Submit Apache Spark Applications Lab

Estimated time needed: 20 minutes

In this lab, you will learn how to submit Apache Spark applications from a python script. This exercise is straightforward thanks to Docker Compose.

Learning Objectives

In this lab, you will:

- Install a Spark Master and Worker using Docker Compose
- Create a python script containing a spark job
- Submit the job to the cluster directly from python (Note: you'll learn how to submit a job from the command line in the Kubernetes Lab)

Prerequisites

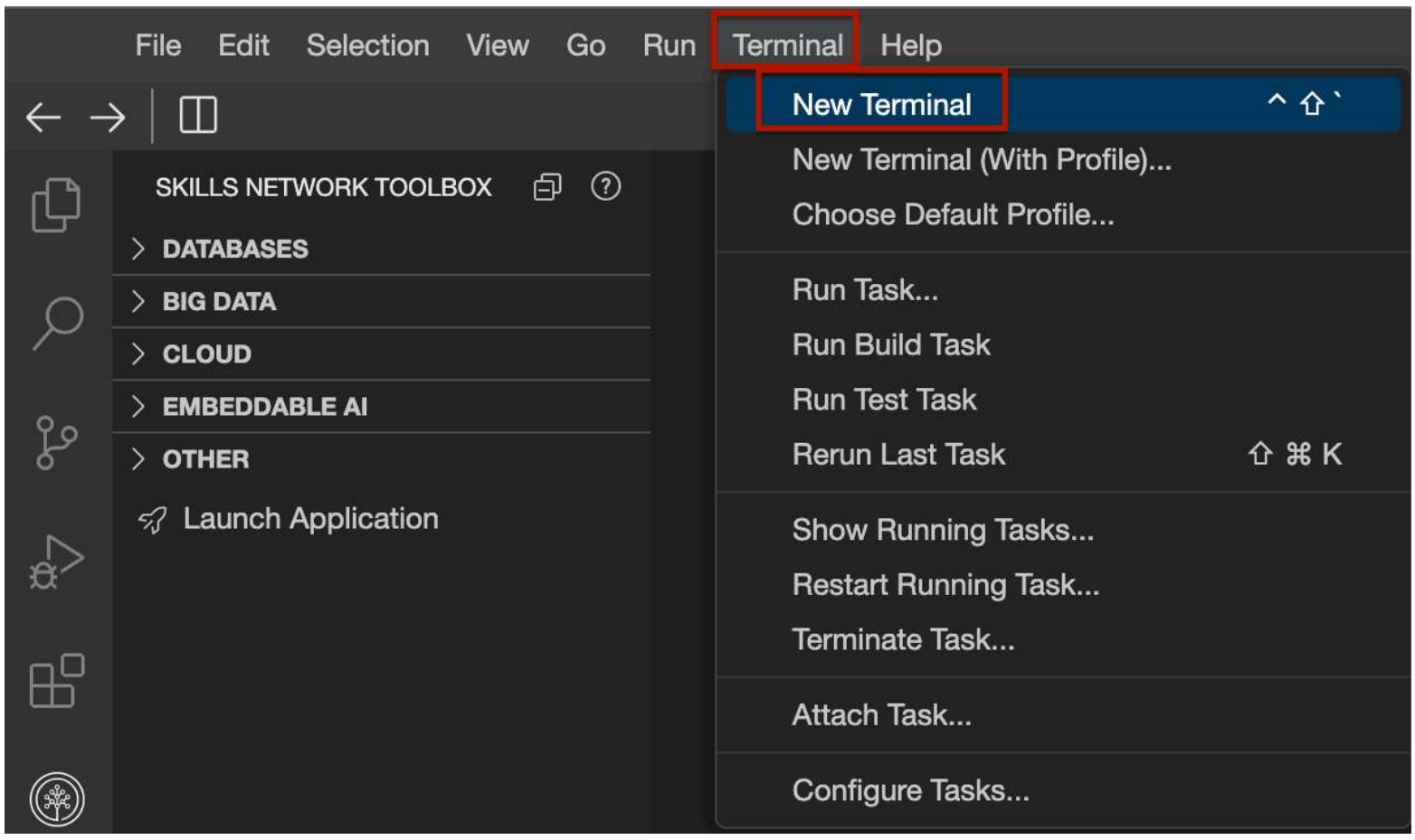
Note: If you are running this lab within the Skillsnetwort Lab environment, all prerequisites are already installed for you

