ExercÃcios prÃ¡ticos:

1) ContÃªiner java:

```
docker pull hub.estaleiro.serpro/pipeline/openjdk:8-alpine
docker images
docker ps -a
docker run --name teste1 -ti hub.estaleiro.serpro/pipeline/openjdk:8-alpine
```

2) ContÃªiner tomcat:

```
mkdir /tmp/teste2
cd /tmp/teste2
echo "<head></head><body>bla bla bla</body>" > index.html
docker run -d --name meuapache1 -p 80:80 -v ${PWD}:/usr/local/apache2/htdocs/ httpd:2.4
```

DEPOIS:

```
docker rm -f meuapache1
docker run -d --name meuapache1 -v ${PWD}:/usr/local/apache2/htdocs/ httpd:2.4
docker exec -it meuapache1 apt-get update
docker exec -it meuapache1 apt-get install curl
docker exec -it meuapache1 curl http://localhost

docker commit meuapache1 vamoverissoai
docker tag vamoverissoai:latest eduardomacielmaia/vamoverissoai
docker login
docker push eduardomacielmaia/vamoverissoai
```

3.1) Dockerfile

```
# Use an official Python runtime as a parent image
FROM python:2.7-slim

# Set the working directory to /app
WORKDIR /app

# Copy the current directory contents into the container at /app
COPY . /app

# Install any needed packages specified in requirements.txt
RUN pip install --trusted-host pypi.python.org -r requirements.txt

# Make port 80 available to the world outside this container
EXPOSE 80
```

```
# Define environment variable
ENV NAME World

# Run app.py when the container launches
CMD ["python", "app.py"]
```

3.2) requirements.txt

```
Flask
Redis
```

3.3) app.py

```
from flask import Flask
from redis import Redis, RedisError
import os
import socket

# Connect to Redis
redis = Redis(host="redis", db=0, socket_connect_timeout=2, socket_timeout=2)

app = Flask(__name__)

@app.route("/")
def hello():
    try:
        visits = redis.incr("counter")
    except RedisError:
        visits = "<i>cannot connect to Redis, counter disabled</i>"

    html = "<h3>Hello {name}!</h3>" \
           "<b>Hostname:</b> {hostname}<br/>" \
           "<b>Visits:</b> {visits}"
    return html.format(name=os.getenv("NAME", "world"), hostname=socket.gethostname(),
visits=visits)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80)
```

3.4)

```
docker build --tag=imagempythonhello .

docker run -p 4000:80 imagempythonhello
```

4.1) Docker Compose:

```
version: '2'

services:

  wildfly:
    image: jboss/wildfly:10.1.0.Final
    container_name: wildfly
    ports:
      - "9000:8080"
    volumes:
      - /tmp/teste3/ROOT.war:/opt/jboss/wildfly/standalone/deployments/ROOT.war

  postgres:
    image: postgres:9.6-alpine
    container_name: postgres
    environment:
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=postgres
      - PGDATA=/var/lib/postgresql/data/pgdata
    ports:
      - '5432:5432'
    volumes:
      - /tmp/teste3/banco:/var/lib/postgresql/data/pgdata
```

4.2) Comandos:

```
cd /tmp/teste3
curl -kO
https://git.serpro/Estaleiro/plataformas/raw/master/wildfly/test/test-app/deployments/ROOT.war
docker-compose up
```

Numa nova aba:

```
ps aux | grep jboss
```

LINKS:

https://dedat.gitpages.serpro/deat5/deat5-gitpages/ambiente-desenvolvimento-com-docker/

https://dedat.gitpages.serpro/deat5/deat5-gitpages/introducao-docker/

https://docs.docker.com/get-started/part2/