

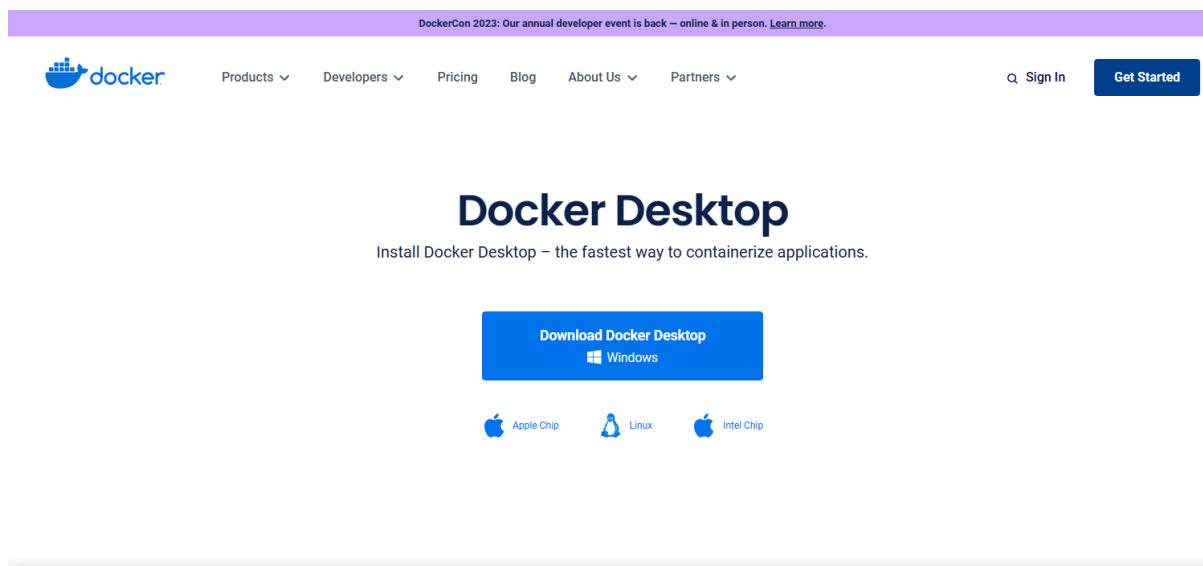
## CURSO SUPERIOR EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS UNIDADE CURRICULAR DE SISTEMAS DISTRIBUÍDOS

Diórgenes Orlando Paczkowski dos Santos Fagundes e Thales Paulo

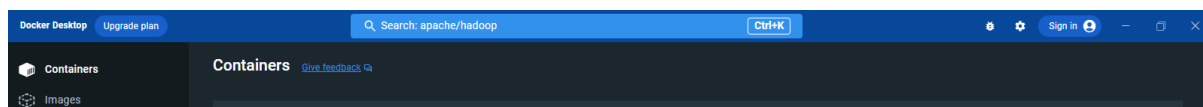
### Criando e configurando um cluster distribuído com Docker e Apache Hadoop