The only pre-requisites to this lab are:

- A working *docker* installation
- Docker Compose
- The *git* command line tool
- A python development environment

Install a Apache Spark cluster using Docker Compose

On the right hand side to this instructions you'll see the Cloud IDE. Select the *Lab* tab. On the menu bar select *Terminal>New Terminal*.



Get the latest code:

```
1. 1
```

```
1. git clone https://github.com/big-data-europe/docker-spark.git
```

Copied! Executed!

Change the directory to the downloaded code:

```
1. 1
```

```
1. cd docker-spark
```

Copied! Executed!

Start the cluster

```
1. 1
```

```
1. docker-compose up
```

Copied! Executed!

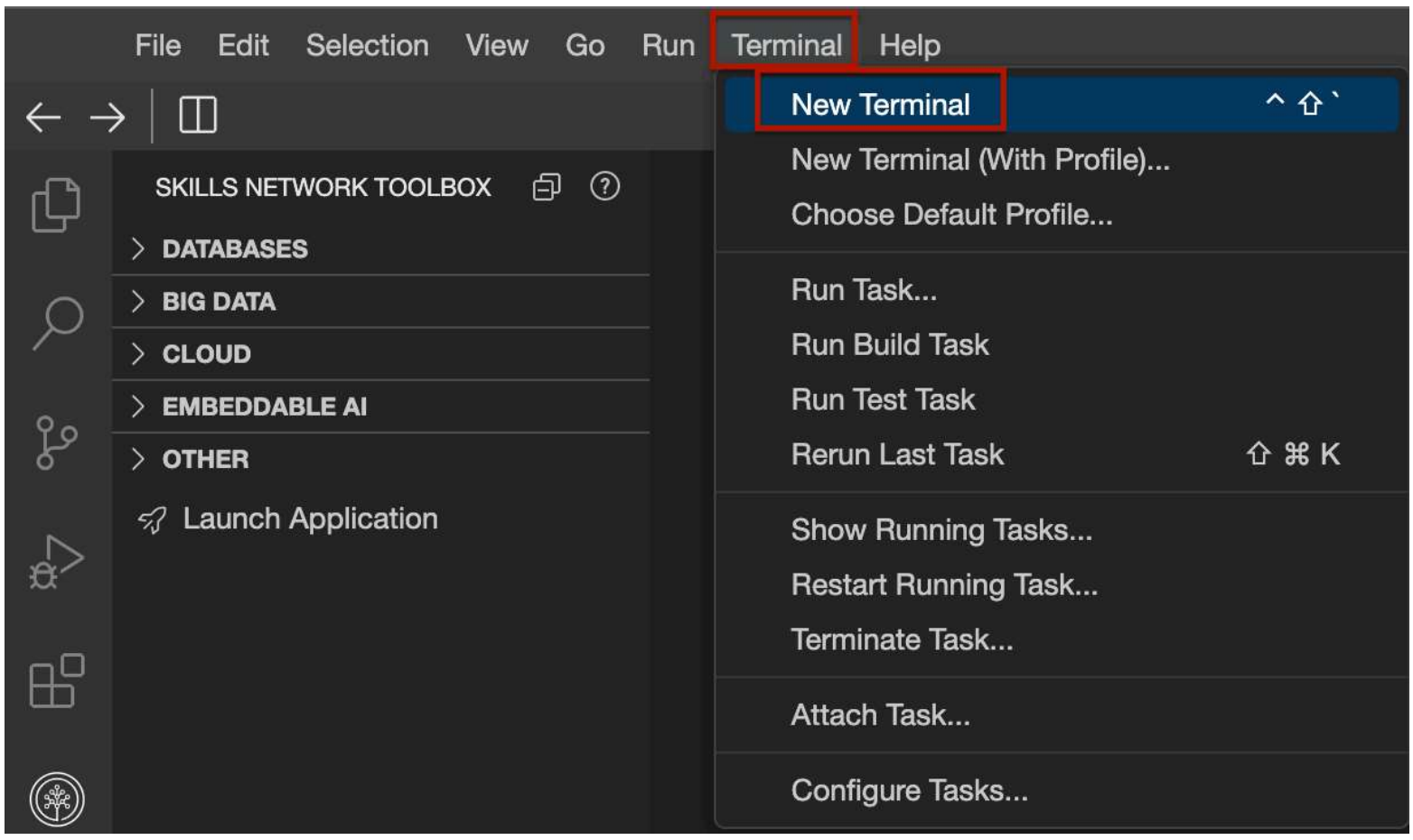
After quite some time you should see the following message:
Successfully registered with master spark://<server address>:7077

```
spark-worker-1  | 22/01/14 05:39:36 INFO Worker: Successfully registered with
master spark://50bfae057469:7077
```

