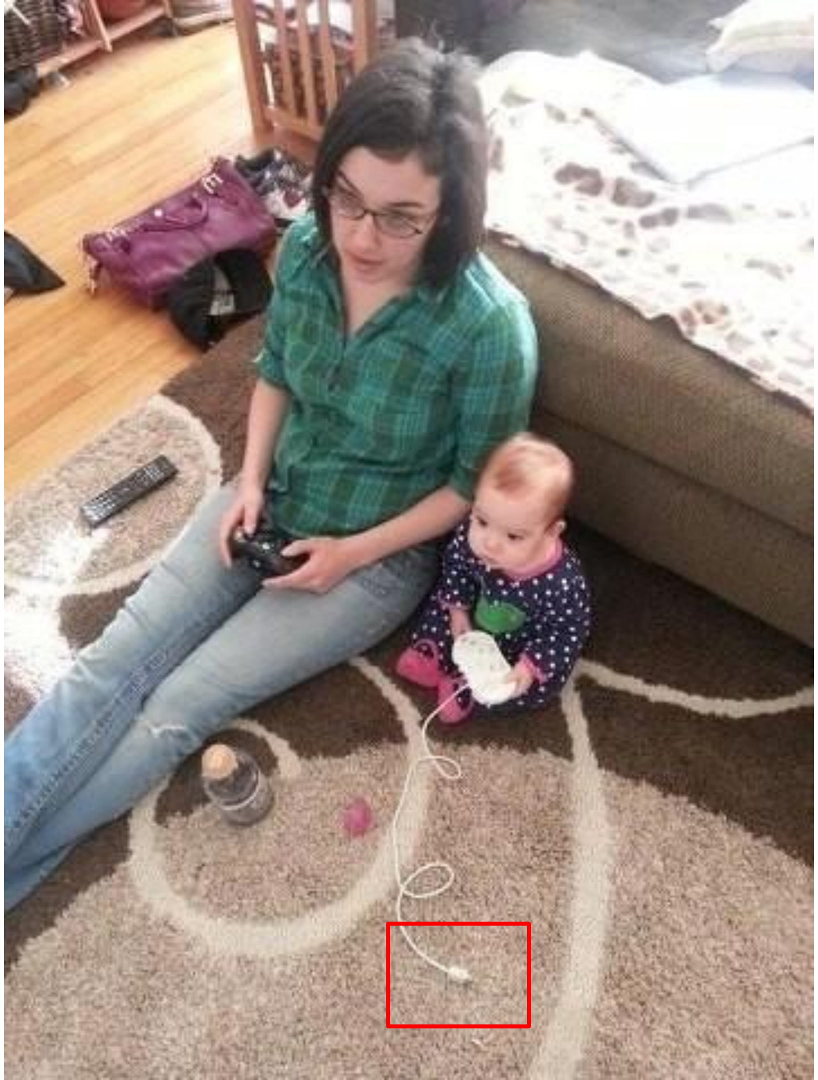
Programação

1. Conceitos
2. Tecnologia Docker
3. Hands-on

Problema

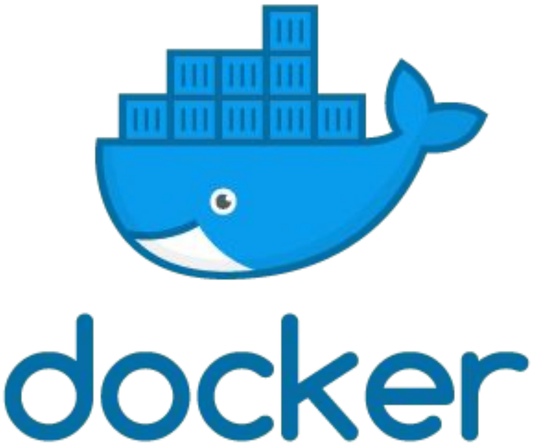
Conceitos

RadioShack
OUSE
AWESOME
SERVICE
7 Av

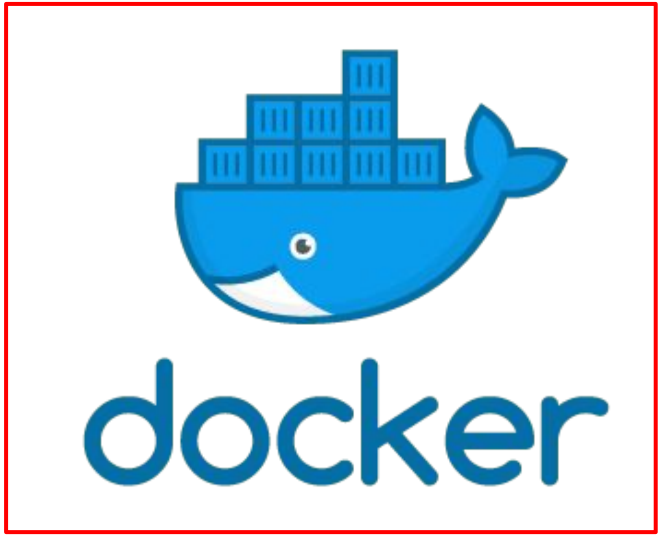


Soluções

Windows
Server



Soluções



Tecnologia Docker

LXC vs docker

- Virtualização em nível de SO

LXC

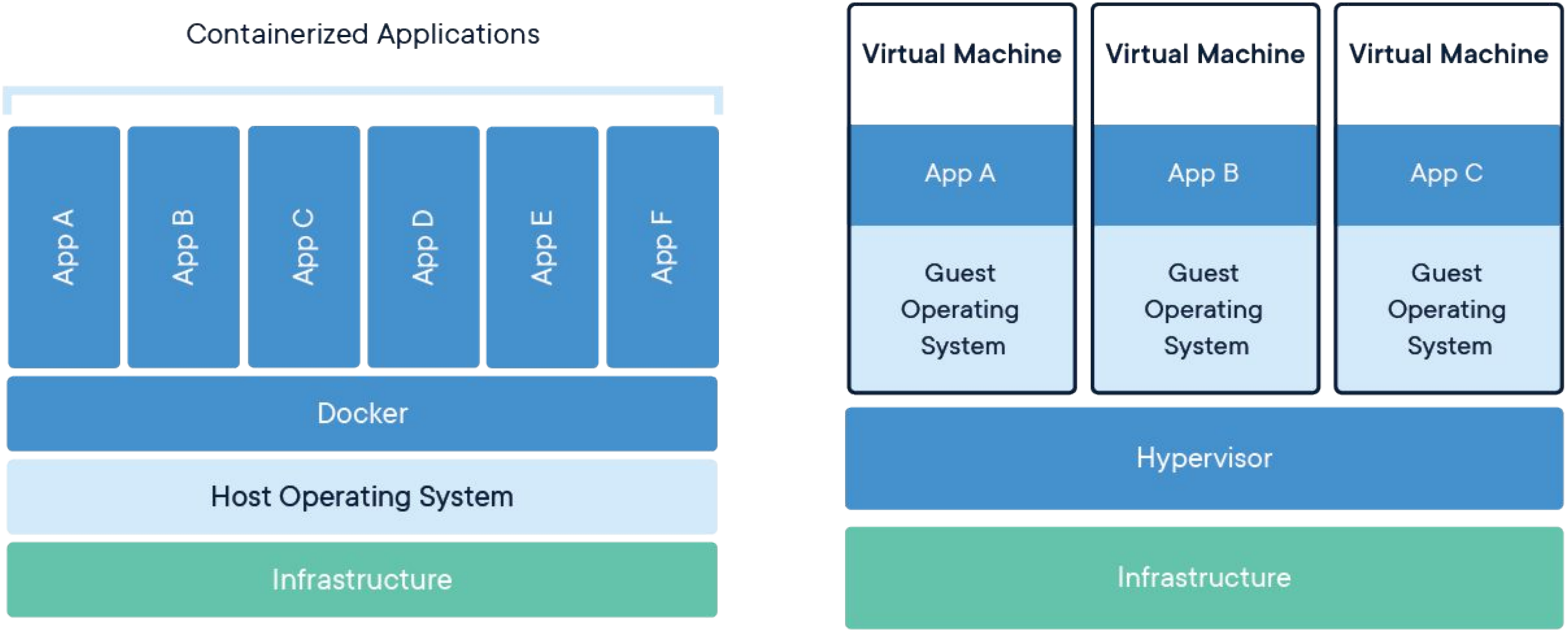
- Virtualização em nível de SO
- Linux Virtual Environment