#### Preparação do ambiente

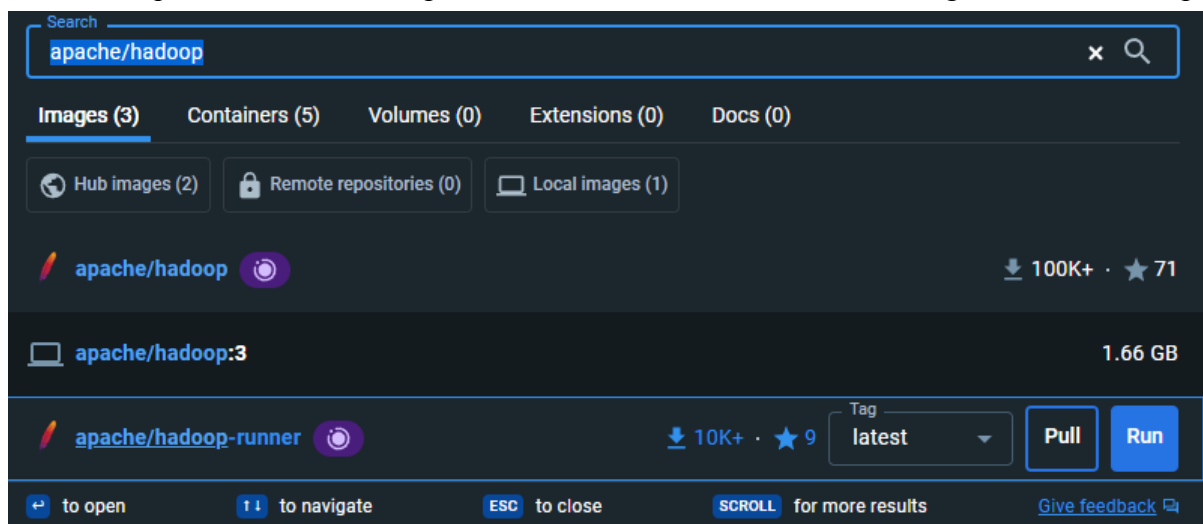
1 - Fazer o download do Docker Desktop em: [Download Docker Desktop | Docker](#) e escolher a versão de acordo com seu Sistema Operacional..



2 - Na caixa a seguir digite apache/hadoop.

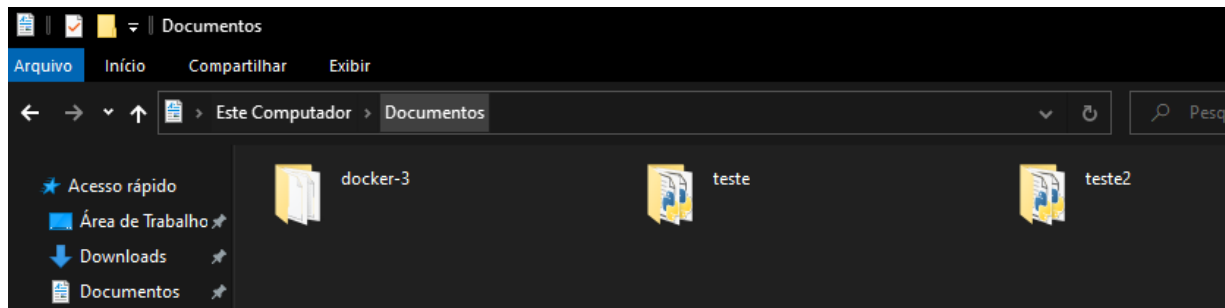


3 - Clique em “Pull” para fazer o download da imagem do hadoop.

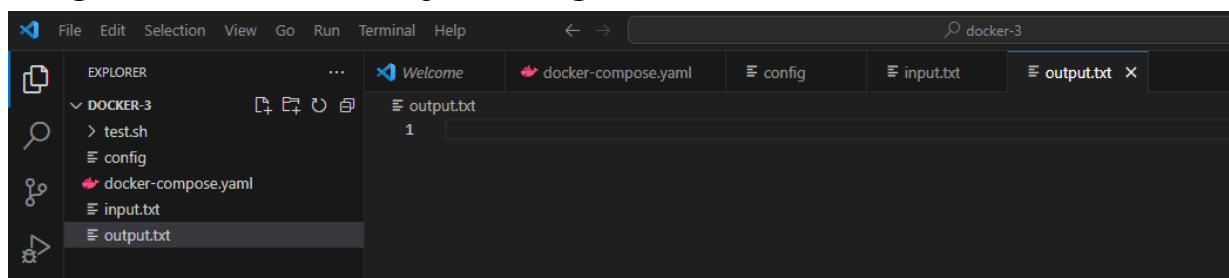


Após a instalação, fazer a configuração dos arquivos necessários.

