University of Wolverhampton

School of Mathematics and Computer Science

5CS022 Distribute and Cloud Systems Programming

Workshop 4: The Apache Spark Framework

Overview

Apache Spark is an open-source data processing framework which can perform analytic operations on Big Data in a distributed environment. It is compatible with both the Scala and Java programming languages.

This workshop shows you how to set up a simple Spark project in Eclipse and run a word counting program in Java.

1. Configuring Hadoop

To enable Apache Spark to work with Eclipse, some Hadoop utilities need to be available for Spark to call. Rather than downloading and installing the whole of Hadoop, which is about 500MB, on Canvas, there is a file called "hadoop.zip" which contains just the utilities that Spark needs for this workshop. Download it and extract the contents so that they are in the directory **C:\hadoop\bin**

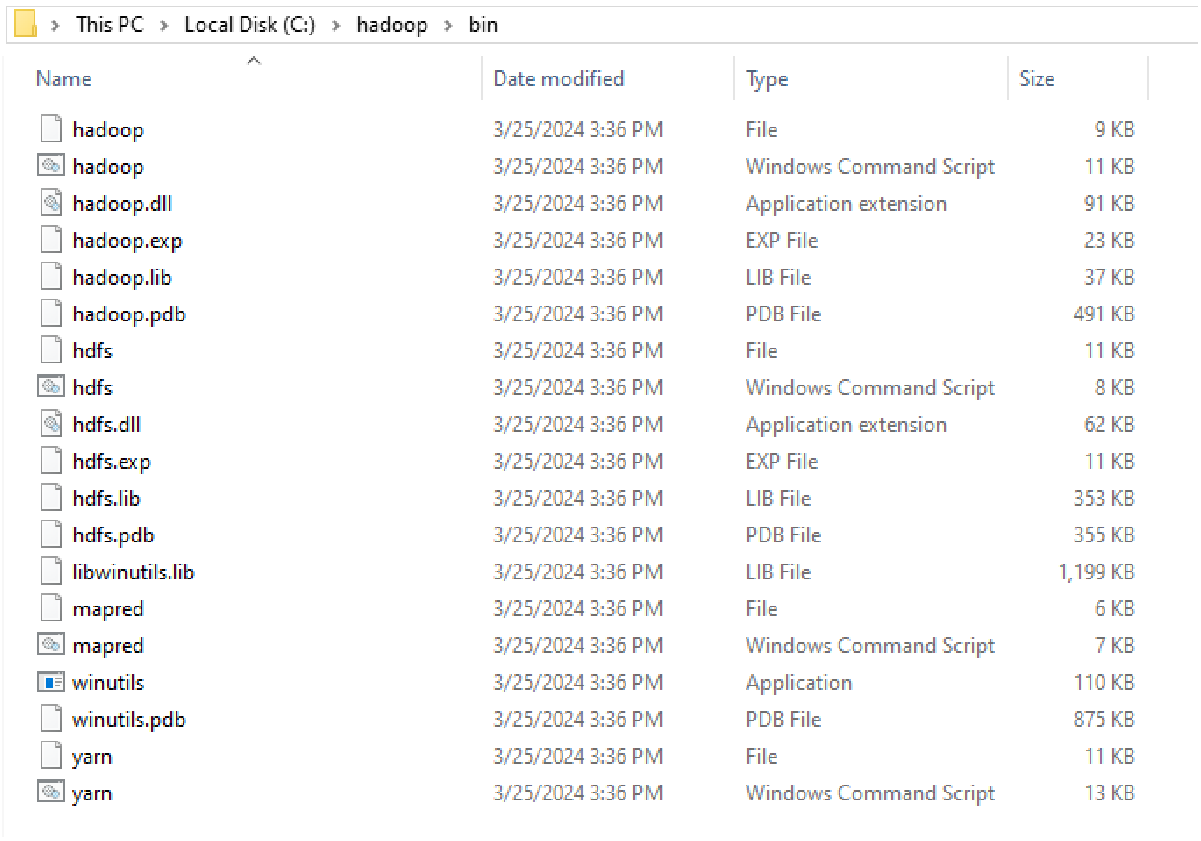
OR,

You can clone this repo: https://github.com/steveloughran/winutils/ and copy the contents of Hadoop-3.0.0 to the C drive and rename the folder to **hadoop.**

If you want to clone, you need to have Git installed on your laptop.
Once installation is done, simply open either Git Bash or your command terminal and type:

git clone https://github.com/steveloughran/winutils.git

This will clone the repository. Find the folder **hadoop-3.0.0** and copy it to the C: drive. Make sure to rename this folder to **hadoop**.