- LXC de aplicação única

- LXC de aplicação única
- Virtualização mais leve

- LXC de aplicação única
- Virtualização mais leve
- Compartilhamento de recurso com Host

Virtual Machine vs docker



Containerized Applications

App A

App B

App C

App D

App E

App F

Docker

Host Operating System

Infrastructure

Virtual Machine

App A

Guest Operating System

Virtual Machine

App B

Guest Operating System

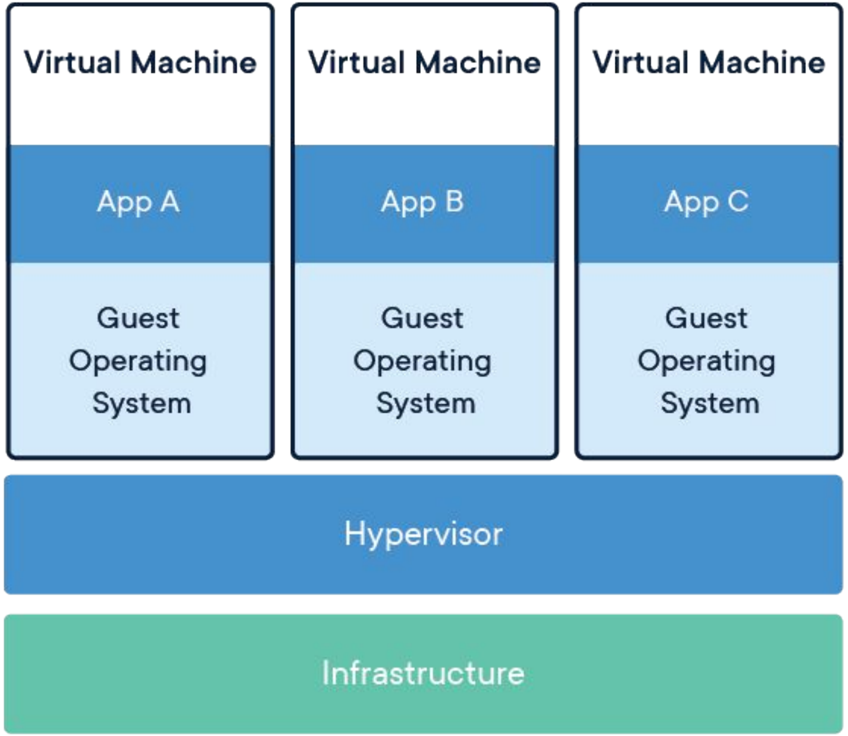
Virtual Machine

App C

Guest Operating System

Hypervisor

Infrastructure



Virtual Machine

App A

Guest Operating System

Virtual Machine

App B

Guest Operating System