#### 4 - Crie uma pasta chamada docker-3



5 - Clique com o botão direito na pasta Docker-3 e execute com o Visual Code e crie os seguintes arquivos: **docker-compose.yaml**, **input.txt**, **output.txt** e um arquivo chamado **config** sem nenhuma extensão, apenas **config**.



6 - Dentro de **docker-compose.yaml**, coloque o seguinte código:

```
version: "2"
services:
  namenode:
    image: apache/hadoop:3
    hostname: namenode
    command: ["hdfs", "namenode"]
    ports:
      - 9870:9870
    env_file:
      - ./config
    environment:
      ENSURE_NAMENODE_DIR: "/tmp/hadoop-root/dfs/name"
  datanode1:
    image: apache/hadoop:3
    command: ["hdfs", "datanode"]
    env_file:
      - ./config
  datanode2:
    image: apache/hadoop:3
    command: ["hdfs", "datanode"]
    env_file:
      - ./config
  resourcemanager:
    image: apache/hadoop:3
    hostname: resourcemanager
    command: ["yarn", "resourcemanager"]
    ports:
```

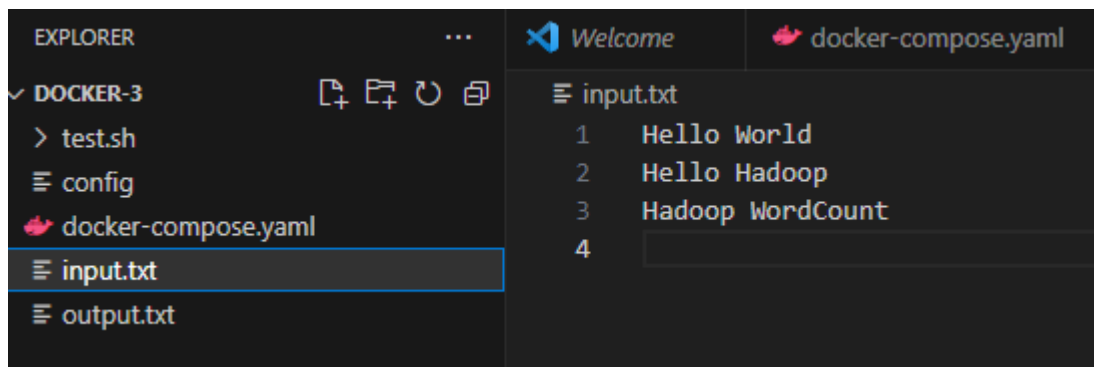


```
- 8088:8088
env_file:
- ./config
volumes:
- ./test.sh:/opt/test.sh
nodemanager:
image: apache/hadoop:3
command: ["yarn", "nodemanager"]
env_file:
- ./config
```

7 - Dentro de **config**, coloque o seguinte código:

```
CORE-SITE.XML_fs.default.name=hdfs://namenode
CORE-SITE.XML_fs.defaultFS=hdfs://namenode
HDFS-SITE.XML_dfs.namenode.rpc-address=namenode:8020
HDFS-SITE.XML_dfs.replication=1
MAPRED-SITE.XML_mapreduce.framework.name=yarn
MAPRED-SITE.XML_yarn.app.mapreduce.am.env=HADOOP_MAPRED_HOME=$HADOOP_HOME
MAPRED-SITE.XML_mapreduce.map.env=HADOOP_MAPRED_HOME=$HADOOP_HOME
MAPRED-SITE.XML_mapreduce.reduce.env=HADOOP_MAPRED_HOME=$HADOOP_HOME
YARN-SITE.XML_yarn.resourcemanager.hostname=resourcemanager
YARN-SITE.XML_yarn.nodemanager.pmem-check-enabled=false
YARN-SITE.XML_yarn.nodemanager.delete.debug-delay-sec=600
YARN-SITE.XML_yarn.nodemanager.vmem-check-enabled=false
YARN-SITE.XML_yarn.nodemanager.aux-services=mapreduce_shuffle
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.maximum-applications=10000
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.maximum-am-resource-percent=0.1
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.resource-calculator=org.apache.hadoop
p.yarn.util.resource.DefaultResourceCalculator
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.queues=default
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.capacity=100
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.user-limit-factor=1
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.maximum-capacity=100
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.state=RUNNING
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.acl_submit_applications
=*
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.root.default.acl_administer_queue=*
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.node-locality-delay=40
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.queue-mappings=
CAPACITY-SCHEDULER.XML_yarn.scheduler.capacity.queue-mappings-override.enable=false
```