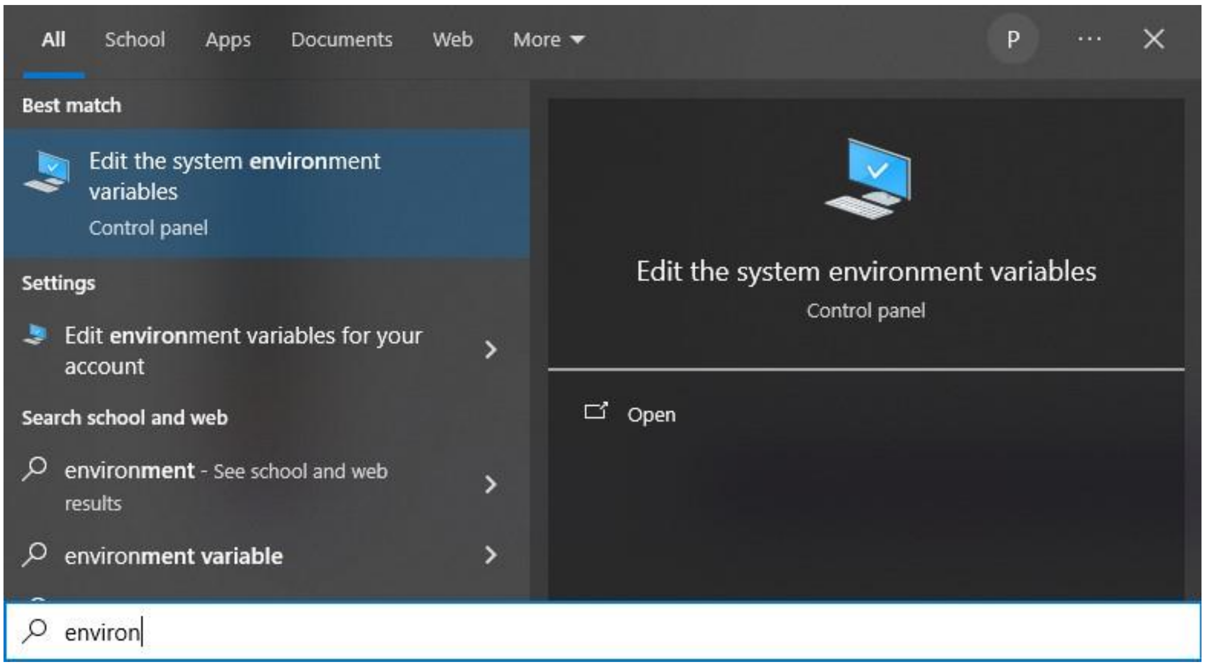


This PC > Local Disk (C:) > hadoop > bin

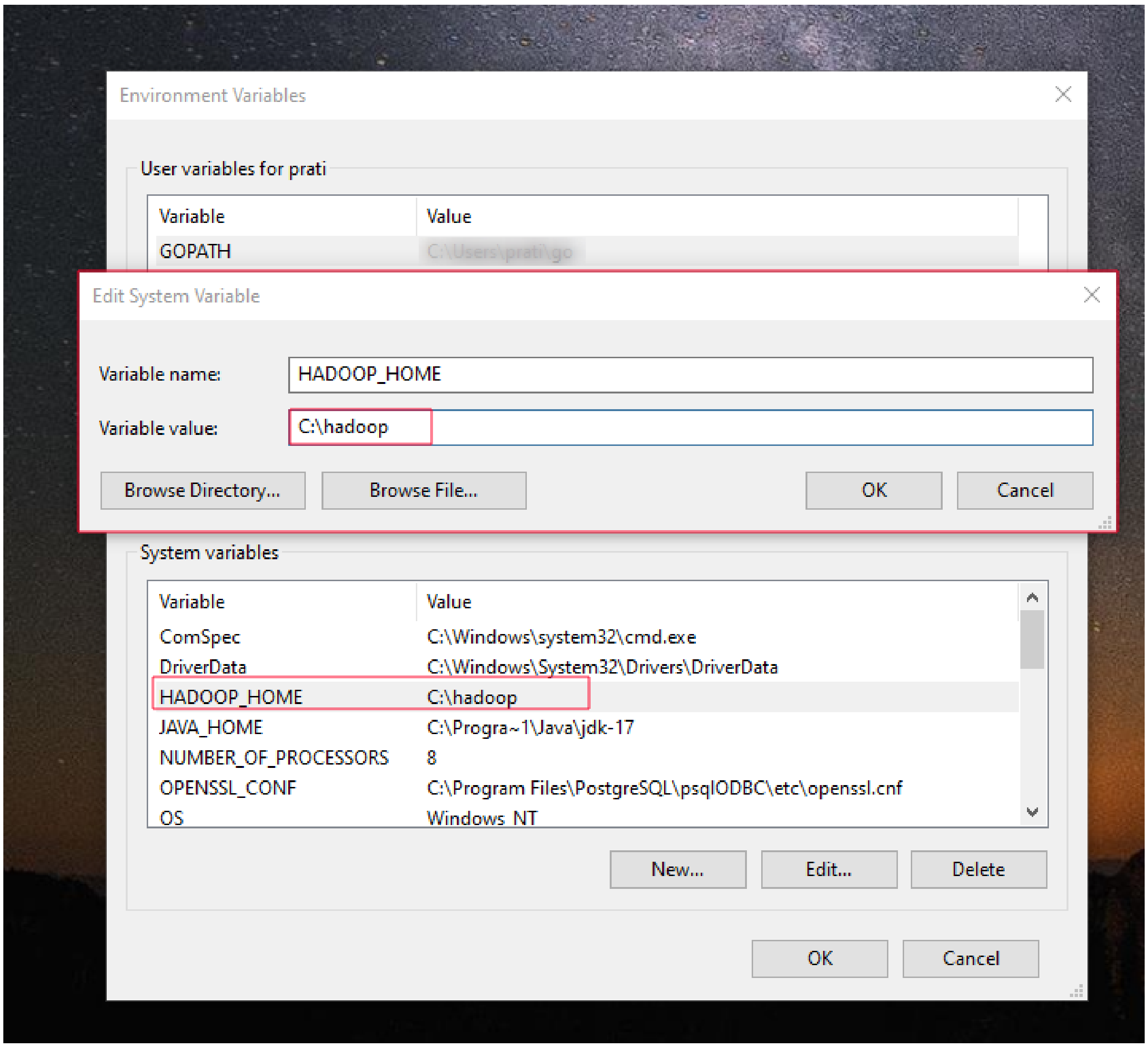
Name	Date modified	Type	Size
hadoop	3/25/2024 3:36 PM	File	9 KB
hadoop	3/25/2024 3:36 PM	Windows Command Script	11 KB
hadoop.dll	3/25/2024 3:36 PM	Application extension	91 KB
hadoop.exp	3/25/2024 3:36 PM	EXP File	23 KB
hadoop.lib	3/25/2024 3:36 PM	LIB File	37 KB
hadoop.pdb	3/25/2024 3:36 PM	PDB File	491 KB
hdfs	3/25/2024 3:36 PM	File	11 KB
hdfs	3/25/2024 3:36 PM	Windows Command Script	8 KB
hdfs.dll	3/25/2024 3:36 PM	Application extension	62 KB
hdfs.exp	3/25/2024 3:36 PM	EXP File	11 KB
hdfs.lib	3/25/2024 3:36 PM	LIB File	353 KB
hdfs.pdb	3/25/2024 3:36 PM	PDB File	355 KB
libwinutils.lib	3/25/2024 3:36 PM	LIB File	1,199 KB
mapred	3/25/2024 3:36 PM	File	6 KB
mapred	3/25/2024 3:36 PM	Windows Command Script	7 KB
winutils	3/25/2024 3:36 PM	Application	110 KB
winutils.pdb	3/25/2024 3:36 PM	PDB File	875 KB
yarn	3/25/2024 3:36 PM	File	11 KB
yarn	3/25/2024 3:36 PM	Windows Command Script	13 KB

Next you need to set up the HADOOP_HOME environment variable to tell Apache Spark where to find Hadoop.

Start the "Advance Systems Settings" control panel in Windows:


All
School
Apps
Documents
Web
More
P
Best match
Edit the system environment variables
Control panel
Settings
Edit environment variables for your account
Search school and web
environment - See school and web results
environment variable
Edit the system environment variables
Control panel
Open
environ

Then create a new System Environment Variable called HADOOP_HOME and set it to "C:\hadoop"



