

**Apache Spark**

Spark to platforma obliczeniowa klastrów, która wykorzystuje podstawowe elementy w pamięci, aby umożliwić programom działanie nawet sto razy szybciej niż aplikacje Hadoop MapReduce. Aplikacje Spark składają się z programu sterownika, który kontroluje wykonywanie równoległych operacji w klastrze. Główna abstrakcja programowania zapewniana przez platformę Spark jest znana jako odporne rozproszone zestawy danych (Resilient Distributed Datasets - RDD). RDD to kolekcje elementów podzielonych na węzły klastra, na których można operować równolegle.

Spark został stworzony, aby działać na wielu platformach i być rozwijany w wielu językach. Spark działa na Hadoopie, oprócz tego obsługuje natywnie takie języki jak: Scala, Java, Python i R. Dodatkowo Spark może być używany interaktywnie z powłoki wiersza polecenia.

Przydatne linki

<https://spark.apache.org/docs/latest/>

<https://www.tutorialspoint.com/apache_spark/apache_spark_introduction.htm>

<https://phoenixnap.com/kb/install-spark-on-windows-10>

<https://www.whizlabs.com/blog/do-you-need-hadoop-to-run-spark/>

Sparka można zainstalować na dowolnym systemie, ale poniżej jest instrukcja dla Ubuntu 20.04.

Potrzebna będzie Java oraz Python. Użyjemy wersji “Hadoop free”, aby móc skorzystać z zainstalowanego wcześniej Hadoopa. W razie potrzeby można skorzystać z wersji pre-build dla Haddop.

**Zadania (10 punktów)**

Zadanie 1: zainstaluj Apache Spark (2 pkt)

Instalacja Apache Spark na Ubuntu 20.04 z zainstalowanym Hadoop.

1. Sprawdzamy wersję javy

java -version

1. Sprawdzamy Pythona

python3 –version

1. Instalujemy Sparka

<https://spark.apache.org/downloads.html>

Graphical user interface, text, application

Description automatically generated

* 1. Pobieramy plik

wget https://downloads.apache.org/spark/spark-3.2.1/spark-3.2.1-bin-hadoop3.2.tgz

* 1. Rozpakowujemy

tar -xvzf spark-3.2.1-bin-without-hadoop.tgz

* 1. Przenosimy folder

sudo mv spark-3.2.1-bin-without-hadoop /usr/local/spark

* 1. Konfiguracja
     1. Kopiujemy template jako plik konfiguracyjny

cp /usr/local/spark/conf/spark-env.sh.template /usr/local/spark/conf/spark-env.sh