8 - Crie os arquivos de entrada “input.txt” e coloque algum texto, com aqui no exemplo e saída “output.txt” que ficará vazio. Depois esses arquivos servirão para executar o exemplo no cluster.



Nesse exemplo é usado o texto:

Hello World  
Hello Hadoop  
Hadoop WordCount

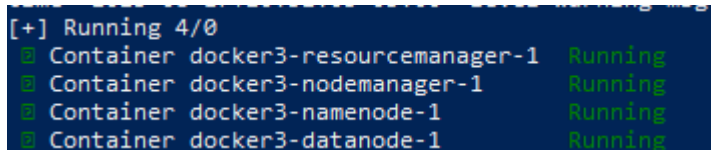
Se quiser, é possível alterar esse arquivo para o que desejar.

### Execução do cluster

1 - Para começar a fazer a configuração dos containers do cluster abra o terminal na pasta do docker-3 em `cd <diretório_da_pasta_docker-3>` que foi criada inicialmente e execute o seguinte comando

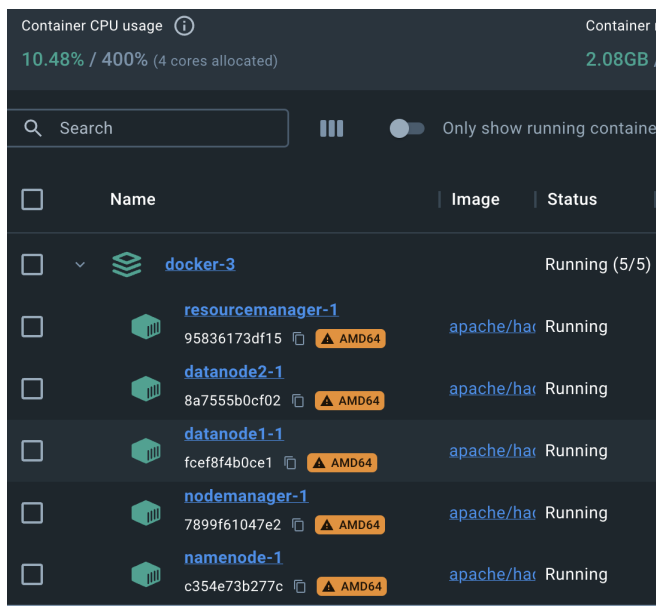
2 - Iniciando os containers : Executar o comando `docker-compose up -d` para iniciar o cluster e criar os nós :

deverá aparecer a mensagem :

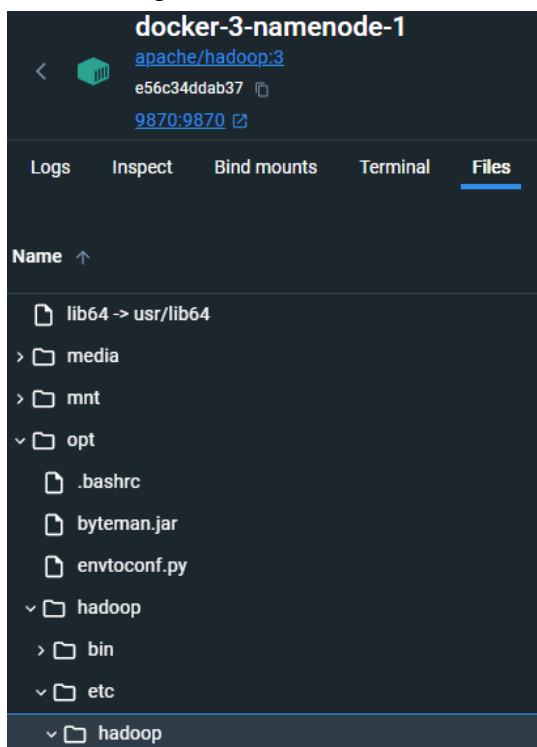


```
[+] Running 4/0
  Container docker3-resourcemanager-1   Running
  Container docker3-nodemanager-1       Running
  Container docker3-namenode-1          Running
  Container docker3-datanode-1          Running
```