NOTE: **Please keep this terminal running and do not close it, as it is essential for further steps in the lab to run correctly.**

Create code

1. Click `Terminal` from the menu, and click `New Terminal` to open a new terminal window.



2. Once the terminal opens up at the bottom of the window, please type in the following command in the terminal to create the Python script.

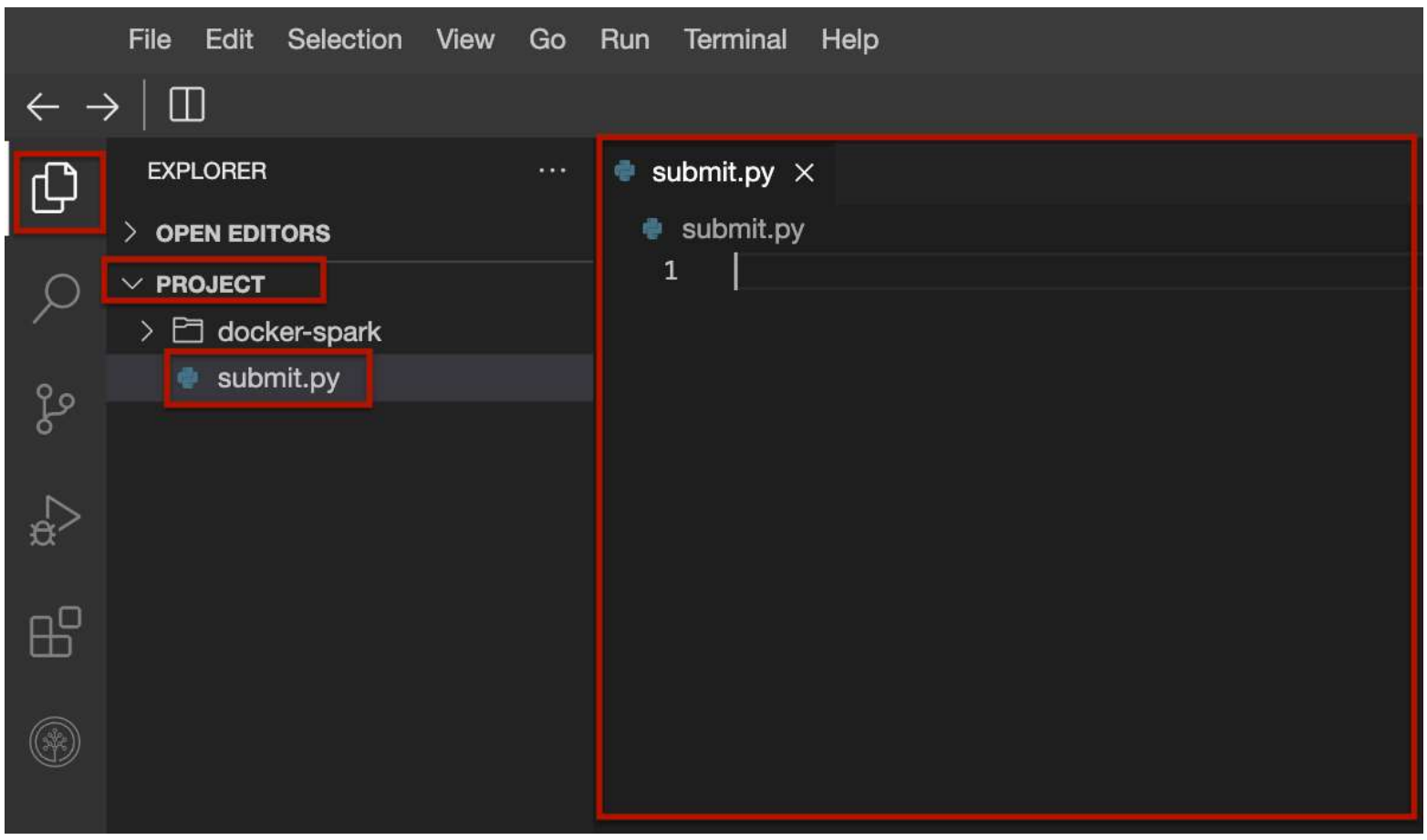
```
1. 1
```

```
1. touch submit.py
```