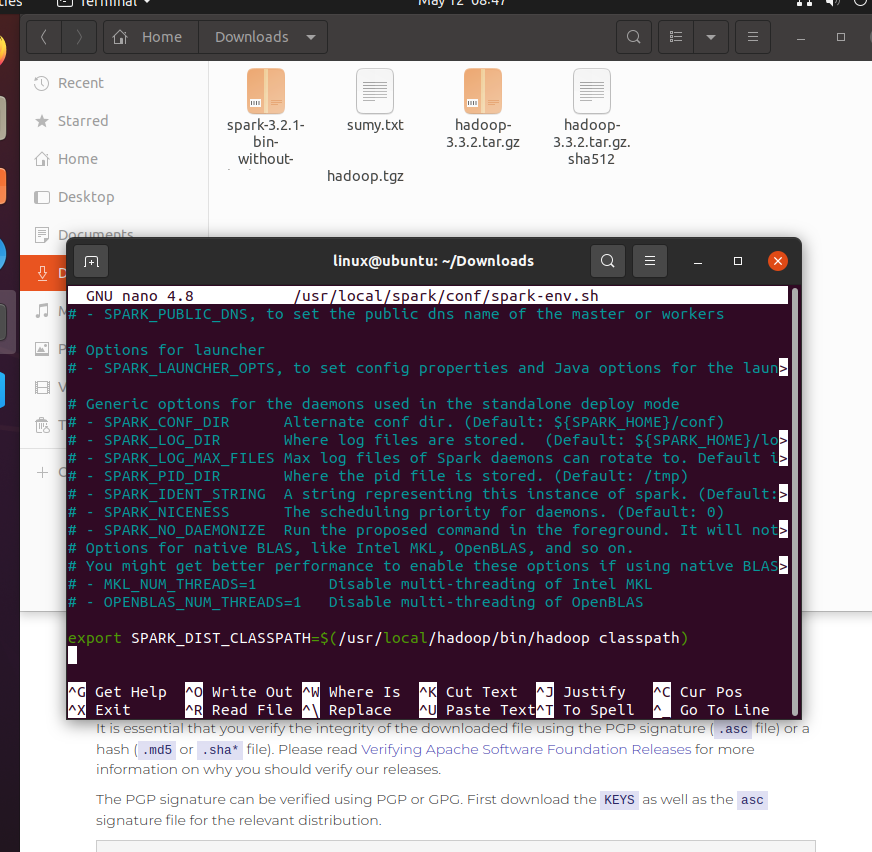
* + 1. Dodajemy wpis do pliku

sudo nano /usr/local/spark/conf/spark-env.sh

dodajemy wpis:

export SPARK\_DIST\_CLASSPATH=$(/usr/local/hadoop/bin/hadoop classpath)

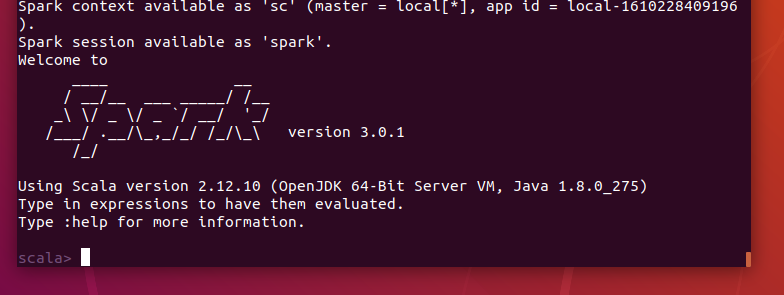
*zapisujemy plik po zmianie (ctr + x, y, enter)*

**

Zadanie 2: sprawdź instalację Apache Spark (wrzuć screeny z shella i local hosta) (2 pkt)