3 - Entre no docker e perceba que os containers estão dessa forma:

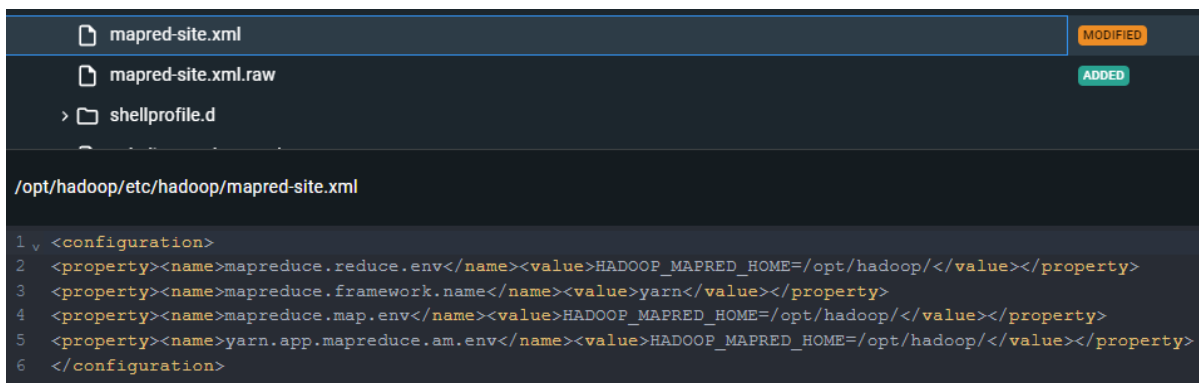


4 - Precisará alterar o arquivo mapred-site.xml , para isso clique em um deles e clique em “Files” e ache o respectivo diretório /opt/hadoop/etc/hadoop/



5 - Ache dentro desse diretório mapred-site.xml e coloque a configuração para todos os containers.

```
<configuration>
<property><name>mapreduce.reduce.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
<property><name>mapreduce.framework.name</name><value>yarn</value></property>
<property><name>mapreduce.map.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
<property><name>yarn.app.mapreduce.am.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
</configuration>
```



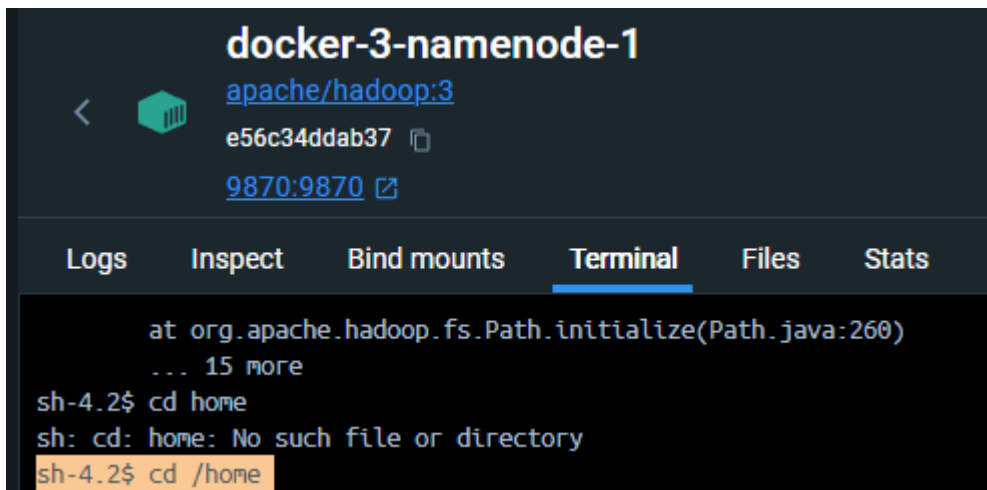
The screenshot shows a file explorer interface. At the top, there are three files: 'mapred-site.xml' (marked as MODIFIED), 'mapred-site.xml.raw' (marked as ADDED), and 'shellprofile.d'. Below the file list, the content of 'mapred-site.xml' is displayed. The content is an XML configuration snippet for Hadoop MapReduce, setting environment variables for the reduce, map, and application master processes to use the Hadoop MapReduce home directory at /opt/hadoop.

```
1 <configuration>
2 <property><name>mapreduce.reduce.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
3 <property><name>mapreduce.framework.name</name><value>yarn</value></property>
4 <property><name>mapreduce.map.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
5 <property><name>yarn.app.mapreduce.am.env</name><value>HADOOP_MAPRED_HOME=/opt/hadoop</value></property>
6 </configuration>
```

