

LAPORAN TUGAS MATA KULIAH  
BIG DATA  
Spark Docker



NAMA : MUHAMMAD IQBAL MUHTARAM  
NIM : 2241720265  
KELAS : TI 3D

PROGRAM STUDI D-IV TEKNIK INFORMATIKA  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI MALANG

Jl. Soekarno Hatta No.9, Jatimulyo, Kec. Lowokwaru, Kota Malang, Jawa Timur 65141

## Prasyarat

1. Windows 10/11 (64-bit) dengan versi Pro, Enterprise, atau Education
2. Docker Desktop untuk Windows diinstal dan berjalan
3. WSL 2 (Windows Subsystem for Linux versi 2) diaktifkan

```
Prompt Perintah
(c) Microsoft Corporation. All rights reserved.

C:\Users\Ponse>wsl --update
The requested operation requires elevation.
Downloading: Windows Subsystem for Linux 2.4.13
Installing: Windows Subsystem for Linux 2.4.13
Windows Subsystem for Linux 2.4.13 has been installed.
The operation completed successfully.
Checking for updates.
The most recent version of Windows Subsystem for Linux is already installed.

C:\Users\Ponse>wsl --set-default-version 2
For information on key differences with WSL 2 please visit https://aka.ms/wsl2
The operation completed successfully.

C:\Users\Ponse>wsl --list --verbose
Windows Subsystem for Linux has no installed distributions.
You can resolve this by installing a distribution with the instructions below:

Use 'wsl.exe --list --online' to list available distributions
and 'wsl.exe --install <Distro>' to install.

C:\Users\Ponse>wsl --list --online
The following is a list of valid distributions that can be installed.
Install using 'wsl.exe --install <Distro>'.

NAME                                FRIENDLY NAME
AlmaLinux-8                         AlmaLinux OS 8
AlmaLinux-9                         AlmaLinux OS 9
AlmaLinux-Kitten-10                 AlmaLinux OS Kitten 10
Debian                             Debian GNU/Linux
FedoraLinux-42                      Fedora Linux 42
SUSE-Linux-Enterprise-15-SP5        SUSE Linux Enterprise 15 SP5
SUSE-Linux-Enterprise-15-SP6        SUSE Linux Enterprise 15 SP6
Ubuntu                             Ubuntu
Ubuntu-24.04                        Ubuntu 24.04 LTS