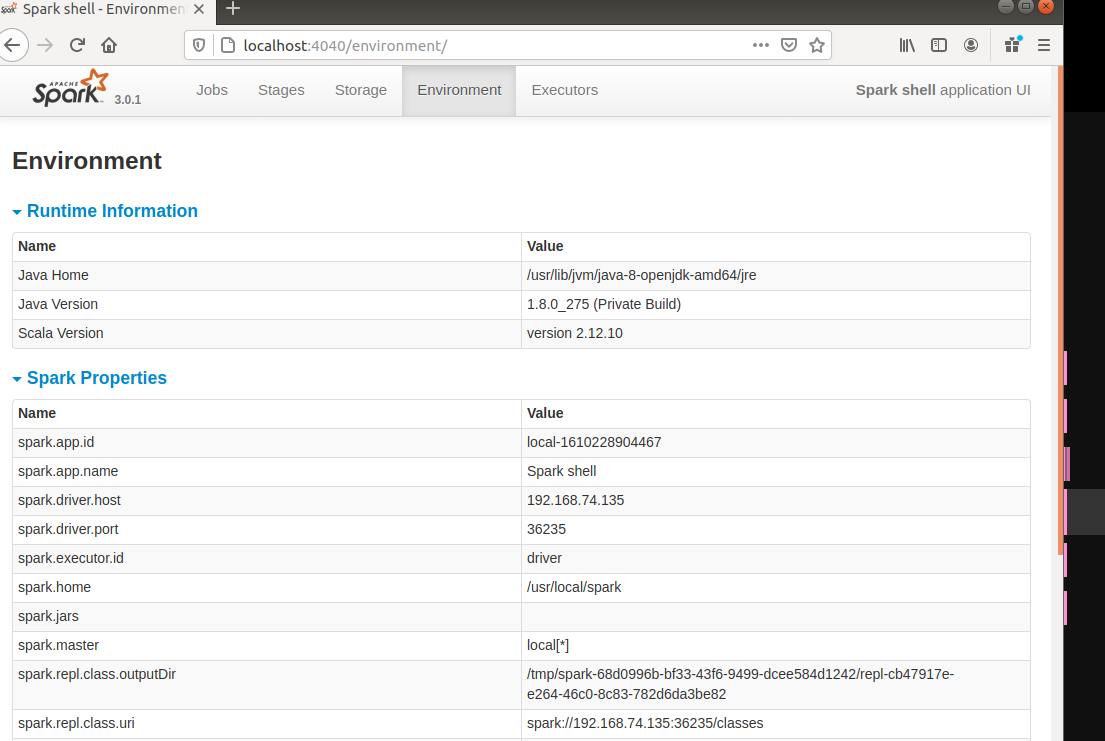
1. Sprawdzamy instalację Sparka

/usr/local/spark/bin/spark-shell



Następnie sprawdzamy w przeglądarce Spark shell application UI wpisując url:

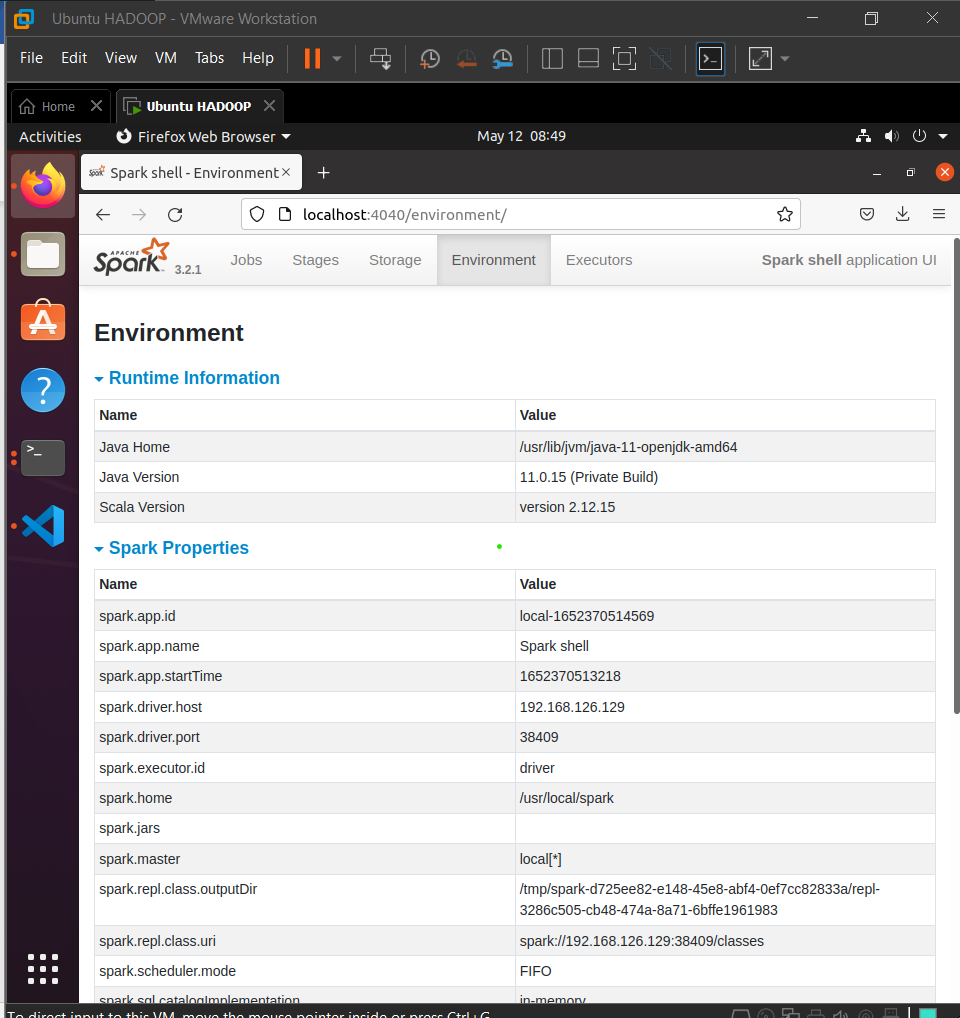
http://localhost:4040



Aby wyjść z shell możemy wpisać np. jedną z poniższych komend:

:q

sys.exit



Zadanie 3: Powłoka Apache Spark (wrzuć screeny z uruchomienia polecenia) (2 pkt)

Powłoka Sparka zapewnia prosty sposób na naukę interfejsu API, a także potężne narzędzie do interaktywnej analizy danych. Jest dostępny w Scali (która działa na maszynie wirtualnej Java i dlatego jest dobrym sposobem na korzystanie z istniejących bibliotek Java) lub w Pythonie. Możemy uruchomić shell Sparka korzystający z Pythona, wpisują w terminal komendę:

/usr/local/spark/bin/pyspark

Jeżeli pojawi się błąd env: ‘python’: No such file or directory

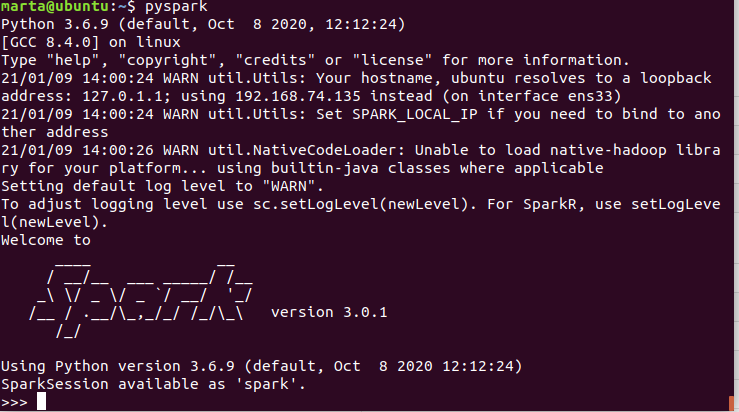
1. Jeżeli nie mamy zainstalowanego python3