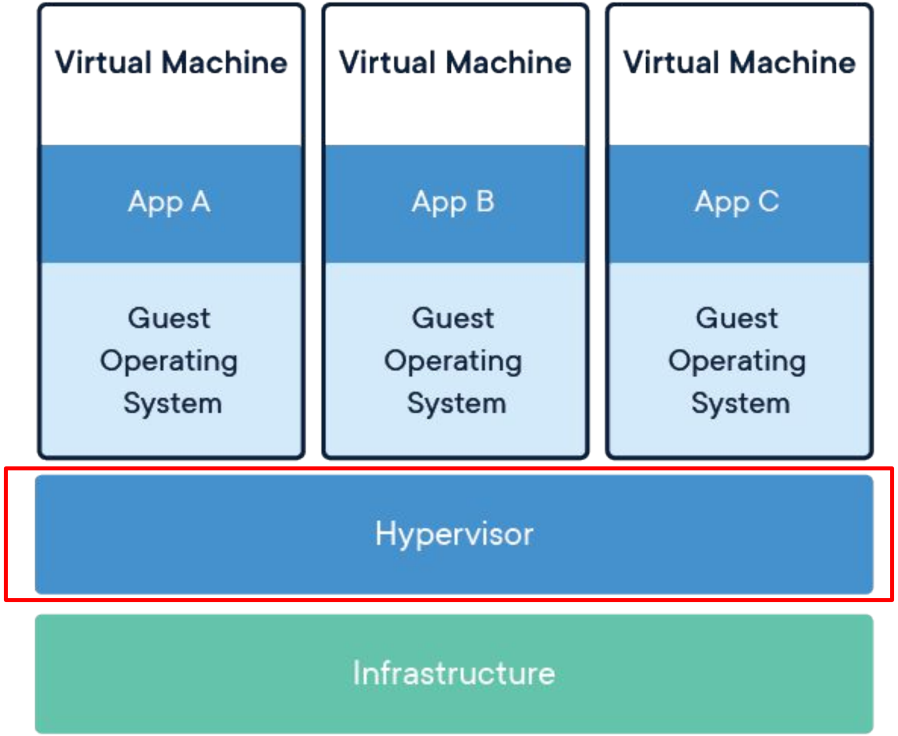
Virtual Machine

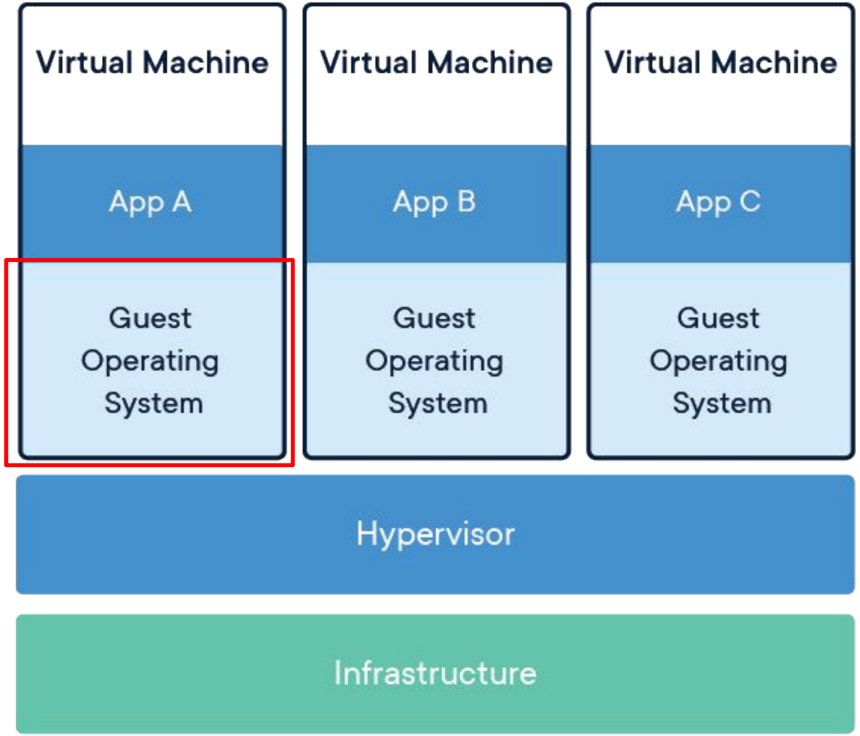
App C

Guest Operating System

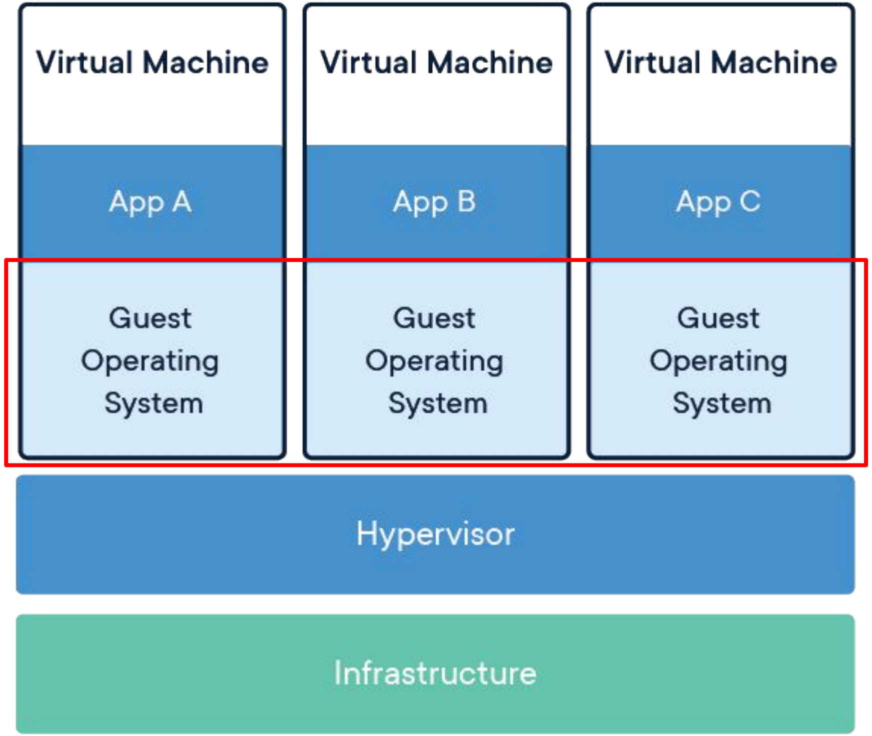
Hypervisor

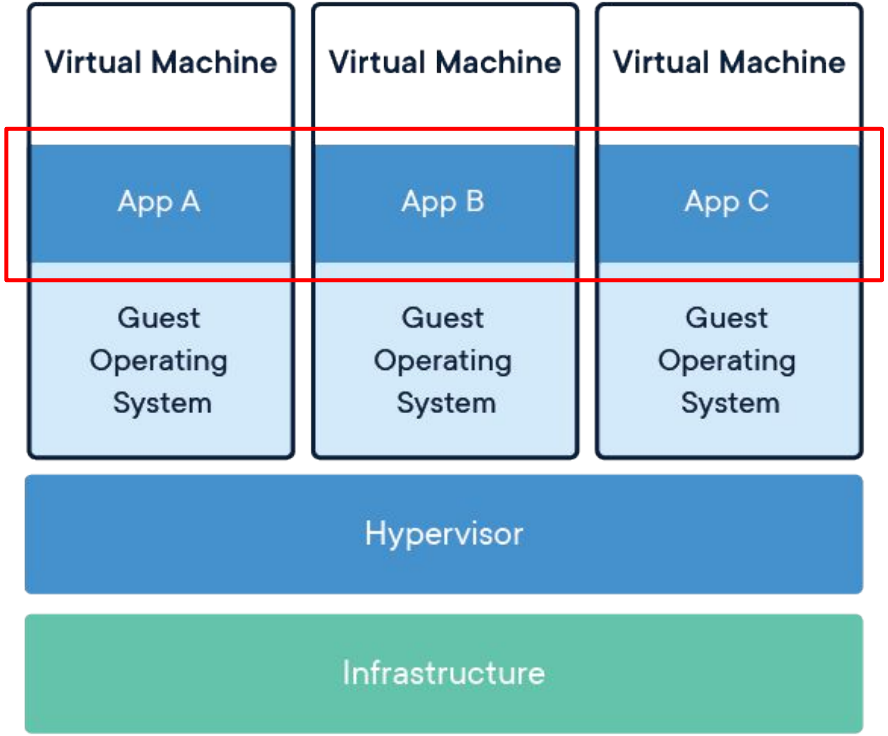
Infrastructure


Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure

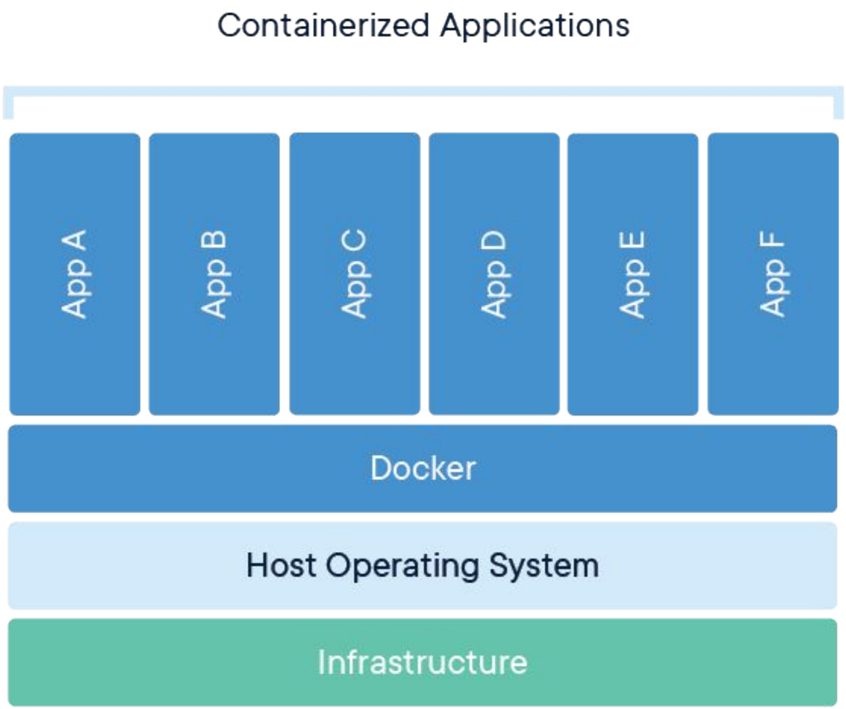


Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure


Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure



Virtual Machine
Virtual Machine
Virtual Machine
App A
App B
App C
Guest Operating System
Guest Operating System
Guest Operating System
Hypervisor
Infrastructure