Environment Variables

User variables for prati

Variable	Value
GOPATH	C:\Users\prati\go

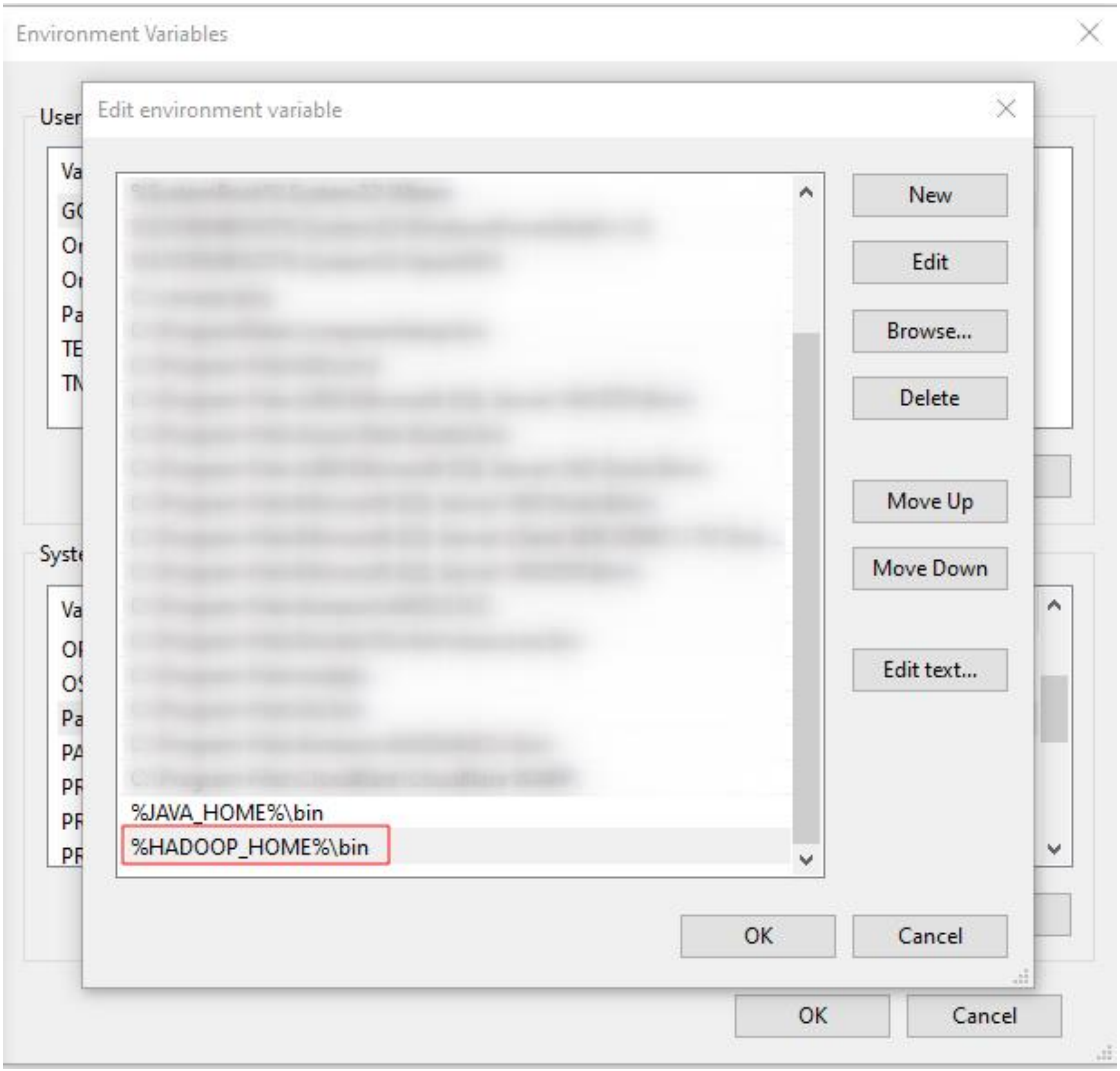
Edit System Variable

Variable name: HADOOP_HOME

Variable value: C:\hadoop

System variables

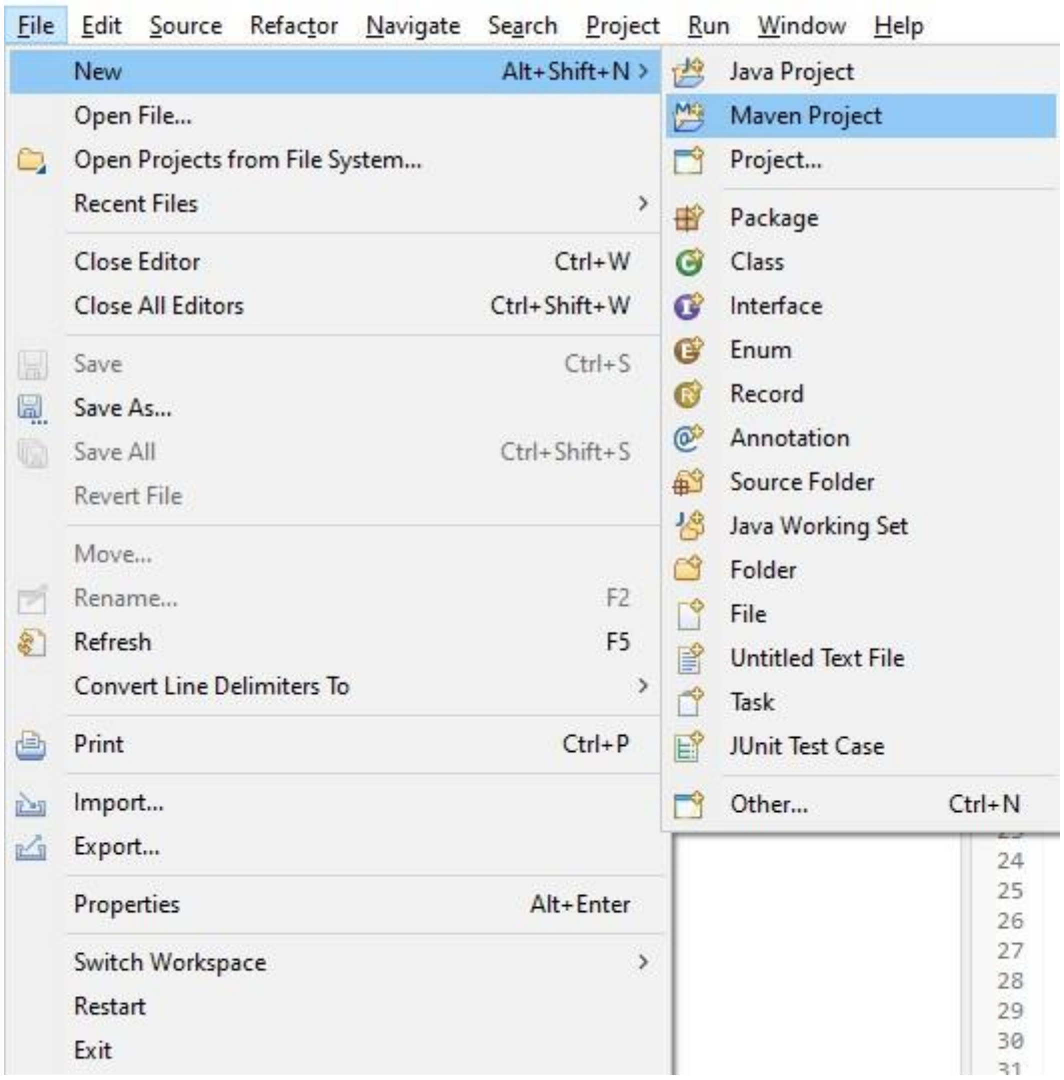
Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
HADOOP_HOME	C:\hadoop
JAVA_HOME	C:\Progra~1\Java\jdk-17
NUMBER_OF_PROCESSORS	8
OPENSSL_CONF	C:\Program Files\PostgreSQL\psqlODBC\etc\openssl.cnf
OS	Windows_NT