Copied! Executed!

A new python file called `submit.py` is created.

3. Open the file in the file editor.



4. Paste the following code to the file and save.

```
 1. 1
 2. 2
 3. 3
 4. 4
 5. 5
 6. 6
 7. 7
 8. 8
 9. 9
10. 10
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
20. 20
21. 21
22. 22
23. 23
24. 24
25. 25
26. 26
```

```
1. import findspark
2. findspark.init()
3. from pyspark import SparkContext, SparkConf
4. from pyspark.sql import SparkSession
5. from pyspark.sql.types import StructField, StructType, IntegerType, StringType
6.
7. sc = SparkContext.getOrCreate(SparkConf().setMaster('spark://localhost:7077'))
```

```
sc.setLogLevel("INFO")

spark = SparkSession.builder.getOrCreate()

spark = SparkSession.builder.getOrCreate()
df = spark.createDataFrame(
    [
        (1, "foo"),
        (2, "bar"),
    ],
    StructType(
        [
            StructField("id", IntegerType(), False),
            StructField("txt", StringType(), False),
        ]
    ),
)
print(df.dtypes)
df.show()
```

Copied!

Execute code / submit Spark job

Now we execute the python file we saved earlier.

1. In the terminal, run the following commands to upgrade the pip installer to ensure you have the latest version by running the following commands.

```
1
2
3
```

```
rm -r ~/.cache/pip/selfcheck/
pip3 install --upgrade pip
pip install --upgrade distro-info
```

Copied! Executed!

`rm -r ~/.cache/pip/selfcheck/` removes any previously cached version of pip and allows to install the latest one.

2. Please enter the following commands in the terminal to download the spark environment.

```
1
```

```
wget https://archive.apache.org/dist/spark/spark-3.3.3/spark-3.3.3-bin-hadoop3.tgz && tar xf spark-3.3.3-bin-hadoop3.tgz && rm -rf spark-3.3.3-bin-hadoop3.tgz
```

Copied! Executed!

This takes a while. This downloads the spark as a zipped archive and unzips it in the current directory.