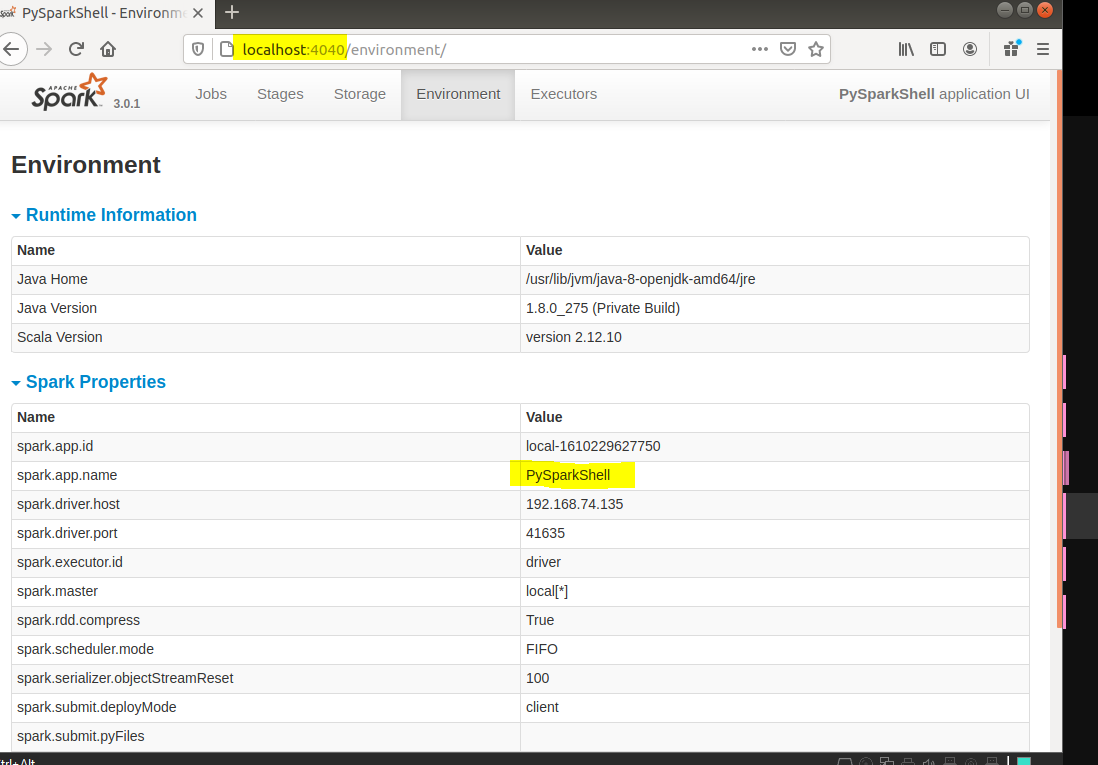
apt-get install python3

1. Jeżeli mamy zainstalowanego python3, a mimo to błąd występuje, może pomóc:

whereis python3

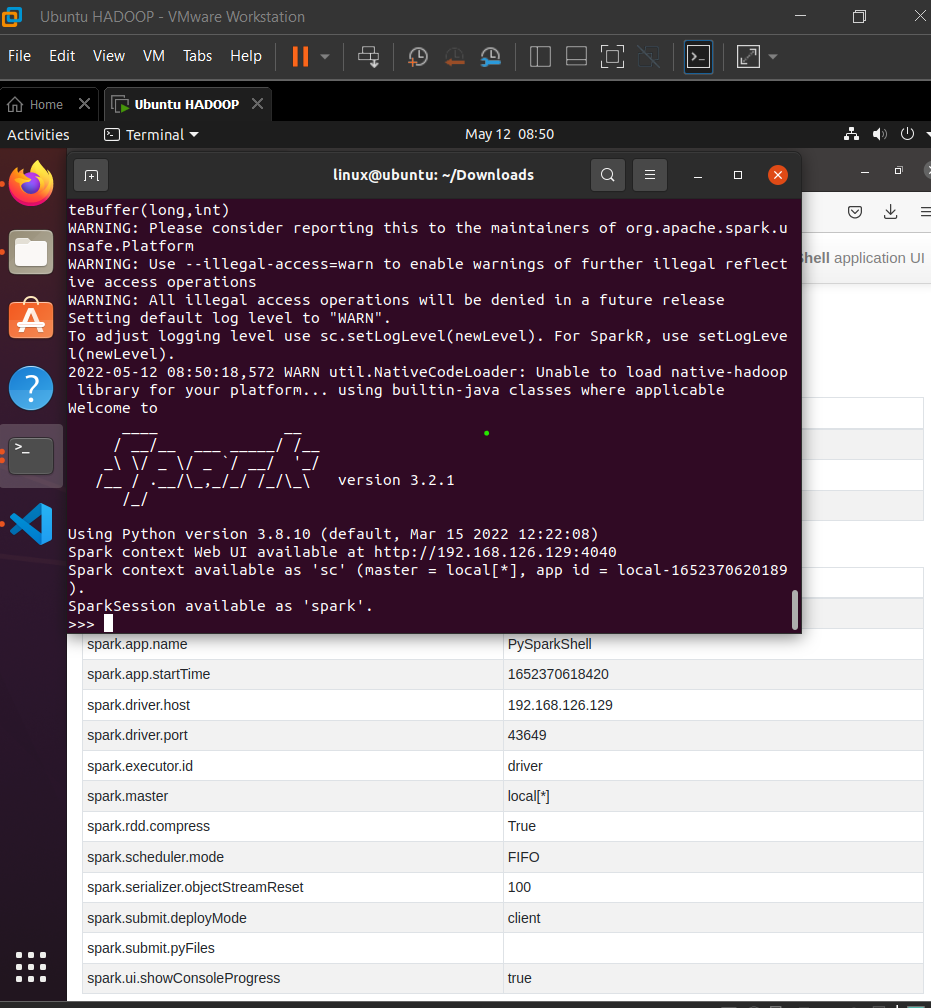
sudo ln -s /usr/bin/python3 /usr/bin/python