Containerized Applications

App A

App B

App C

App D

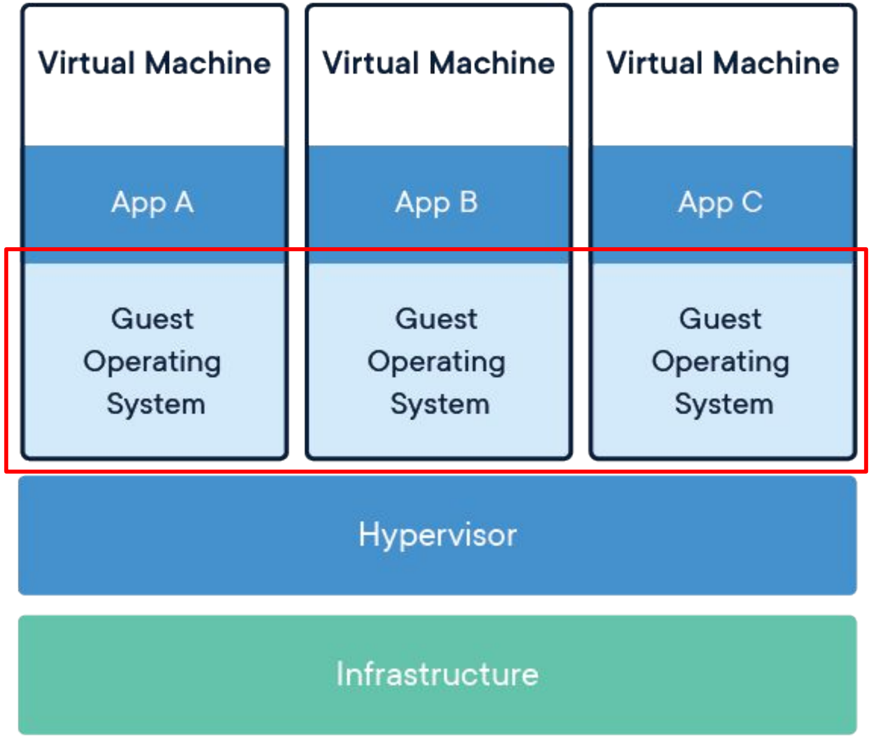
App E

App F

Docker

Host Operating System

Infrastructure



Virtual Machine

Virtual Machine

Virtual Machine

App A

App B

App C

Guest Operating System

Guest Operating System

Guest Operating System

Hypervisor

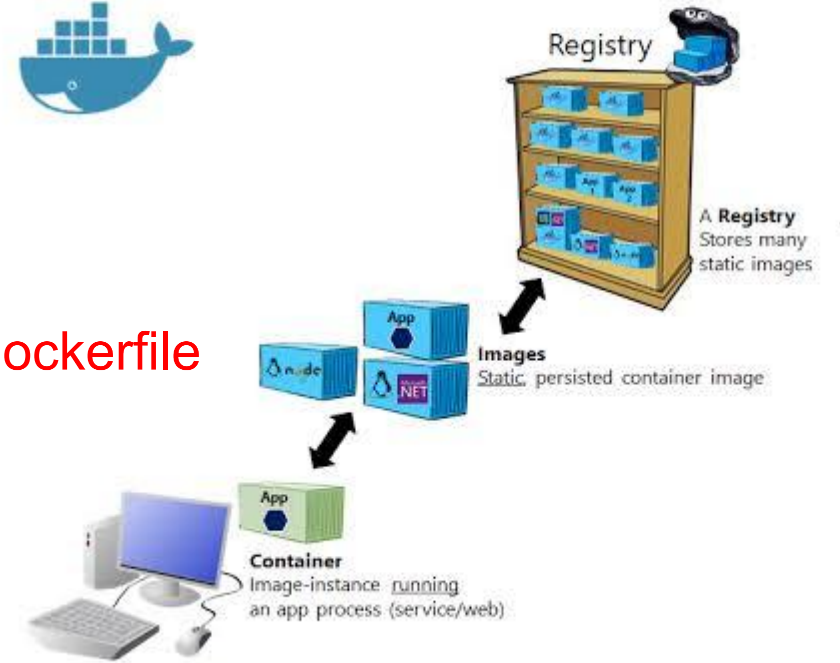
Infrastructure

VM vs Docker

Virtual Machine	Docker
Virtualização a nível de Hardware	Virtualização em nível de SO
Isolamento completo do Host	Compartilhar recursos com o Host
Cada VM tem seu SO	Compartilhamento de recursos de SO
Startup em minutos	Startup in segundos
Utiliza muito recurso	Utiliza pouco recurso
Virtualização de Apps demorada	Necessário apenas configurar Dockerfile

Taxonomia

Dockerfile



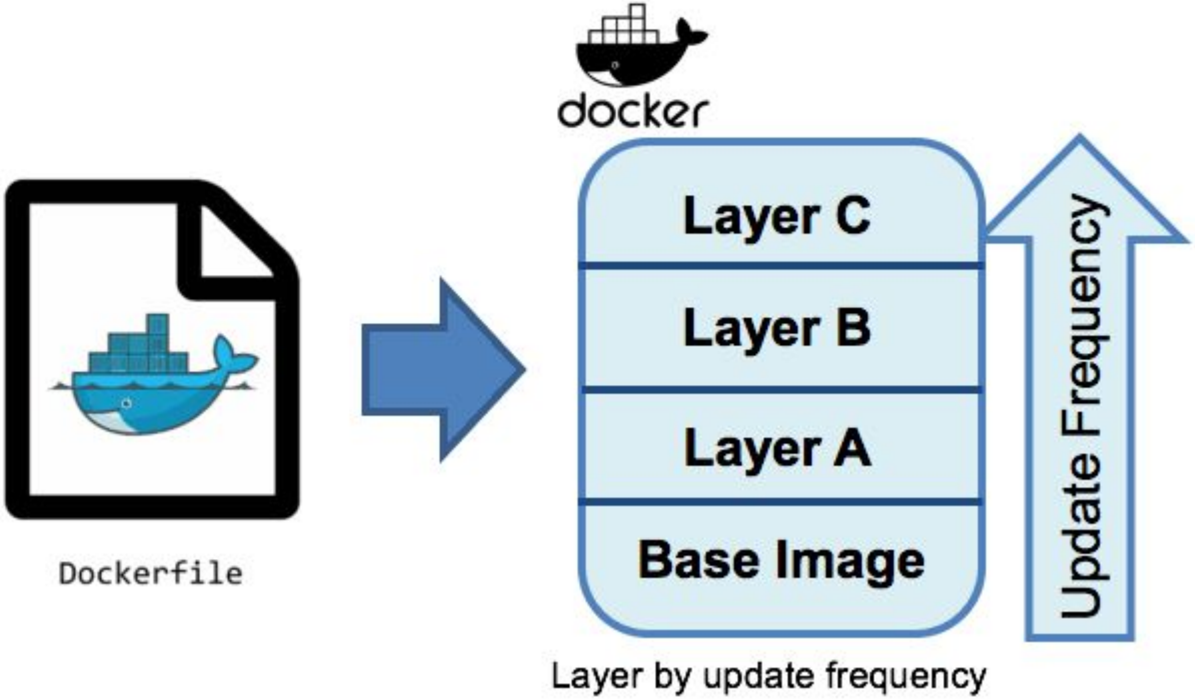
Registry

A **Registry** Stores many static images

Images
Static, persisted container image

Container
Image-instance running an app process (service/web)

