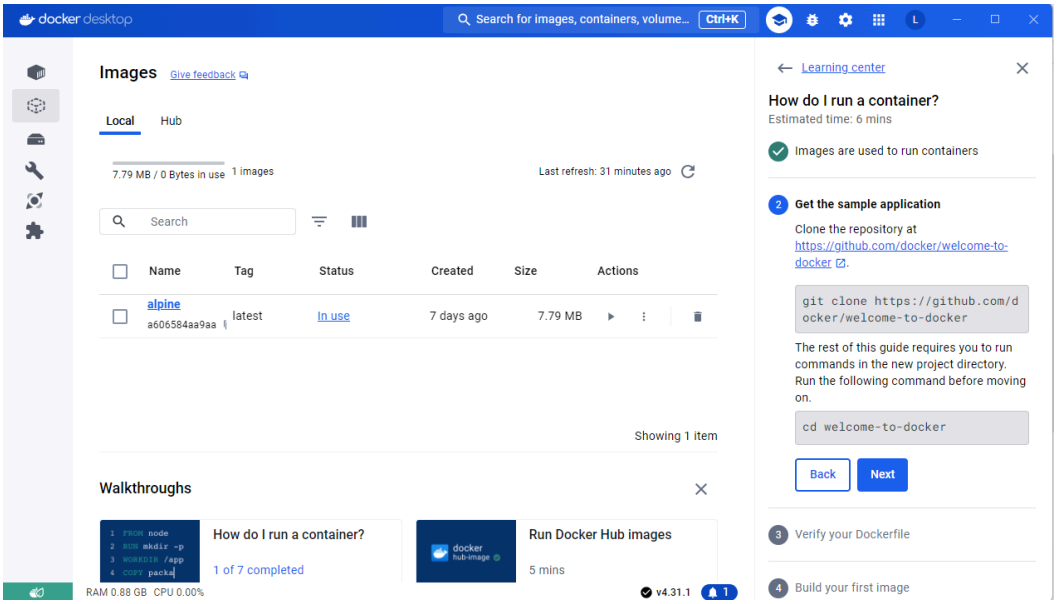
Luiz Felipe Nagatani Rabello

Relatório LAMIA - Módulo 16

Seção 1

A seção 1 é composta por um breve vídeo ensinando a instalação e os básicos da utilização do docker no windows e no Linux, sendo necessário um aplicativo chamado Docker Desktop para a utilização do docker no windows.



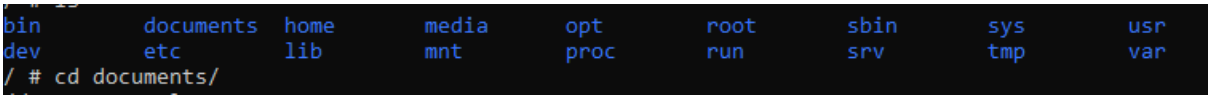
Seção 2 e 3

O instrutor aborda comandos para a command line interface, os primeiros apresentados servem para criar container, remover eles, e remover a imagem do container sem que o container seja deletado.

Um erro que apareceu durante a execução desta seção foi o Unauthorized: username and password incorrect. Que apareceu mesmo eu estando logado.

Foi necessário utilizar docker logout e docker login para fazer o login pelo cmd para poder dar pull no alpine e implementar os comandos apresentados na seção.

Em seguida é apresentado o mapeamento de volumes, podendo relacionar os diretórios da máquina host com os diretórios do container a fim de salvar um estado de um container



Para efetuar operações com portas para que outras máquinas possam acessar o mesmo container, foi necessário utilizar o nginx.

```
C:\Users\shink>docker container exec -it nginx sh
# ls
bin   dev                docker-entrypoint.sh  home  lib64  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint.d  etc                  lib   media  opt  root  sbin  sys  usr
# -
```

```
    "Gateway": "172.17.0.1",
    "IPAddress": "172.17.0.2",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "DriverOpts": null,
    "DNSNames": null
```