Aby wyjść wpisujemy:

exit()



Zadanie 4: Apache Spark i HDFS (wrzuć screeny z uruchomienia polecenia) (2 pkt)

Spark oraz Hadoop są produktami Apache, dlatego dobrze ze sobą wpsółpracują. Spróbujmy korzystając ze Spark odczytać plik umieszczony w HDFS.

1. Uruchamiamy usługi:

rm -rf ~/hadoop

/usr/local/hadoop/bin/hdfs namenode -format

/usr/local/hadoop/sbin/start-dfs.sh

jps

1. Kopiujemy przykładowy plik sparka do hdfs

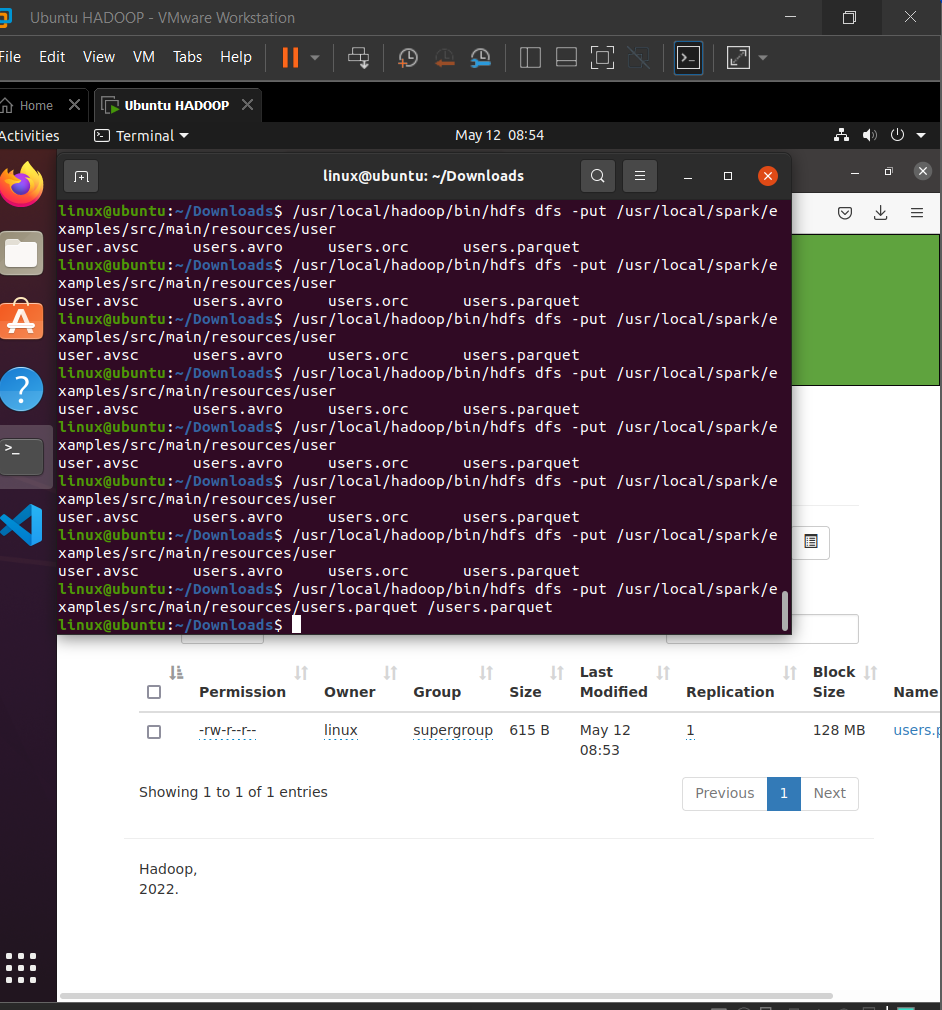
/usr/local/hadoop/bin/hdfs dfs -put /usr/local/spark/examples/src/main/resources/users.parquet /users.parquet