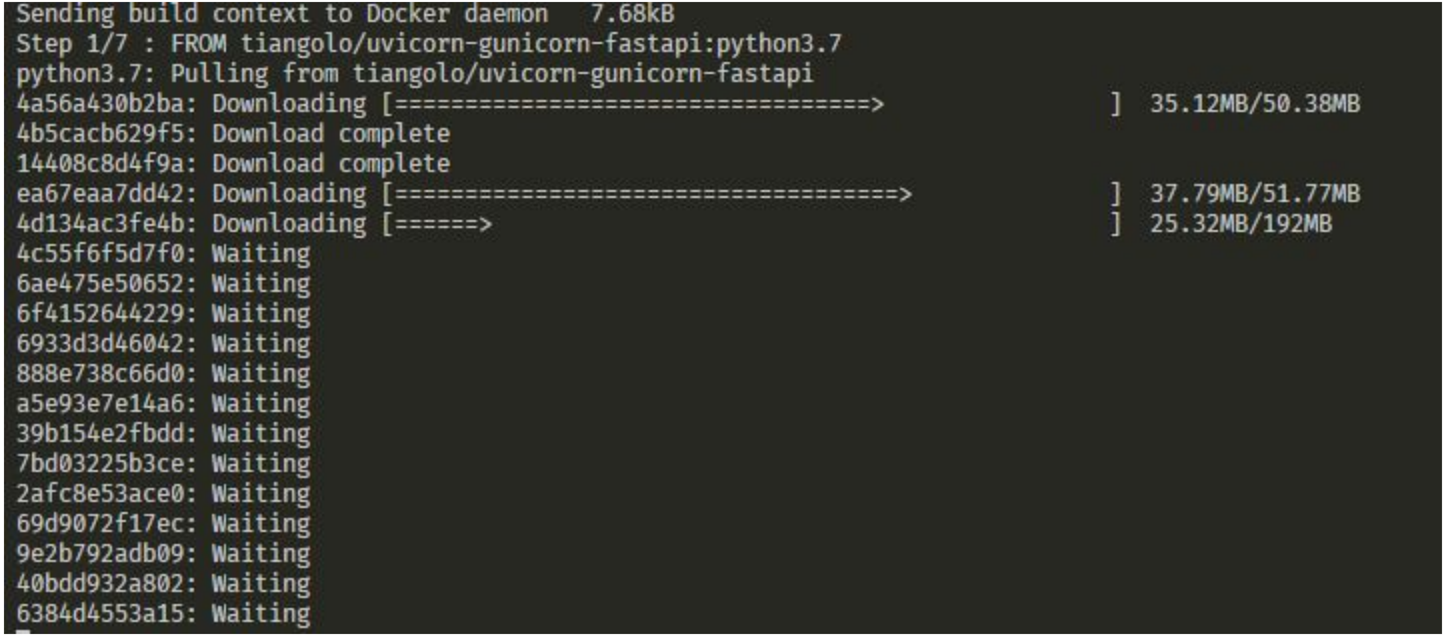
Layers


docker
Dockerfile
Layer C
Layer B
Layer A
Base Image
Update Frequency
Layer by update frequency

Hands-on

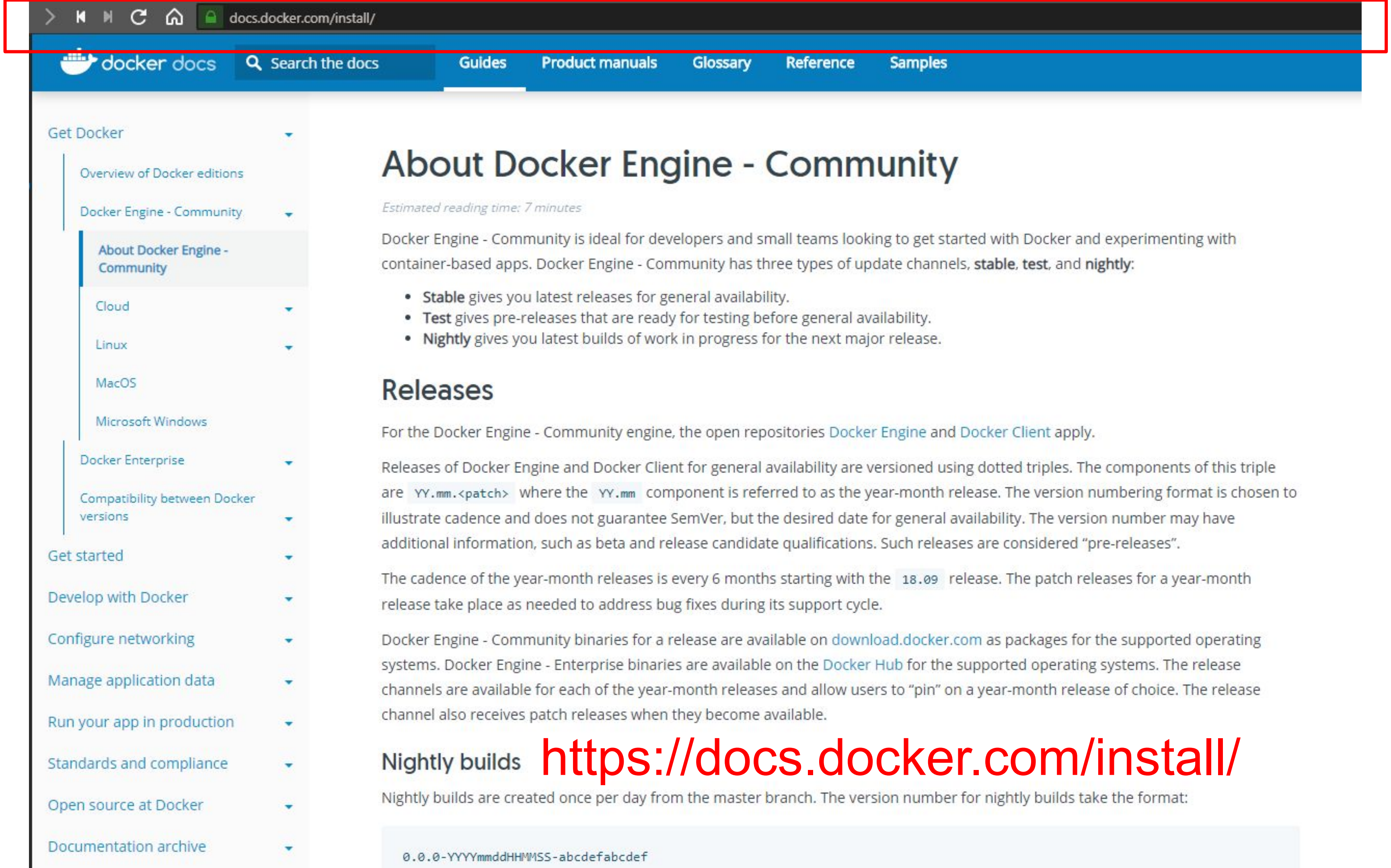
Pré-requisitos

- Docker
- Docker-compose
- Daemon em execução
- Editor
- *Git-bash (Windows)
- Internet



```
Sending build context to Docker daemon  7.68kB
Step 1/7 : FROM tiangolo/uvicorn-gunicorn-fastapi:python3.7
python3.7: Pulling from tiangolo/uvicorn-gunicorn-fastapi
4a56a430b2ba: Downloading [==================================>                ]  35.12MB/50.38MB
4b5cacb629f5: Download complete
14408c8d4f9a: Download complete
ea67eaa7dd42: Downloading [====================================>              ]  37.79MB/51.77MB
4d134ac3fe4b: Downloading [======>                                            ]  25.32MB/192MB
4c55f6f5d7f0: Waiting
6ae475e50652: Waiting
6f4152644229: Waiting
6933d3d46042: Waiting
888e738c66d0: Waiting
a5e93e7e14a6: Waiting
39b154e2fbdd: Waiting
7bd03225b3ce: Waiting
2afc8e53ace0: Waiting
69d9072f17ec: Waiting
9e2b792adb09: Waiting
40bdd932a802: Waiting
6384d4553a15: Waiting
```

Instalação do Docker



About Docker Engine - Community

Estimated reading time: 7 minutes

Docker Engine - Community is ideal for developers and small teams looking to get started with Docker and experimenting with container-based apps. Docker Engine - Community has three types of update channels, **stable**, **test**, and **nightly**:

- **Stable** gives you latest releases for general availability.
- **Test** gives pre-releases that are ready for testing before general availability.
- **Nightly** gives you latest builds of work in progress for the next major release.

Releases

For the Docker Engine - Community engine, the open repositories Docker Engine and Docker Client apply.

Releases of Docker Engine and Docker Client for general availability are versioned using dotted triples. The components of this triple are `YY.mm.<patch>` where the `YY.mm` component is referred to as the year-month release. The version numbering format is chosen to illustrate cadence and does not guarantee SemVer, but the desired date for general availability. The version number may have additional information, such as beta and release candidate qualifications. Such releases are considered “pre-releases”.

The cadence of the year-month releases is every 6 months starting with the `18.09` release. The patch releases for a year-month release take place as needed to address bug fixes during its support cycle.

Docker Engine - Community binaries for a release are available on download.docker.com as packages for the supported operating systems. Docker Engine - Enterprise binaries are available on the Docker Hub for the supported operating systems. The release channels are available for each of the year-month releases and allow users to “pin” on a year-month release of choice. The release channel also receives patch releases when they become available.