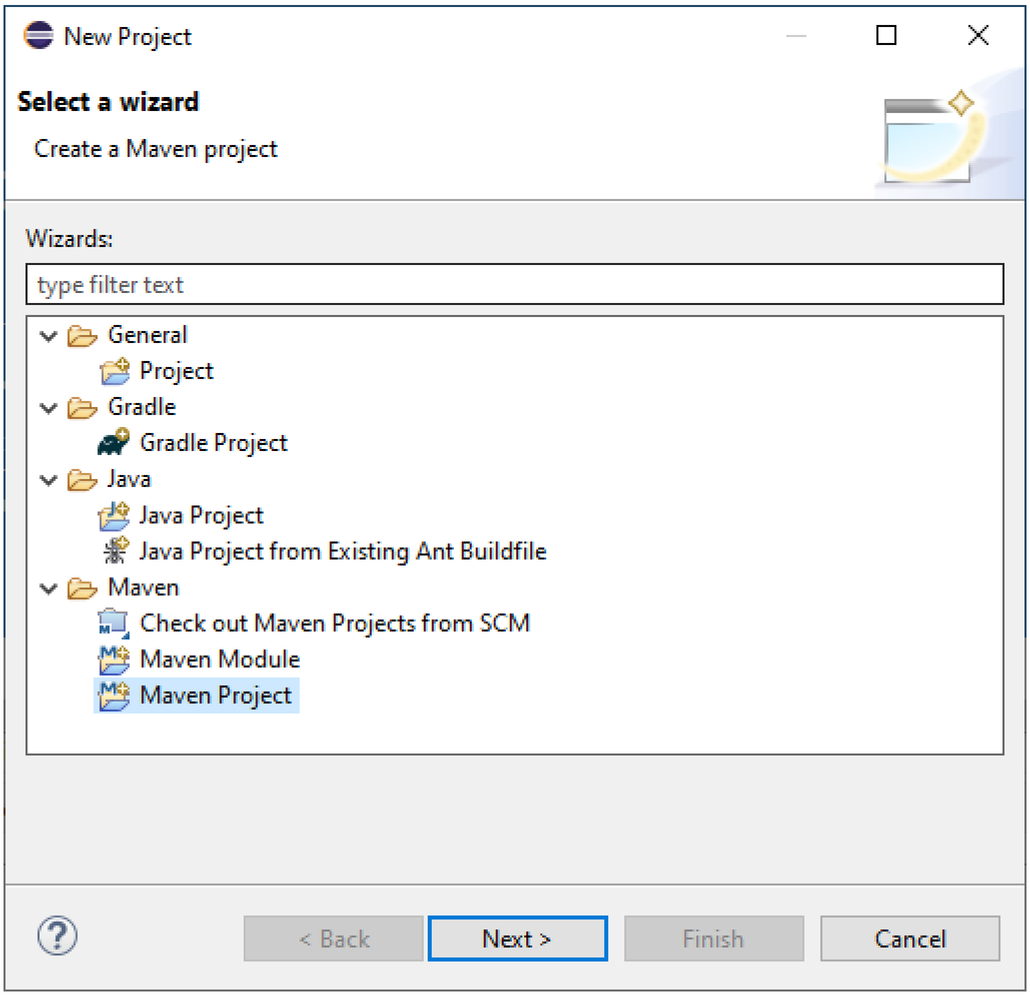
%JAVA_HOME%\bin
%HADOOP_HOME%\bin

2. Creating the project in Eclipse

Start Eclipse, and create a new project. Then select Maven Project:



OR,


New Project

Select a wizard

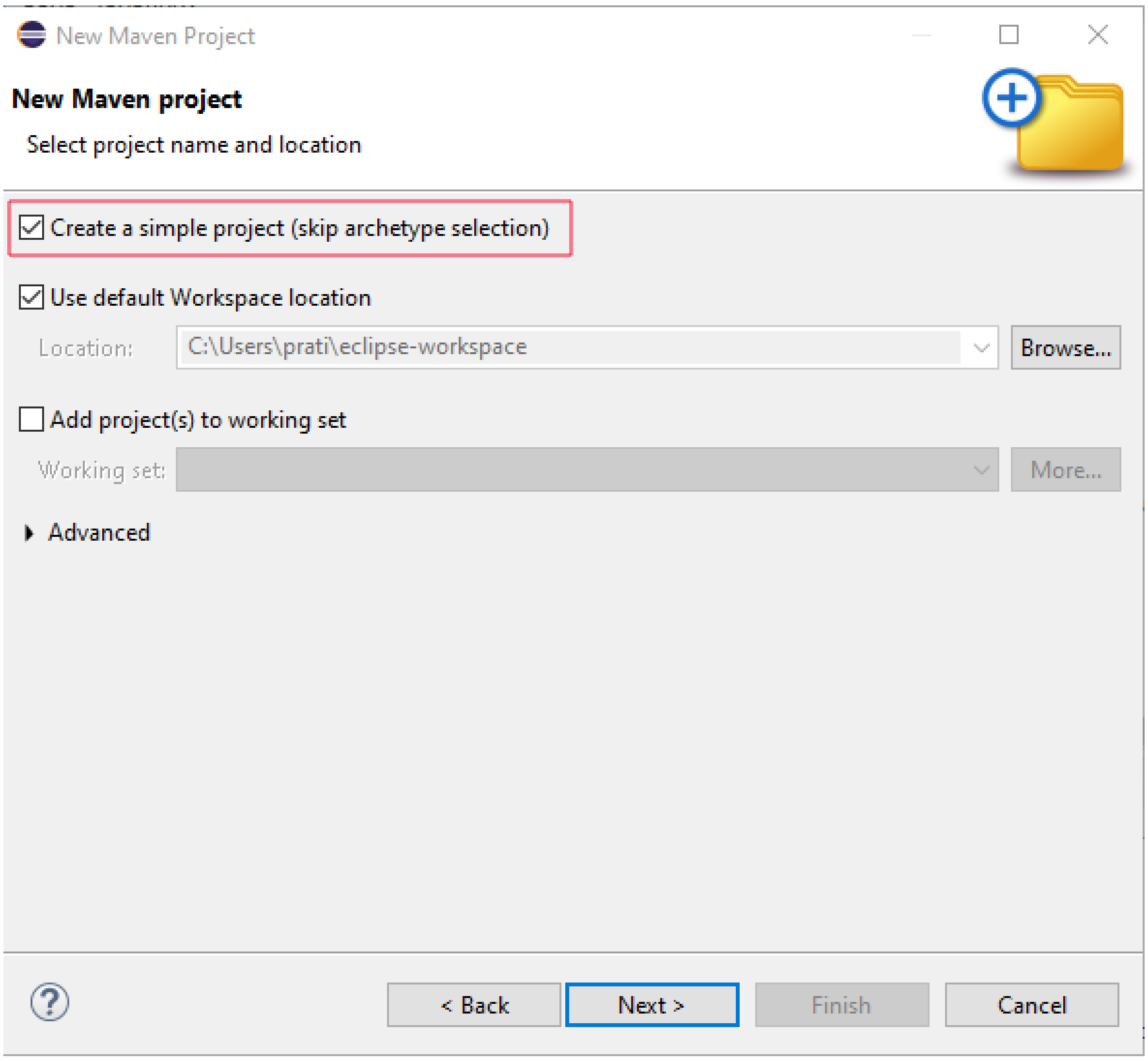
Create a Maven project

Wizards:

type filter text

- General
  - Project
- Gradle
  - Gradle Project
- Java
  - Java Project
  - Java Project from Existing Ant Buildfile
- Maven
  - Check out Maven Projects from SCM
  - Maven Module
  - Maven Project

Then on the next page, make sure that the "simple project" is checked:



New Maven project

Select project name and location

☑ Create a simple project (skip archetype selection)

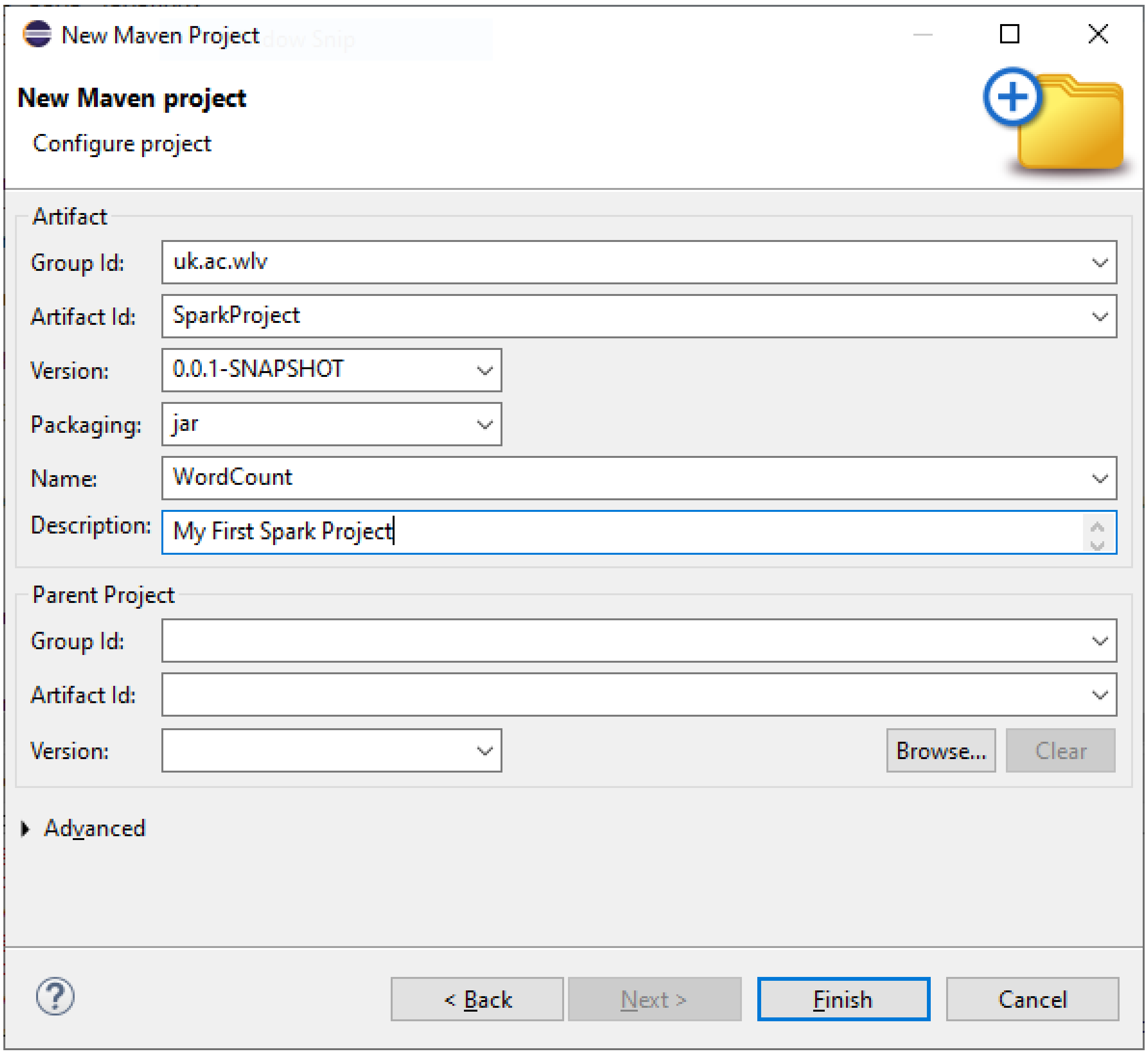
☑ Use default Workspace location

Location: C:\Users\prati\eclipse-workspace

☐ Add project(s) to working set

Working set:

Then fill out the project information as follows:


New Maven Project

New Maven project

Configure project

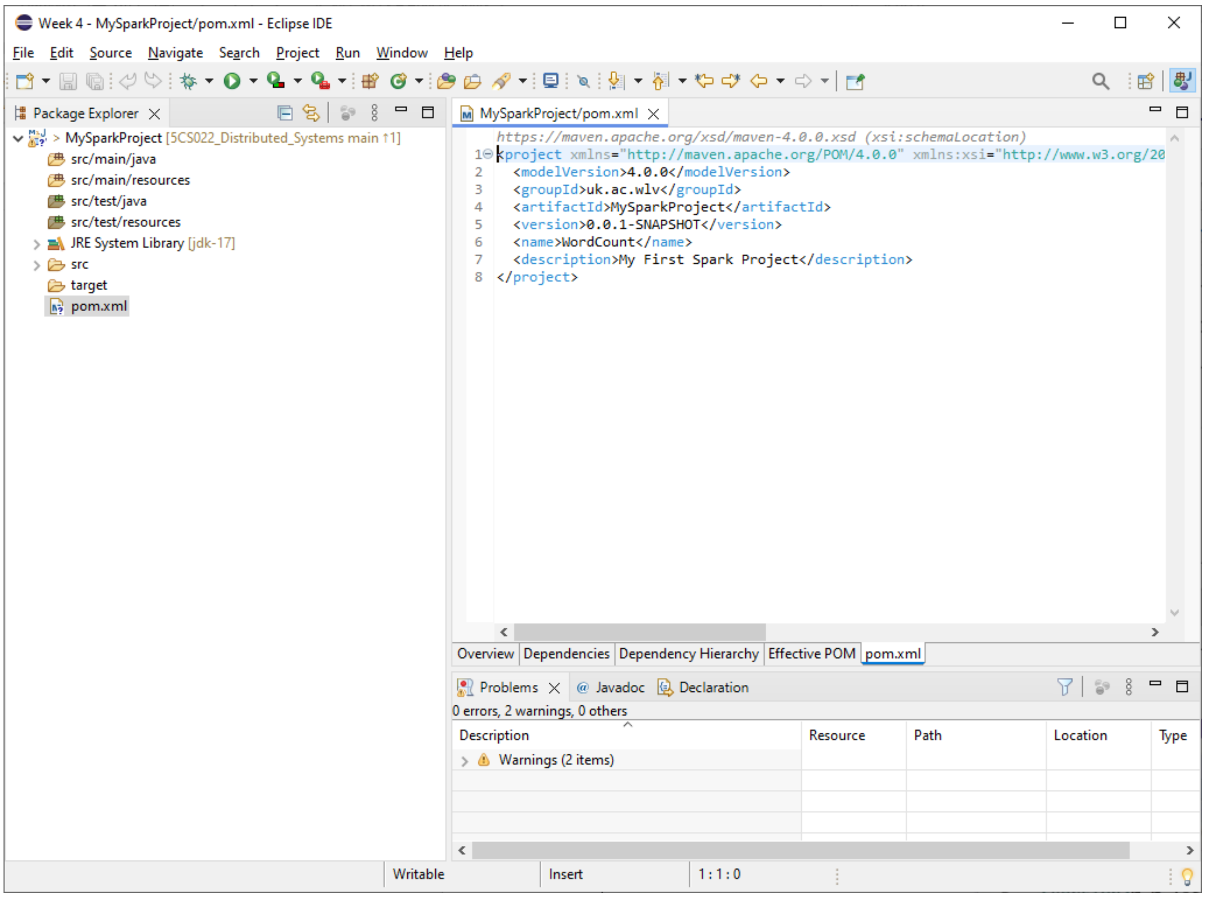
Artifact

Group Id:	uk.ac.wlv
Artifact Id:	SparkProject
Version:	0.0.1-SNAPSHOT
Packaging:	jar
Name:	WordCount
Description:	My First Spark Project

Parent Project

Group Id:	
Artifact Id:	
Version:	

When Eclipse has finished creating the project, open the "pom.xml" (This is the Maven project configuration file).


Week 4 - MySparkProject/pom.xml - Eclipse IDE
File Edit Source Navigate Search Project Run Window Help
Package Explorer
MySparkProject [5CS022_Distributed_Systems main ↑1]
src/main/java
src/main/resources
src/test/java
src/test/resources
JRE System Library [jdk-17]
src
target
pom.xml
MySparkProject/pom.xml
Overview Dependencies Dependency Hierarchy Effective POM pom.xml
Problems Javadoc Declaration
Description Resource Path Location Type
Warnings (2 items)
Writable Insert 1:1:0

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/20
  <modelVersion>4.0.0</modelVersion>
  <groupId>uk.ac.wlv</groupId>
  <artifactId>MySparkProject</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>WordCount</name>
  <description>My First Spark Project</description>
</project>
```

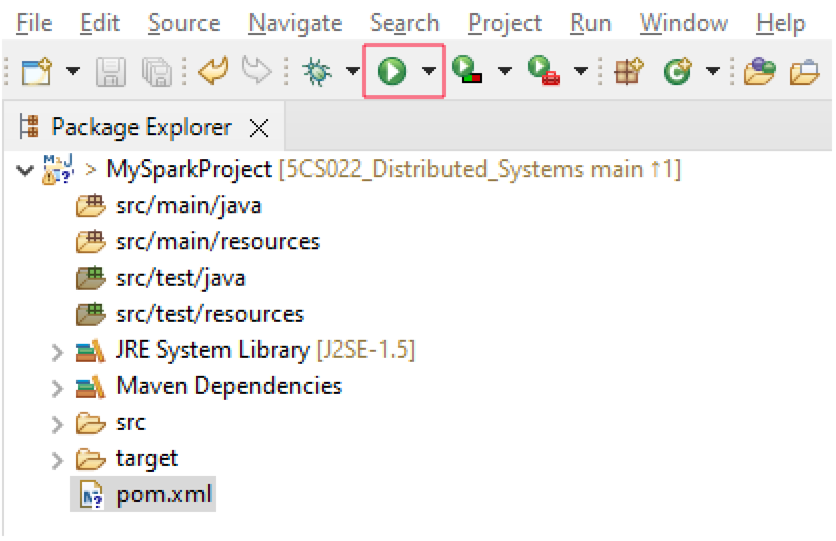
0 errors, 2 warnings, 0 others

Replace it with the following:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>uk.ac.wlv</groupId>
  <artifactId>MySparkProject</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>WordCount</name>
  <description>My First Spark Project</description>

  <dependencies>
    <dependency>
           <groupId>org.apache.spark</groupId>
           <artifactId>spark-core_2.13</artifactId>
      <version>3.5.1</version>
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
           <artifactId>maven-compiler-plugin</artifactId>
```

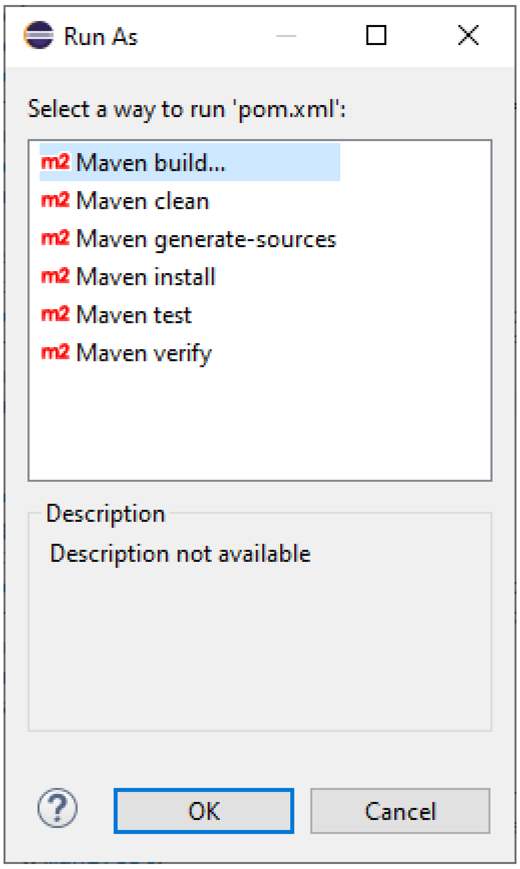
```
          <version>3.13.0</version>
            <configuration>
                  <source>${java.version}</source>
                  <target>${java.version}</target>
            </configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
          <artifactId>maven-jar-plugin</artifactId>
          <version>3.3.0</version>
        <configuration>
          <archive>
            <manifest>
              <addClasspath>true</addClasspath>
              <classpathPrefix>lib/</classpathPrefix>
              <mainClass>uk.ac.wlv.WordCount</mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-dependency-plugin</artifactId>
        <executions>
          <execution>
            <id>copy</id>
            <phase>install</phase>
            <goals>
              <goal>copy-dependencies</goal>
            </goals>
            <configuration>
              <outputDirectory>${project.build.directory}/lib</outputDirectory>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>
</project>
```