Atenção faça isso para os demais containers.

## Execução do exemplo

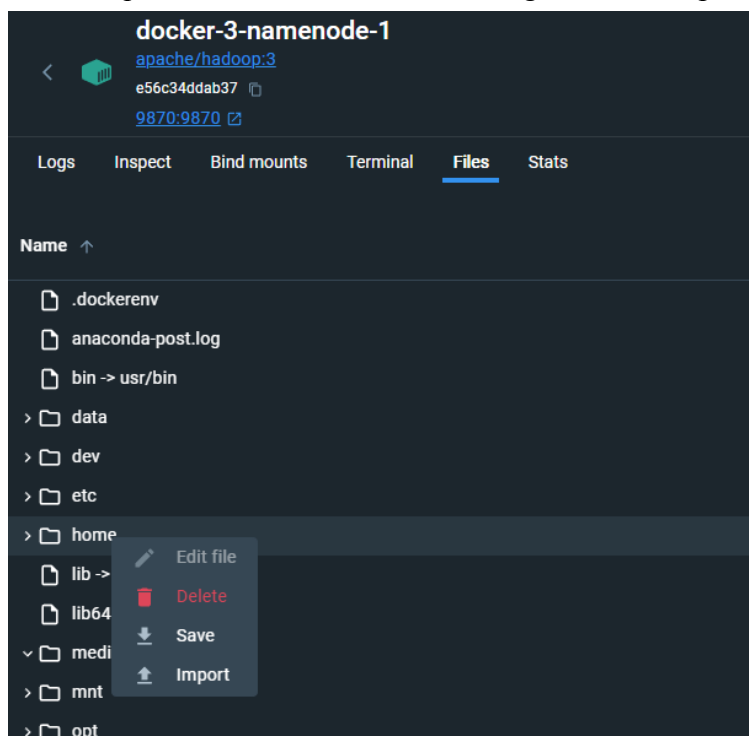
1 - No terminal do container **docker-3-namenode** no docker execute `cd /home` para entrar na pasta home do container, onde será colocada a pasta docker-3