3. Run the following commands to set up the `JAVA_HOME` which is preinstalled in the environment and `SPARK_HOME` which you just downloaded.

```
1
2
```

```
export JAVA_HOME=/usr/lib/jvm/java-1.11.0-openjdk-amd64
export SPARK_HOME=/home/project/spark-3.3.3-bin-hadoop3
```

Copied! Executed!

4. Install the required packages to set up the spark environment.

```
1
```

```
pip install pyspark
```

Copied! Executed!

```
1
```

```
python3 -m pip install findspark
```

Copied! Executed!

5. Type in the following command in the terminal to execute the Python script.

```
1
```

```
python3 submit.py
```

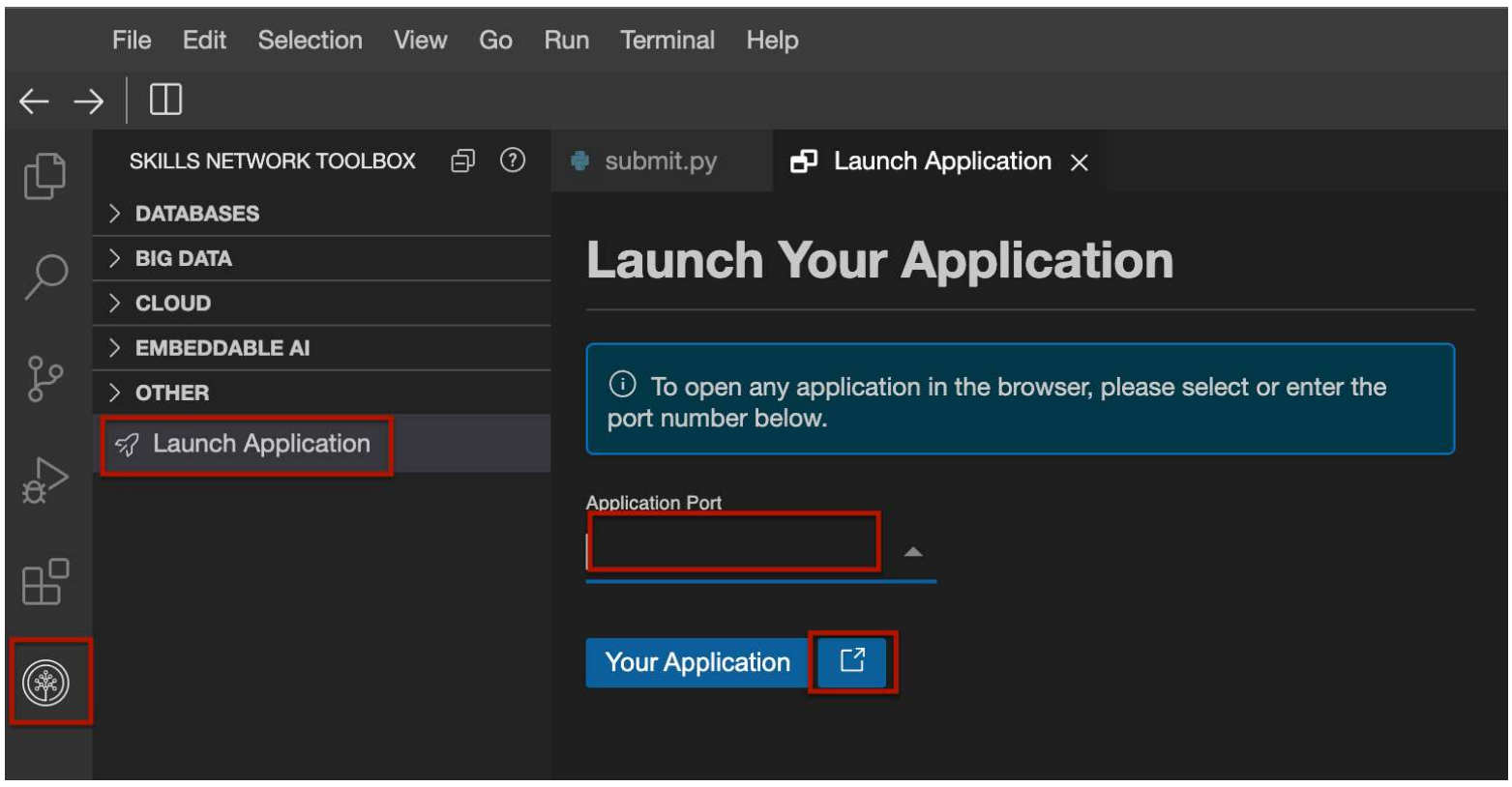
Copied! Executed!