Save the file and then click the "run" button on the toolbar to update the configuration in Eclipse:

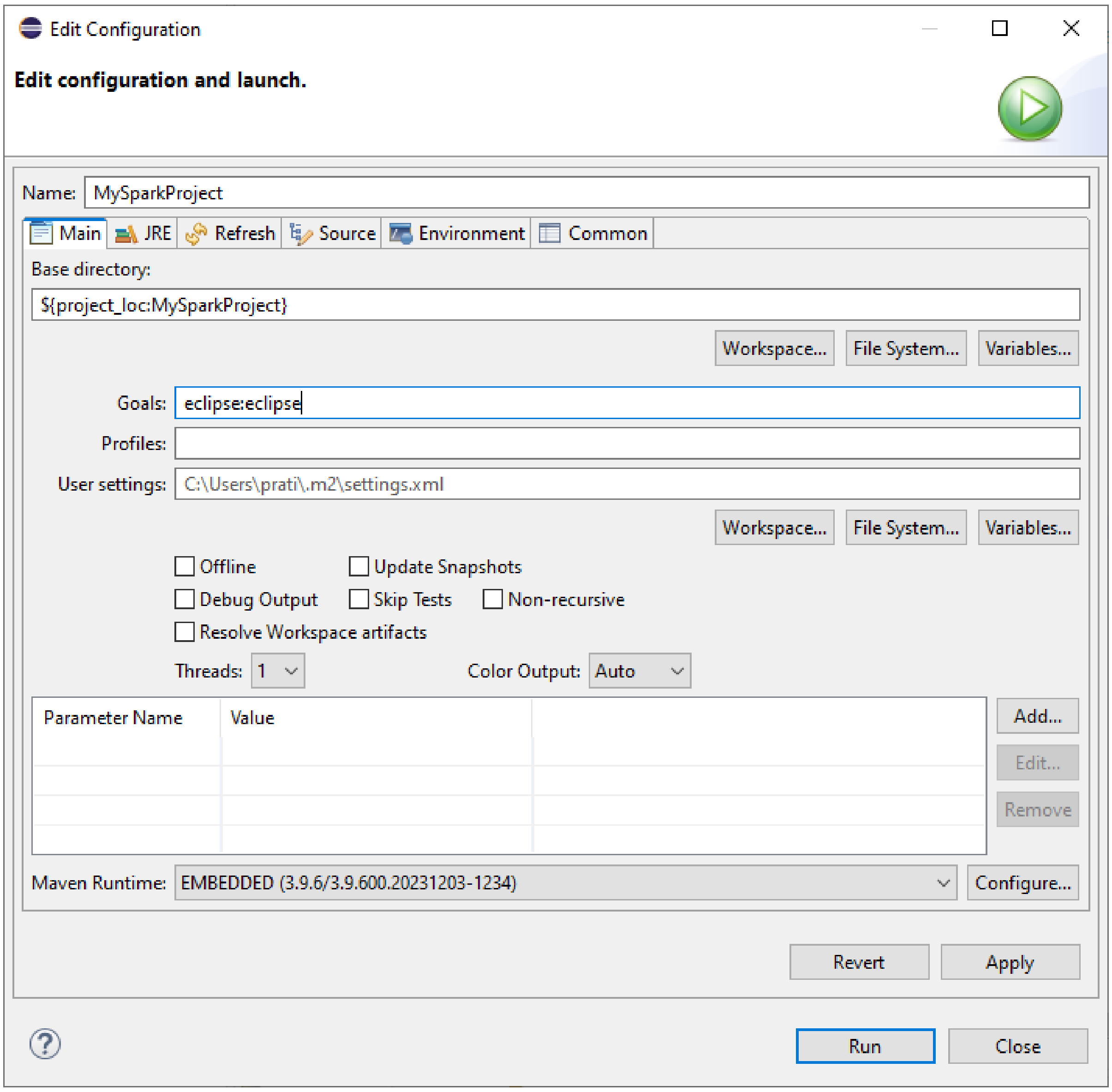


```
MySparkProject [5CS022_Distributed_Systems main ↑1]
    src/main/java
    src/main/resources
    src/test/java
    src/test/resources
    JRE System Library [J2SE-1.5]
    Maven Dependencies
    src
    target
    pom.xml
```

Then select the first "Maven build":



Then enter "eclipse:eclipse" for the "Goals" and click the Run button:


Edit Configuration

Edit configuration and launch.

Name: MySparkProject

Base directory:
${project_loc:MySparkProject}

Goals: eclipse:eclipse

Profiles:

User settings: C:\Users\prati\.m2\settings.xml

☐ Offline ☐ Update Snapshots
☐ Debug Output ☐ Skip Tests ☐ Non-recursive
☐ Resolve Workspace artifacts