Więcej o parquet format:

<https://spark.apache.org/docs/latest/sql-data-sources-parquet.html>

<https://databricks.com/session/spark-parquet-in-depth>

<http://parquet.apache.org/>



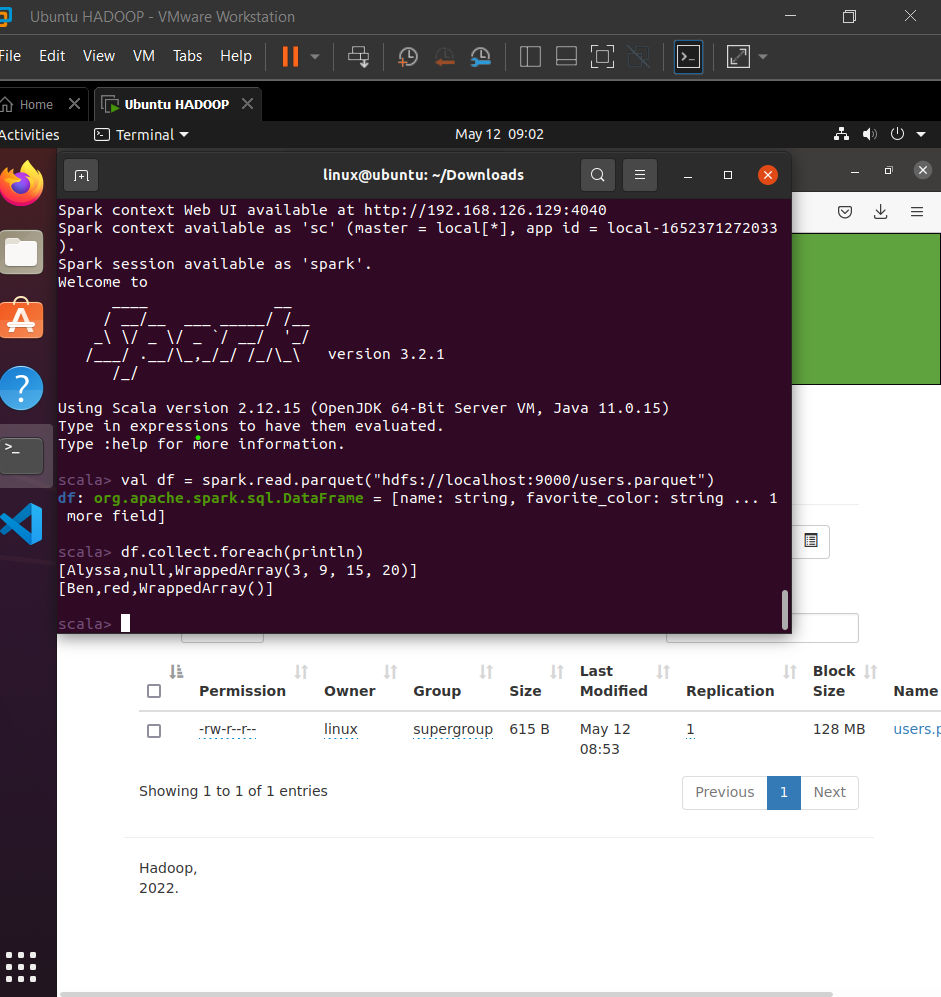
Zadanie 5: Apache Spark i HDFS (wrzuć screeny z uruchomienia polecenia) (2 pkt)

1. Otwórz spark-shell i odczytaj plik

df.collect.foreach(println)

*Oczekiwany output:*





1. Sprawdź jak to wygląda w UI

*Oczekiwany output:*