Nightly builds

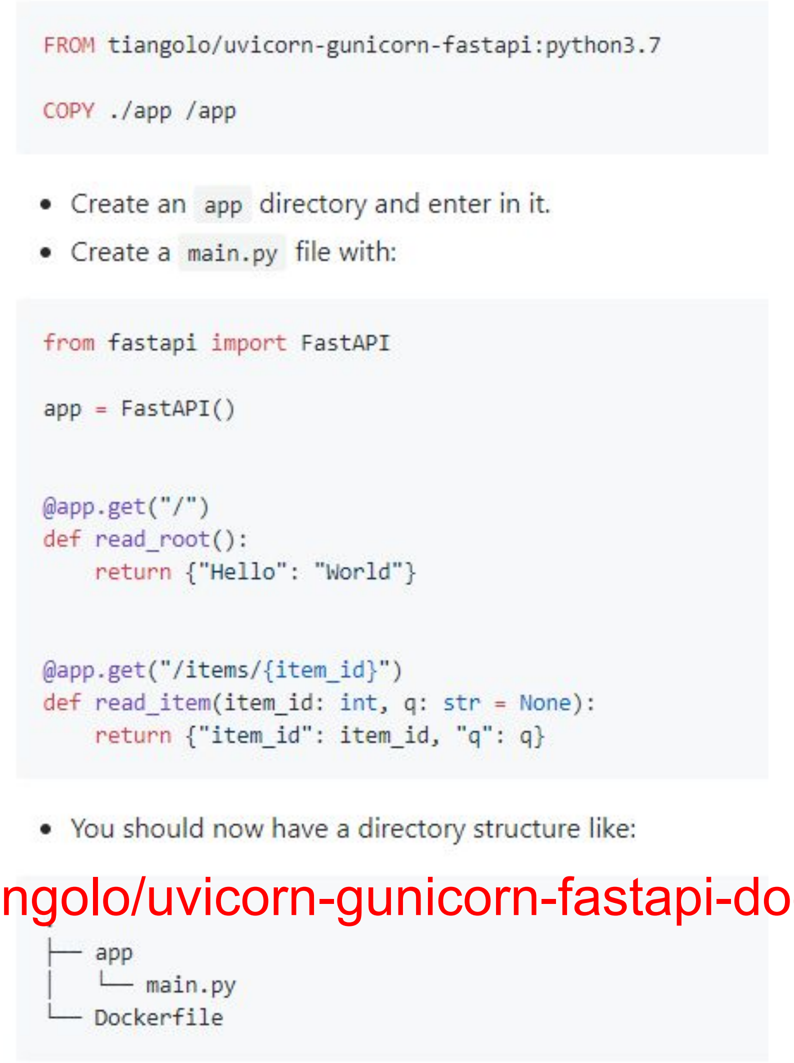
Nightly builds are created once per day from the master branch. The version number for nightly builds take the format:

```
0.0.0-YYYYmmddHHMMSS-abcdefabcdef
```

https://docs.docker.com/install/

Início da API

Base da API



```
FROM tiangolo/uvicorn-gunicorn-fastapi:python3.7

COPY ./app /app
```

- Create an `app` directory and enter in it.
- Create a `main.py` file with:

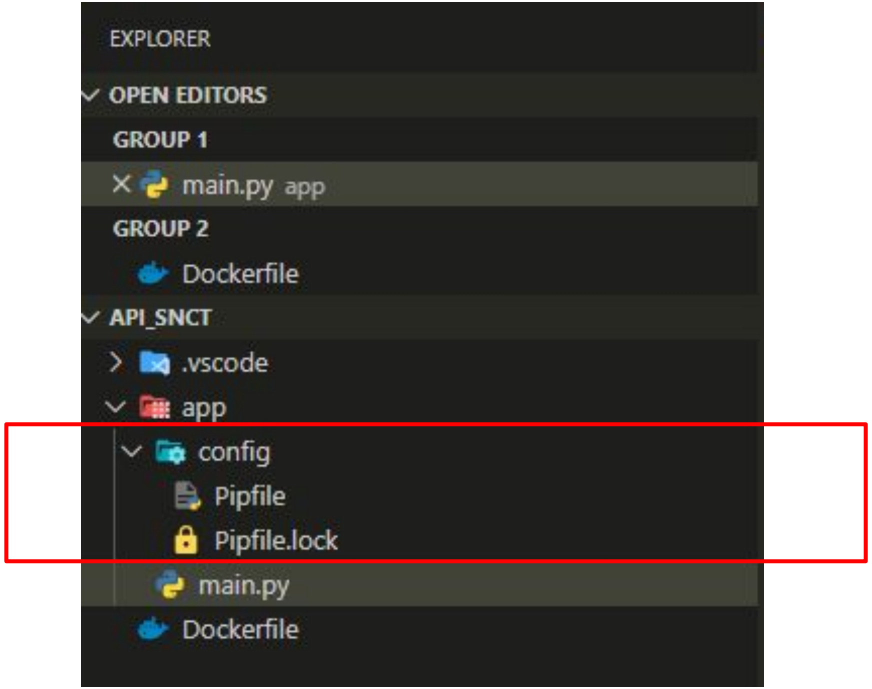
```python
from fastapi import FastAPI

app = FastAPI()


@app.get("/")
def read_root():
    return {"Hello": "World"}


@app.get("/items/{item_id}")
def read_item(item_id: int, q: str = None):
    return {"item_id": item_id, "q": q}
```

- You should now have a directory structure like:

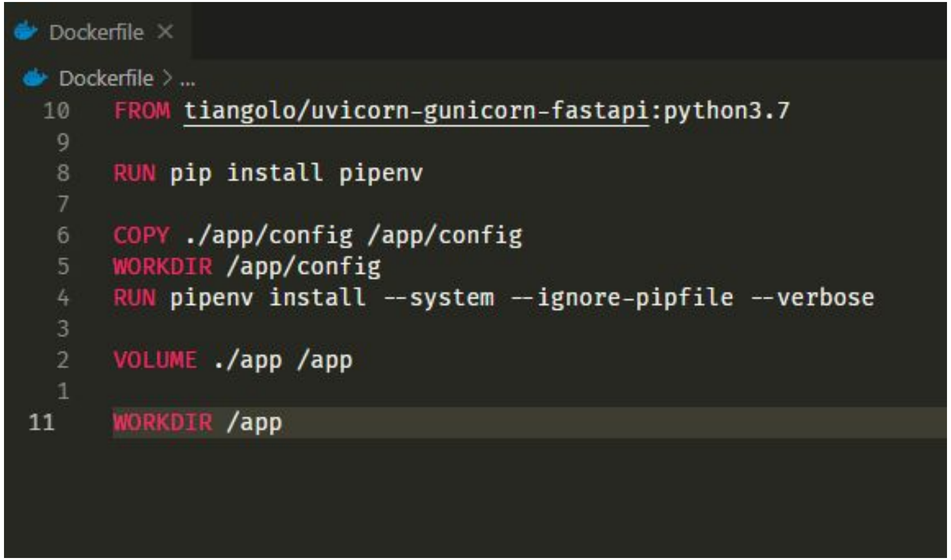
```
├── app
│   └── main.py
└── Dockerfile
```