Experiment yourself

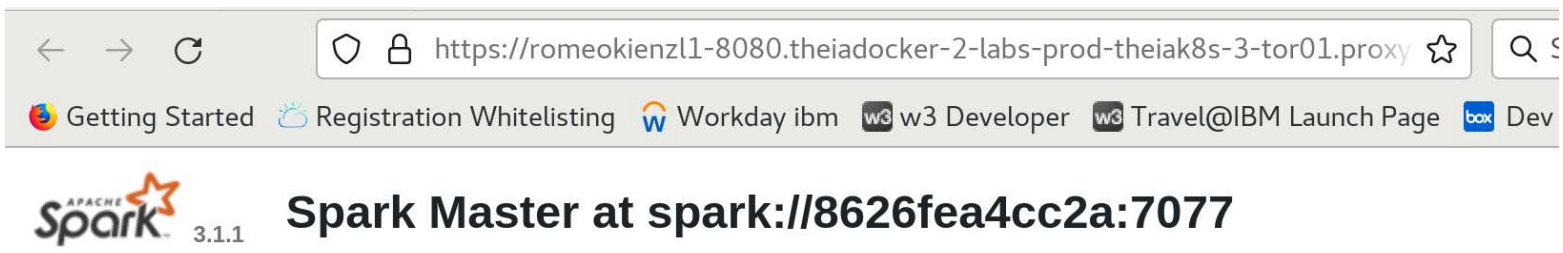
Please have a look at the UI of the Apache Spark master and worker.