Seção 4

A seção 4 aborda criação e exportação de imagens a partir de um container

```
C:\Users\shink>docker container run -it --rm nginx--allumy-img sh
/ # ls]
sh: ls]: not found
/ # ls
allumy  dev     home    media   opt     root    sbin    sys     usr
bin     etc     lib     mnt     proc    run     srv     tmp     var
/ # cd allumy/
/allumy # ls
docker
/allumy # cat docker
TESTE CONTAINER PARA IMAGM
/allumy #
```

É possível salvar uma imagem em .tar para facilitar a exportação para outros usuários utilizando docker image save -o [arquivo] -i [imagem].

```
C:\Users\shink>docker image save -o exportnginx.tar nginx--allumy-img
```

Para consultar o histórico de uma imagem, deve se utilizar o comando docker image history [nome da imagem]

```
C:\Users\shink>docker image history nginx--allumy-img
IMAGE          CREATED         CREATED BY                                      SIZE      COMMENT
446e71e88f50   11 minutes ago  /bin/sh                                         129B
a606584aa9aa   13 days ago     /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      13 days ago     /bin/sh -c #(nop) ADD file:33ebe56b967747a97…   7.8MB
```

Para finalizar, alguns outros comando são apresentados, como:

Prune: Deleta imagens que não estão associadas a nenhum container.

Pull: Baixa imagens.
Push:Disponibiliza uma imagem no seu repositório do docker.

Seção 5

A seção 5 se trata da criação de imagens através de DockerFiles(vários comandos contidos em um arquivo de texto para criar uma imagem docker)

```
C:\Users\shink>docker image build -t teste .
[+] Building 4.1s (1/1) FINISHED
```

Seção 6

O conceito de volumes para salvar estados de containers é aprofundado na seção 6, utilizando docker volume create[nome] para criação do mesmo e
docker run -v [volume]:[caminho do volume no container] [imagem] para mapear um container no volume

Seção 7

Na seção 7 são abordados alguns dos comandos para administração do ambiente docker utilizando docker system df, para mostrar uma tabela com o número de containers, imagens e volumes, juntamente com o espaço ocupado pelos mesmos.

```
C:\Users\shink>docker system df
TYPE            TOTAL     ACTIVE    SIZE      RECLAIMABLE
Images          4         3         203.2MB   7.799MB (3%)
Containers      3         0         1.253kB   1.253kB (100%)
Local Volumes   0         0         0B        0B
Build Cache     1         0         0B        0B
```

Seção 8

A seção 8 consiste em uma curta revisão do que foi passado em todos os módulos, juntamente de um exercício para reforçar os comandos do docker CLI, nele foi requisitado que se criasse uma imagem, e que ela fosse salvada, removida, importada e em seguida carregada para um container, para demonstrar a capacidade do comando save de conter informações contidas na imagem em um arquivo .tar.

```
C:\Users\shink\nginx>docker image save exercicio -o exercicio

C:\Users\shink\nginx>docker container rm -f exercicio
exercicio

C:\Users\shink\nginx>docker image rm exercicio
Untagged: exercicio:latest
Deleted: sha256:edc82d947f59e3d671158bdea8a9ff44c3e1bb4527ddef982df224ca8e28a670
Deleted: sha256:449df47b580c6090ab0258d3337ec70dd3d4912a9adee5c519b27c61ca1b56db

C:\Users\shink\nginx>docker image ls
REPOSITORY          TAG       IMAGE ID       CREATED         SIZE
importado           latest    9e34bd975468   21 hours ago    7.8MB
nginx--allumy-img   latest    446e71e88f50   22 hours ago    7.8MB
nginx               latest    fffffc90d343   13 days ago     188MB
alpine              latest    a606584aa9aa   13 days ago     7.8MB
ubuntu              latest    35a88802559d   3 weeks ago     78.1MB

C:\Users\shink\nginx>docker image load -i exercicio
1e1638b24ad9: Loading layer [==================================================>]  46.29MB/46.29MB
Loaded image: exercicio:latest

C:\Users\shink\nginx>docker image ls
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
exercicio           latest    edc82d947f59   7 minutes ago    124MB
```