https://github.com/tiangolo/uvicorn-gunicorn-fastapi-docker#quick-start

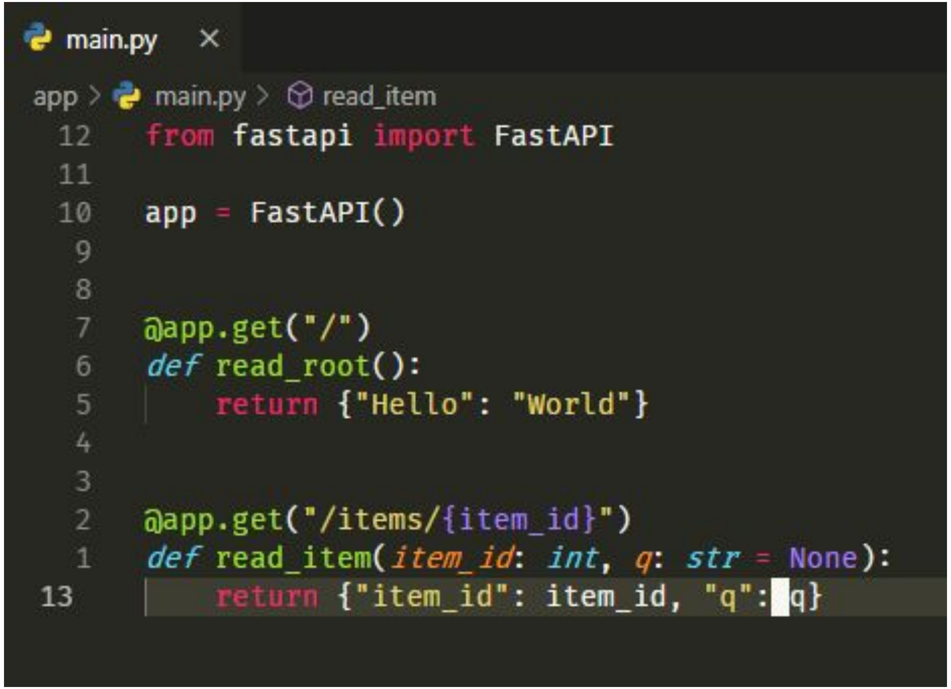
Customização



EXPLORER
OPEN EDITORS
GROUP 1
main.py app
GROUP 2
Dockerfile
API_SNCT
.vscode
app
config
Pipfile
Pipfile.lock
main.py
Dockerfile



```dockerfile
FROM tiangolo/uvicorn-gunicorn-fastapi:python3.7

RUN pip install pipenv

COPY ./app/config /app/config
WORKDIR /app/config
RUN pipenv install --system --ignore-pipfile --verbose

VOLUME ./app /app

WORKDIR /app
```



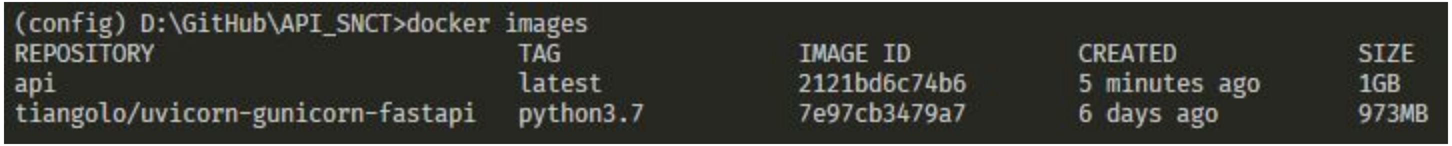
```python
from fastapi import FastAPI

app = FastAPI()


@app.get("/")
def read_root():
    return {"Hello": "World"}


@app.get("/items/{item_id}")
def read_item(item_id: int, q: str = None):
    return {"item_id": item_id, "q": q}
```

Build da Image

```
$> docker build -t api .
```

Verificação de Imagens Locais

```
$> docker images
```



```
(config) D:\GitHub\API_SNCT>docker images
REPOSITORY                      TAG          IMAGE ID       CREATED          SIZE
api                             latest       2121bd6c74b6   5 minutes ago    1GB
tiangolo/uvicorn-gunicorn-fastapi   python3.7    7e97cb3479a7   6 days ago       973MB
```

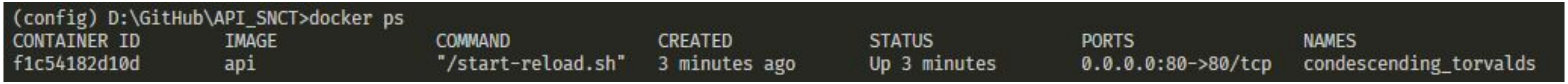
Criação e execução do Container

```
$> docker run --rm -d -p 80:80 -v
`pwd`:/app api /start-reload.sh
```

```
$> docker run --rm -d -p 80:80 -v
`D:\GitHub\API_SNCT`:/app api
/start-reload.sh
```

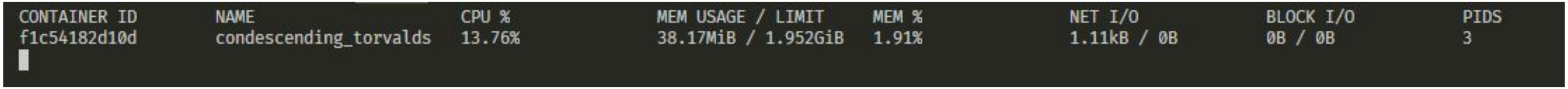
Verificação de containers em execução

```
$> docker ps
```



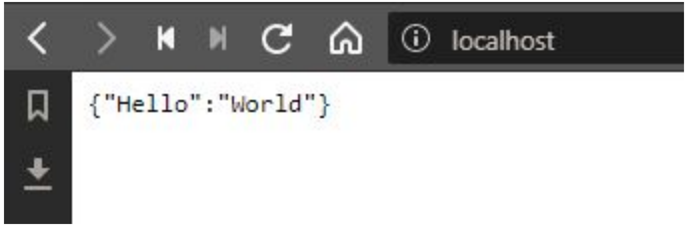
```
(config) D:\GitHub\API_SNCT>docker ps
CONTAINER ID        IMAGE               COMMAND              CREATED             STATUS              PORTS                NAMES
f1c54182d10d        api                 "/start-reload.sh"   3 minutes ago       Up 3 minutes        0.0.0.0:80->80/tcp   condescending_torvalds
```

```
$> docker stats
```