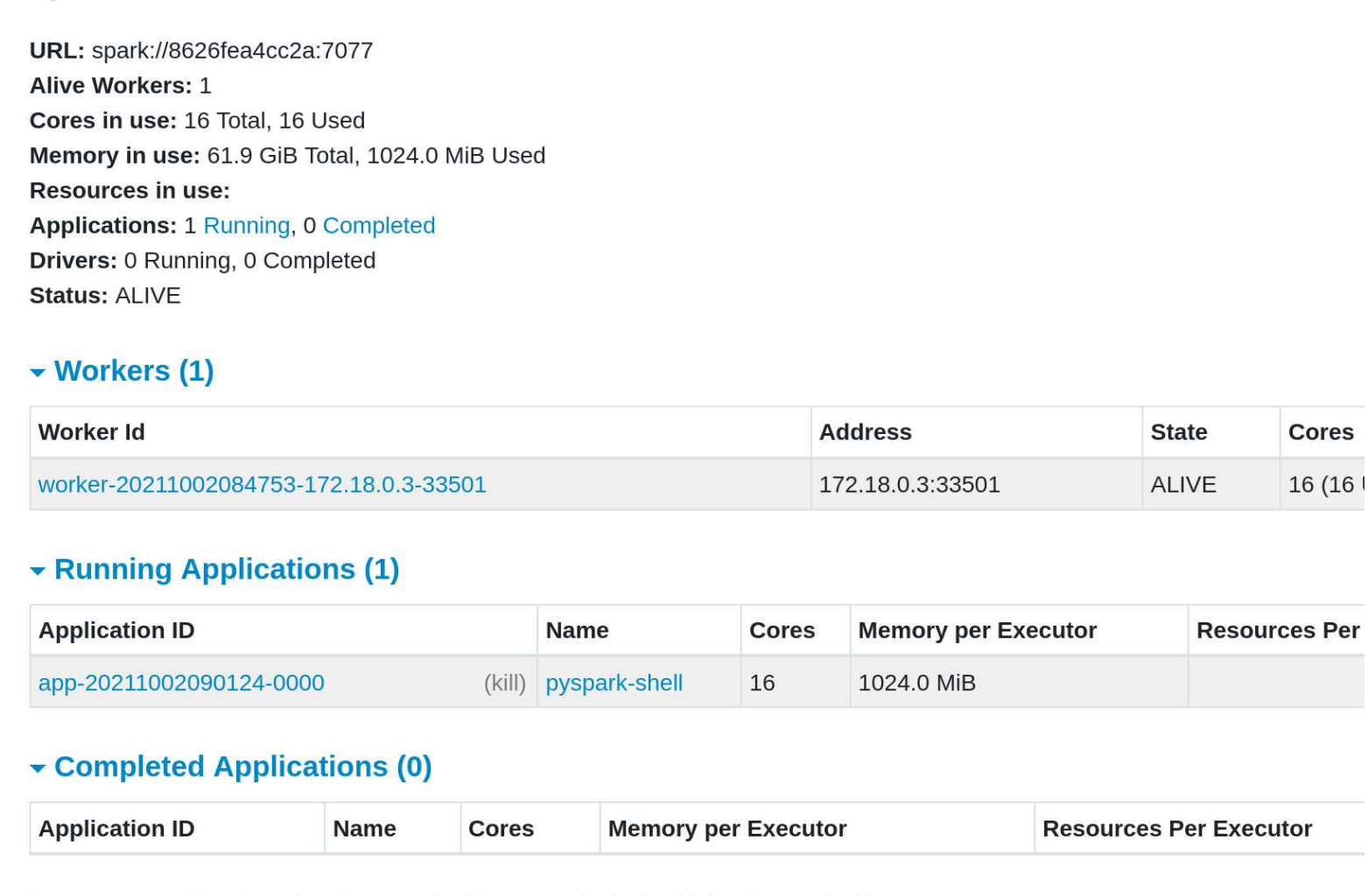
1. Click on the button below to launch the `Spark Master`. Alternatively, click on the Skills Network button on the left, it will open the “Skills Network Toolbox”. Then click the `Other`, then `Launch Application`. From there you should be able to enter the port number as `8080` and launch.

Spark Master



2. This will take you to the admin UI of the Spark master (if your popup blocker doesn't prevent it).





3. Please notice that you can see all registered workers (one in this case) and submitted jobs (also one in this case)

   Note: The way how the lab environment works you can't click on links in the UI - in a real installation, this of course is possible.

4. Click the button below to open the `Spark Worker` on 8081. Alternatively, click on the Skills Network button on the left, it will open the "Skills Network Toolbox". Then click the `Other`, then `Launch Application`. From there, you should be able to enter the port number as `8081` and launch.

Spark Worker

You should find your currently running job here as well.

Summary

In this lab you've learned how to setup an experimental Apache Spark cluster on top of Docker Compose. You are now able to submit a Spark job directly from python code. In the Kubernetes lab you'll learn how to subit Spark jobs from command line as well.

Author(s)

Romeo Kienzler
Lavanya T S

Changelog

Date	Version	Changed by	Change Description
October 2021	1.0	Romeo Kienzler	Initial version
01-09-2022	1.1	K Sundararajan	Updated instructions for `Launch Application` as per new Theia IDE & the format for `Changelog`
30-12-2022	1.2	K Sundararajan	Updated `pyspark` installation command
16-10-2023	1.3	Lavanya T S	Updated and restructured the lab

Date	Version	Changed by	Change Description
17-10-2023	1.4	K Sundararajan	Made few more updates based on Beta testing osbervations