archlinux                           Arch Linux
kali-linux                          Kali Linux Rolling
openSUSE-Tumbleweed                 openSUSE Tumbleweed
openSUSE-Leap-15.6                  openSUSE Leap 15.6
Ubuntu-18.04                        Ubuntu 18.04 LTS
Ubuntu-20.04                        Ubuntu 20.04 LTS
Ubuntu-22.04                        Ubuntu 22.04 LTS
OracleLinux_7_9                     Oracle Linux 7.9
OracleLinux_8_7                     Oracle Linux 8.7
OracleLinux_9_1                     Oracle Linux 9.1
```

```
Prompt Perintah

val rdd = sc.parallelize(textData)
val wordCounts = rdd.flatMap(line => line.split(" "))
                      .map(word => (word, 1))
                      .reduceByKey(_ + _)
wordCounts.collect().foreach(println)
textData: List[String] = List(Hello Spark, Hello Docker, Spark is awesome, Docker makes Spark easy)

scala> rdd: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[4] at parallelize at <console>:24

scala> wordCounts: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[5] at flatMap at <console>:23

scala> res4: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[6] at map at <console>:24

scala> res5: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[7] at reduceByKey at <console>:24

scala> Hello
Spark
Hello
Docker
Spark
is
awesome
Docker
makes
Spark
easy

scala> val textData = List("Hello Spark", "Hello Docker", "Spark is awesome", "Docker makes Spark easy")
val rdd = sc.parallelize(textData)
val wordCounts = rdd.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
wordCounts.collect().foreach(println)
textData: List[String] = List(Hello Spark, Hello Docker, Spark is awesome, Docker makes Spark easy)

scala> rdd: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[8] at parallelize at <console>:24

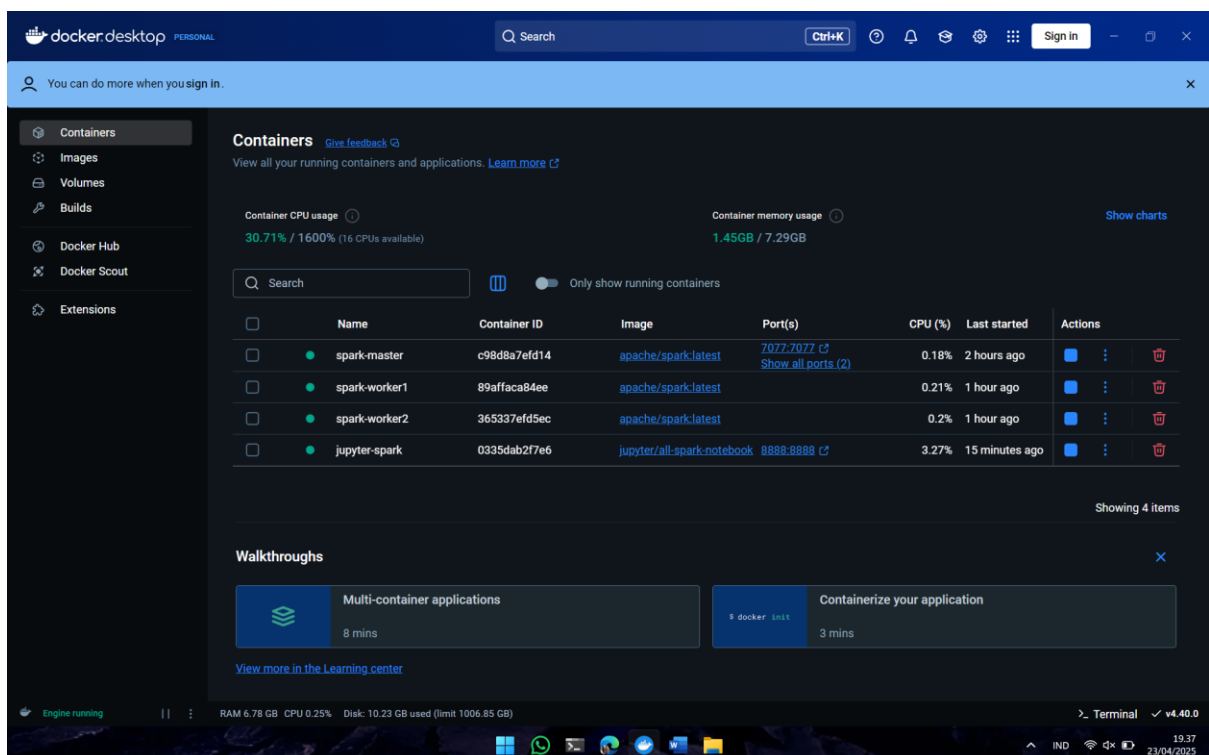
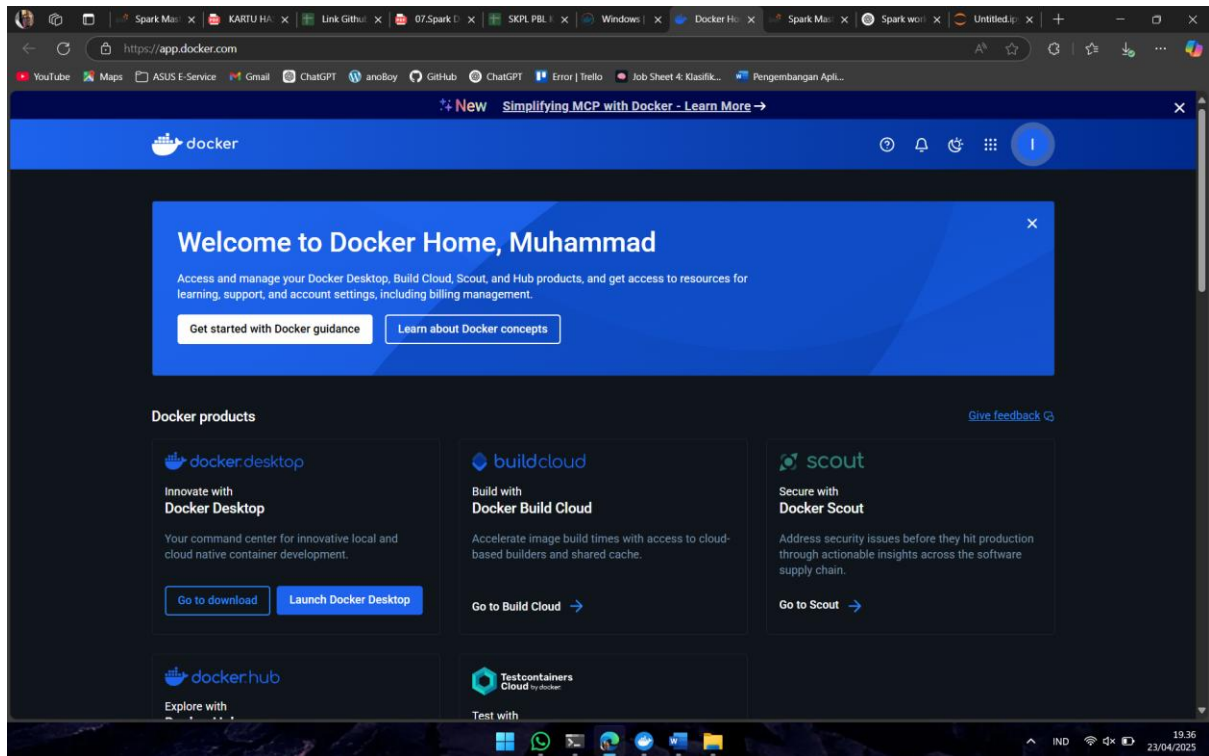
scala> wordCounts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[11] at reduceByKey at <console>:23

(Docker,2)
(Spark,3)
(makes,1)
(is,1)
(Hello,2)
(easy,1)
(awesome,1)

scala> System.exit(0)
```