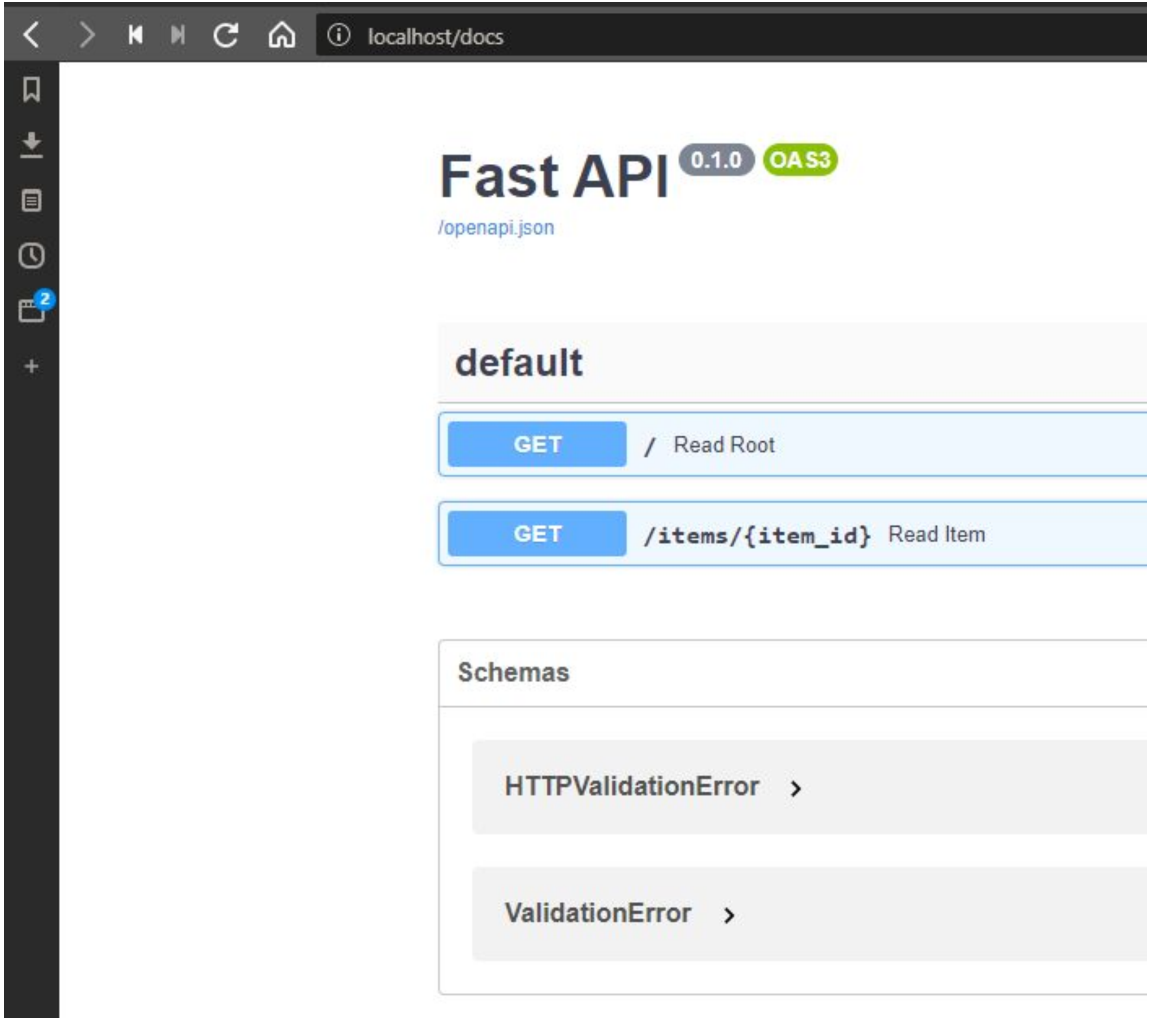


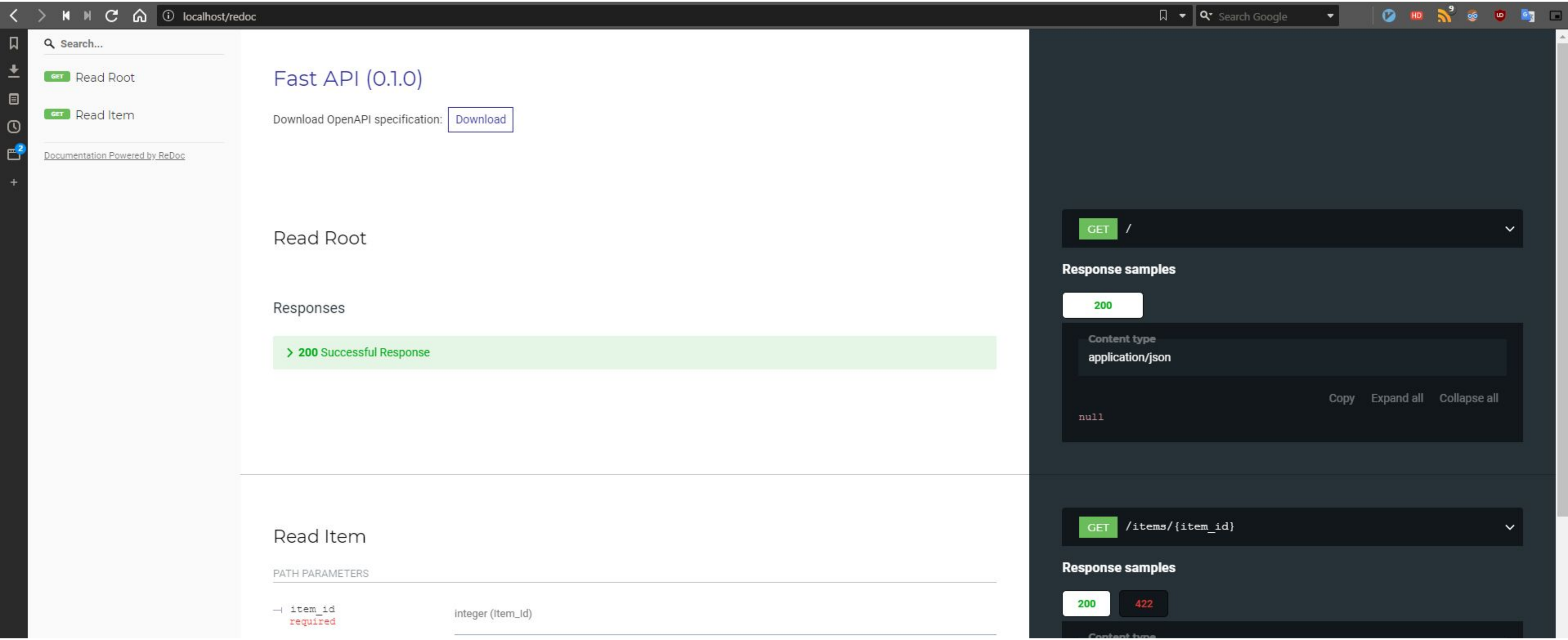
```
CONTAINER ID        NAME                     CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
f1c54182d10d        condescending_torvalds   13.76%              38.17MiB / 1.952GiB   1.91%               1.11kB / 0B         0B / 0B             3
```

Teste


localhost

```
{"Hello":"World"}
```


localhost/docs
Fast API
0.1.0
OAS3
/openapi.json
default
GET
/
Read Root
GET
/items/{item_id}
Read Item
Schemas
HTTPValidationError
ValidationError
2



Fast API (0.1.0)

Download OpenAPI specification: Download

Read Root

Responses

> 200 Successful Response

GET /

Response samples

200

Content type
application/json

Copy Expand all Collapse all

```
null
```

Read Item

PATH PARAMETERS

item_id required	integer (Item_Id)

GET /items/{item_id}

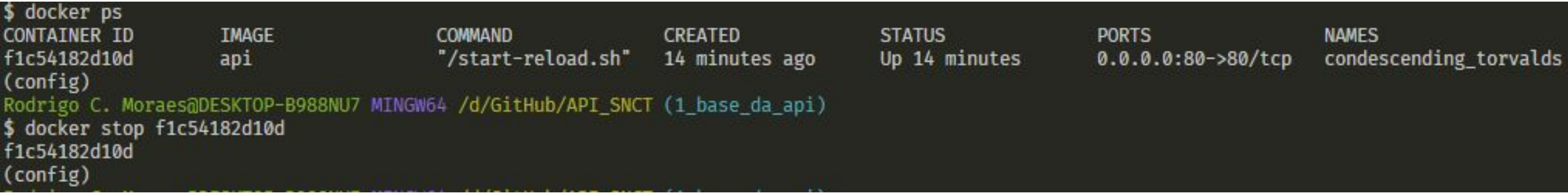
Response samples

200 422

Content type

Parar container em execução

```
$> docker stop CONTAINER_ID
```



```
$ docker ps
CONTAINER ID    IMAGE         COMMAND             CREATED           STATUS          PORTS                 NAMES
f1c54182d10d    api           "/start-reload.sh"  14 minutes ago    Up 14 minutes   0.0.0.0:80->80/tcp    condescending_torvalds
(config)
Rodrigo C. Moraes@DESKTOP-B988NU7 MINGW64 /d/GitHub/API_SNCT (1_base_da_api)
$ docker stop f1c54182d10d
f1c54182d10d
(config)
```

Dúvidas?

Introdução a Docker

By Rodrigo C. Moraes

https://github.com/rodrigocmoraes

rdcmdev@gmail.com

kaggle https://www.kaggle.com/rdcmdev

@rodrigocmoraes