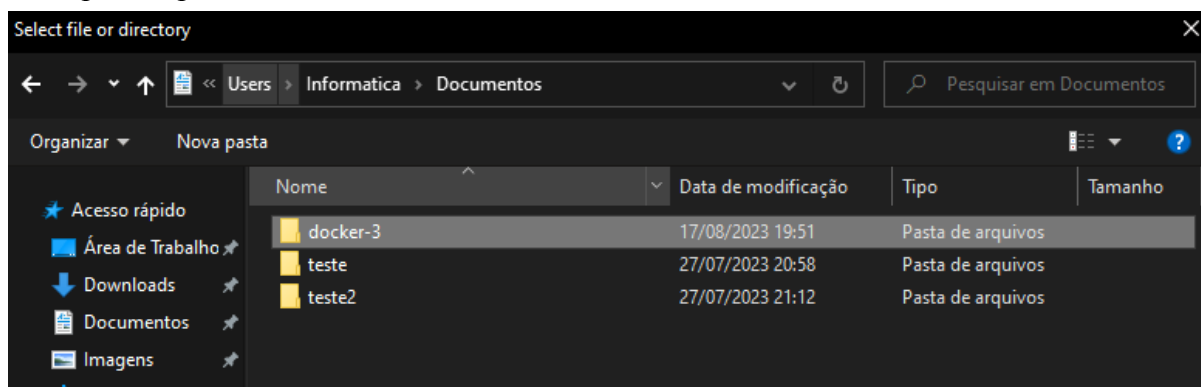
The screenshot shows the Docker Desktop interface for a container named 'docker-3-namenode-1'. The container is based on 'apache/hadoop:3' and has the ID 'e56c34ddab37'. The ports '9870:9870' are mapped. The 'Terminal' tab is active, showing the command prompt 'sh-4.2\$'. The user has entered 'cd /home', which resulted in an error message: 'sh: cd: /home: No such file or directory'. The user is now entering 'cd /home' again.

```
at org.apache.hadoop.fs.Path.initialize(Path.java:260)
... 15 more
sh-4.2$ cd /home
sh: cd: /home: No such file or directory
sh-4.2$ cd /home
```

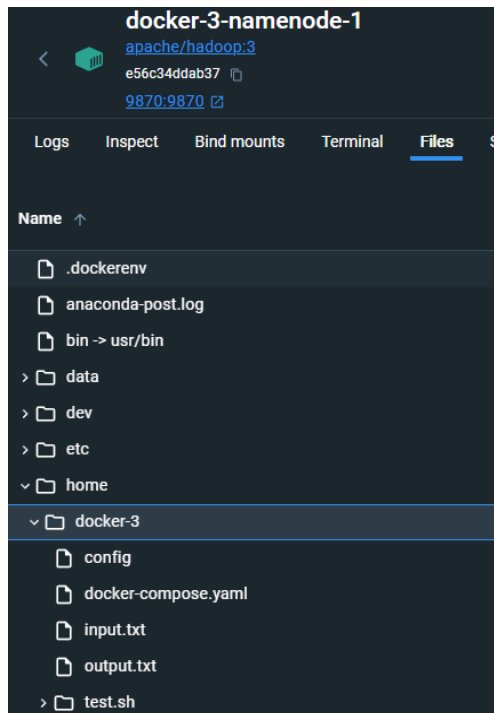
2 - Para executar um programa exemplo que irá ler a quantidade de palavras de um arquivo de texto importe a pasta **docker-3** de um container. No exemplo será usado o **docker-3-namenode-1**. Clique em “Files” dentro do container selecionado, encontre a pasta home clique com o botão direito e em seguida em “Import”.



3 - Importe a pasta **docker-3**



4 - Nessa pasta home aparecerá a pasta docker-3 com seus respectivos arquivos.



5 - Para executar o arquivo `input.txt` que terá as palavras que serão contadas é necessário colocar os seguintes comandos.

Volte no terminal e coloque os seguintes comandos `ls` para verificar se a pasta docker-3 está presente no diretório `/home`, em seguida entre em `cd docker-3` e `ls` para verificar se os arquivos estão presentes

```
sh-4.2$ ls
docker-3
sh-4.2$ cd docker-3
sh-4.2$ ls
config  docker-compose.yaml  input.txt  output.txt  test.sh
```

6 - Após verificar os arquivos presentes, digite: `hdfs dfs -put input.txt`

```
sh-4.2$ hdfs dfs -put input.txt
```

Nota: Em caso de erro na execução desse comando

```
sh-4.2$ hdfs dfs -put input.txt
put: `.`: No such file or directory: `hdfs://namenode/user/hadoop'
```