## 2. Menjalankan Spark Master

## 3. Menjalankan Spark Worker



## 4. Mengakses Spark Web UI

**Spark Master at spark://172.18.0.4:7077**

URL: spark://172.18.0.4:7077  
 Alive Workers: 2  
 Cores in use: 4 Total, 0 Used  
 Memory in use: 4.0 GiB Total, 0.0 B Used  
 Resources in use:  
 Applications: 0 Running, 4 Completed  
 Drivers: 0 Running, 0 Completed  
 Status: ALIVE

**Workers (2)**

Worker Id	Address	State	Cores	Memory	Resources
worker-20250423111246-172.18.0.2-37931	172.18.0.2:37931	ALIVE	2 (0 Used)	2.0 GiB (0.0 B Used)	
worker-20250423111246-172.18.0.3-36707	172.18.0.3:36707	ALIVE	2 (0 Used)	2.0 GiB (0.0 B Used)	

**Running Applications (0)**

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

**Completed Applications (4)**

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20250423121028-0003	WordCount	4	1024.0 MiB		2025/04/23 12:10:28	joyan	FINISHED	11 min
app-20250423114542-0002	PySparkShell	4	1024.0 MiB		2025/04/23 11:45:42	spark	FINISHED	6.1 min
app-20250423113527-0001	PySparkShell	4	1024.0 MiB		2025/04/23 11:35:27	spark	FINISHED	9.9 min
app-20250423111732-0000	Spark shell	4	1024.0 MiB		2025/04/23 11:17:32	spark	FINISHED	18 min

5. Menjalankan Spark Shell

6. Menggunakan Jupyter Notebook dengan Spark

## Contoh Program Word Count dengan Spark di Docker

Berikut adalah contoh program Word Count (menghitung kemunculan kata) menggunakan Apache Spark yang bisa dijalankan di lingkungan Docker