Execute `hdfs dfs -mkdir -p /user/hadoop` e volte a executar o comando anterior.

```
sh-4.2$ hdfs dfs -mkdir -p /user/hadoop
sh-4.2$ hdfs dfs -put input.txt
sh-4.2$
```



## 7 - Agora digite:

```
yarn jar /opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar wordcount input.txt output.txt
```

```
sh-4.2$ yarn jar /opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar wordcount input.txt output.txt
2023-08-17 23:13:37 INFO DefaultNoHARMFalloverProxyProvider:64 - Connecting to ResourceManager at resourcemanager/172.18.0.3:8032
2023-08-17 23:13:37 INFO JobResourceUploader:907 - Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hadoop/.staging/job_1692310718604_
2023-08-17 23:13:37 INFO FileInputFormat:300 - Total input files to process : 1
2023-08-17 23:13:37 INFO JobSubmitter:202 - number of splits:1
2023-08-17 23:13:37 INFO JobSubmitter:298 - Submitting tokens for job: job_1692310718604_0010
2023-08-17 23:13:37 INFO JobSubmitter:299 - Executing with tokens: []
2023-08-17 23:13:37 INFO Configuration:2854 - resource-types.xml not found
2023-08-17 23:13:37 INFO ResourceUtils:476 - Unable to find 'resource-types.xml'.
2023-08-17 23:13:37 INFO YarnClientImpl:338 - Submitted application application_1692310718604_0010
2023-08-17 23:13:37 INFO Job:1682 - The url to track the job: http://resourcemanager:8088/proxy/application_1692310718604_0010/
2023-08-17 23:13:37 INFO Job:1727 - Running job: job_1692310718604_0010
2023-08-17 23:13:42 INFO Job:1748 - Job job_1692310718604_0010 running in uber mode : false
```

Esse comando rodará o exemplo em questão e retornará a quantidade de palavras no arquivo output.txt, no entanto para verificar o arquivo output.txt será pelo terminal também.

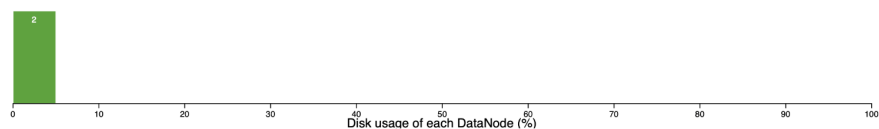
8 - Após a execução do exemplo, coloque o comando `hdfs dfs -ls` para verificar os arquivos no diretório em seguida coloque: `hdfs dfs -ls output.txt` para ver a saída no arquivo output.txt, nota-se que é gerada uma saída em output.txt a `output.txt/part-r-00000`, para visualizar esse arquivo que contém o resultado da contagem das palavras coloque o comando `hdfs dfs -cat output.txt/part-r-00000` e irá aparecer a quantidade de palavras de acordo com o arquivo de texto **input.txt**

```
Bytes Written=37
sh-4.2$ hdfs dfs -ls
Found 2 items
-rw-r--r-- 1 hadoop supergroup 45 2023-08-17 23:12 input.txt
drwxr-xr-x - hadoop supergroup 0 2023-08-17 23:13 output.txt
sh-4.2$ hdfs dfs -ls output.txt
Found 2 items
-rw-r--r-- 1 hadoop supergroup 0 2023-08-17 23:13 output.txt/_SUCCESS
-rw-r--r-- 1 hadoop supergroup 37 2023-08-17 23:13 output.txt/part-r-00000
sh-4.2$ hdfs dfs -get output.txt/part-r-00000
get: /home/docker-3/part-r-00000._COPYING_ (Permission denied)
sh-4.2$ hdfs dfs -cat output.txt/part-r-00000
Hadoop 2
Hello 2
WordCount 1
World 1
```

## Datanode Information



### Datanode usage histogram



In operation

DataNode State		All	Show	25	entries	Search: <input type="text"/>				
					</					