Cara 1: Menggunakan Spark Shell

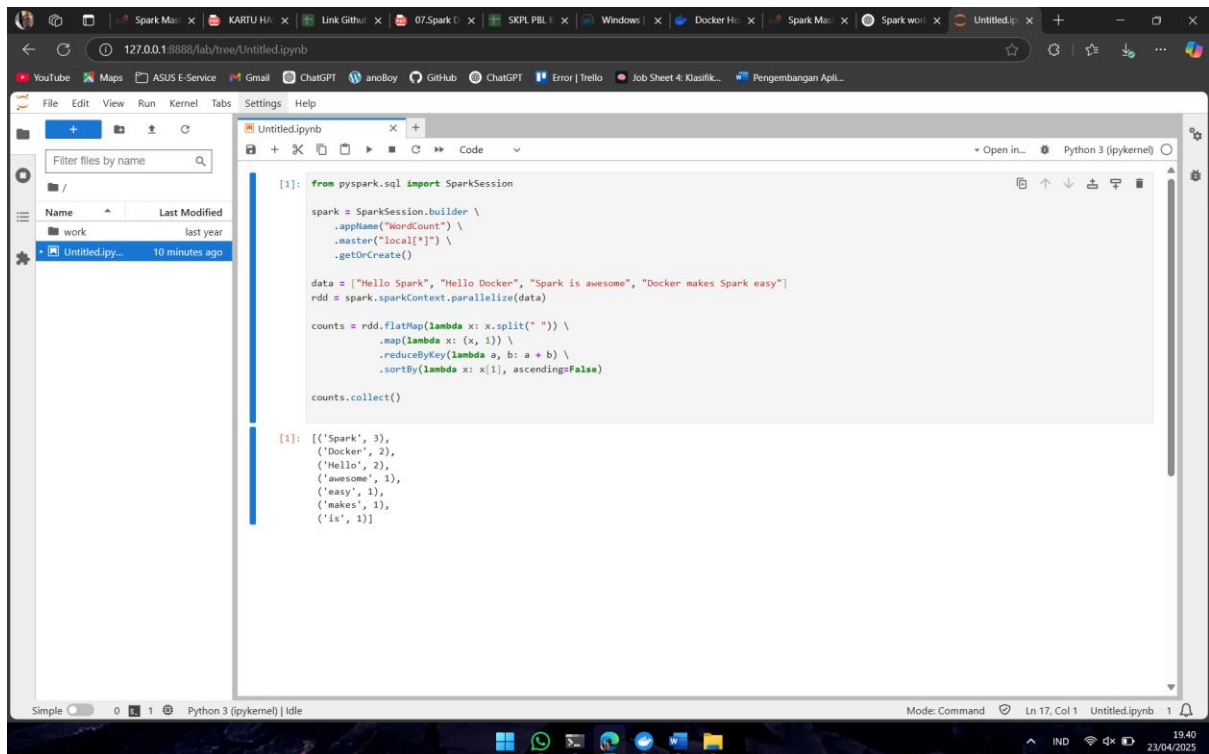
1. Jalankan Spark Shell di Docker seperti contoh di atas

2. Ketikkan kode berikut di Spark Shell:

Cara 2: Menggunakan PySpark (Python)

Cara 3: Menggunakan Jupyter Notebook

## Menjalankan Program sebagai Script



The screenshot shows a Jupyter Notebook titled 'Untitled.ipynb' running on a Python 3 (ipykernel) environment. The notebook contains a Spark word count program. The code defines a SparkSession, creates an RDD from a list of strings, and performs a word count using flatMap, map, reduceByKey, and sortBy. The output of the program is displayed as a list of tuples, where each tuple contains a word and its count.

```
[1]: from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName("WordCount") \
    .master("local[*]") \
    .getOrCreate()

data = ["Hello Spark", "Hello Docker", "Spark is awesome", "Docker makes Spark easy"]
rdd = spark.sparkContext.parallelize(data)

counts = rdd.flatMap(lambda x: x.split(" ")) \
    .map(lambda x: (x, 1)) \
    .reduceByKey(lambda a, b: a + b) \
    .sortBy(lambda x: x[1], ascending=False)

counts.collect()
```

```
[1]: [('Spark', 3),
      ('Docker', 2),
      ('Hello', 2),
      ('awesome', 1),
      ('easy', 1),
      ('makes', 1),
      ('is', 1)]
```