Threads: 1 Color Output: Auto

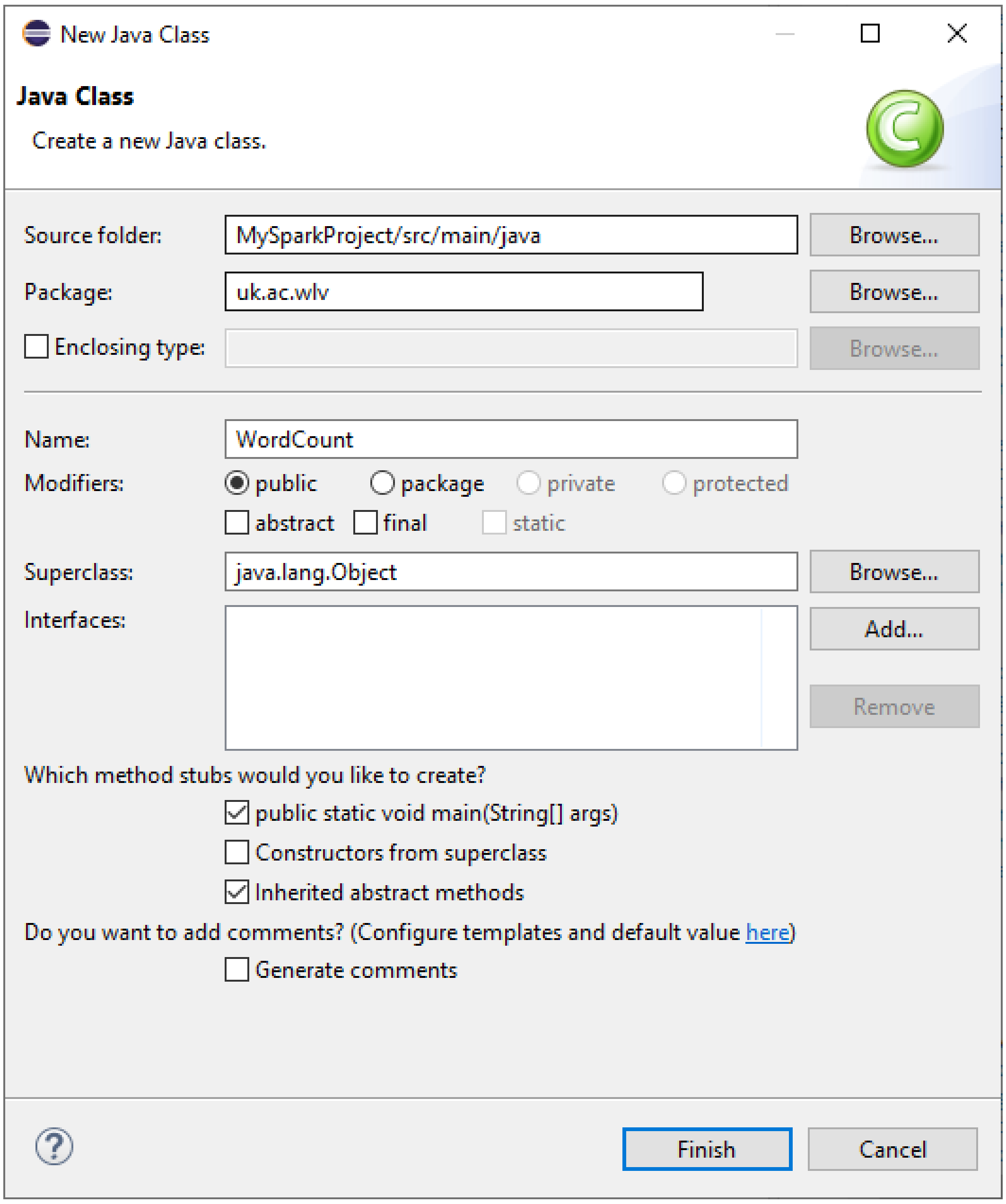
Parameter Name	Value

Maven Runtime: EMBEDDED (3.9.6/3.9.600.20231203-1234)

This will synchronise the Maven build configuration with Eclipse and bring Eclipse up to date.

3. Creating the WordCount program

Right click on the src/main folder in Eclipse and create a new Java class:


New Java Class

Java Class

Create a new Java class.

Source folder: MySparkProject/src/main/java

Package: uk.ac.wlv

☐ Enclosing type:

Name: WordCount

Modifiers: [✓ public] package private protected

☐ abstract ☐ final ☐ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

☑ public static void main(String[] args)

☐ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments

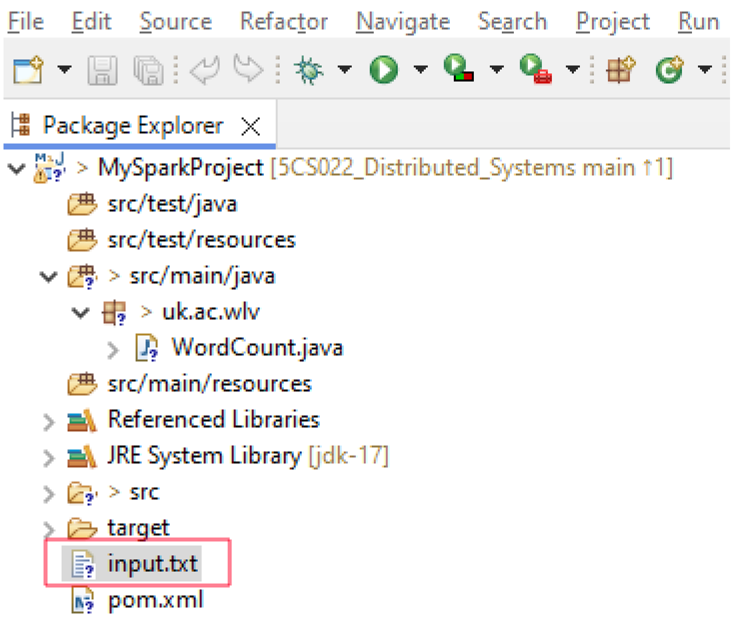
Enter "uk.ac.wlv" for the package, and WordCount for the Name, and click finish.

Then replace the WordCount.java code with the following:

```
package uk.ac.wlv;

import java.io.IOException;
import java.util.Arrays;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;
import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContext;
import scala.Tuple2;

public class WordCount {

    public static void main(String[] args) {
        SparkConf sparkConf = new SparkConf();
        sparkConf.setAppName("Spark WordCount example using Java");
        /* Tell Spark that we are running on this computer alone */
        sparkConf.setMaster("local");

        JavaSparkContext sparkContext = new JavaSparkContext(sparkConf);

        /* Reading input file */
        JavaRDD < String > textFile = sparkContext.textFile("input.txt");

        /* This code snippet creates an RDD (Resilient Distributed Dataset) of
words from each line of the input file and the flatMap function is used to split
the text file into an ArrayList of words by applying the split(" ") method to
each line, which separates the line into individual words. */
        JavaRDD < String > words = textFile.flatMap(l -> Arrays.asList(l.split("
")).iterator());

        /* Generate Pair of Word with count */
        JavaPairRDD < String, Integer > pairs = words.mapToPair(w -> new
Tuple2<String, Integer>(w, 1));

        /* Aggregate Pairs of Same Words with count */
        JavaPairRDD < String, Integer > counts = pairs.reduceByKey((x, y) -> x +
y);


        /* Deleting output directory if it already exists and saving the result
file */
        String outputPath = "output"; // Change this to your desired output
directory
        try {
            FileSystem.get(sparkContext.hadoopConfiguration()).delete(new
Path(outputPath), true);
        } catch (IOException e) {
            e.printStackTrace();
        }

        /* Saving the result file */
        try {
            counts.saveAsTextFile(outputPath);
        } catch (Exception e) {
            e.printStackTrace();
        }

        /* System.out.println(counts.collect()); */
        System.out.println("Word Counts:");

        for (Tuple2<String, Integer> tuple : counts.collect()) {
            System.out.println(tuple._1() + ": " + tuple._2());
        }
        sparkContext.stop();
        sparkContext.close();
    }
}
```

4. Creating the input text file

Using Notepad (or similar), create a file called "input.txt" and fill it with some example English text. Save it in the same directory as the pom.xml file, that is, the project directory.

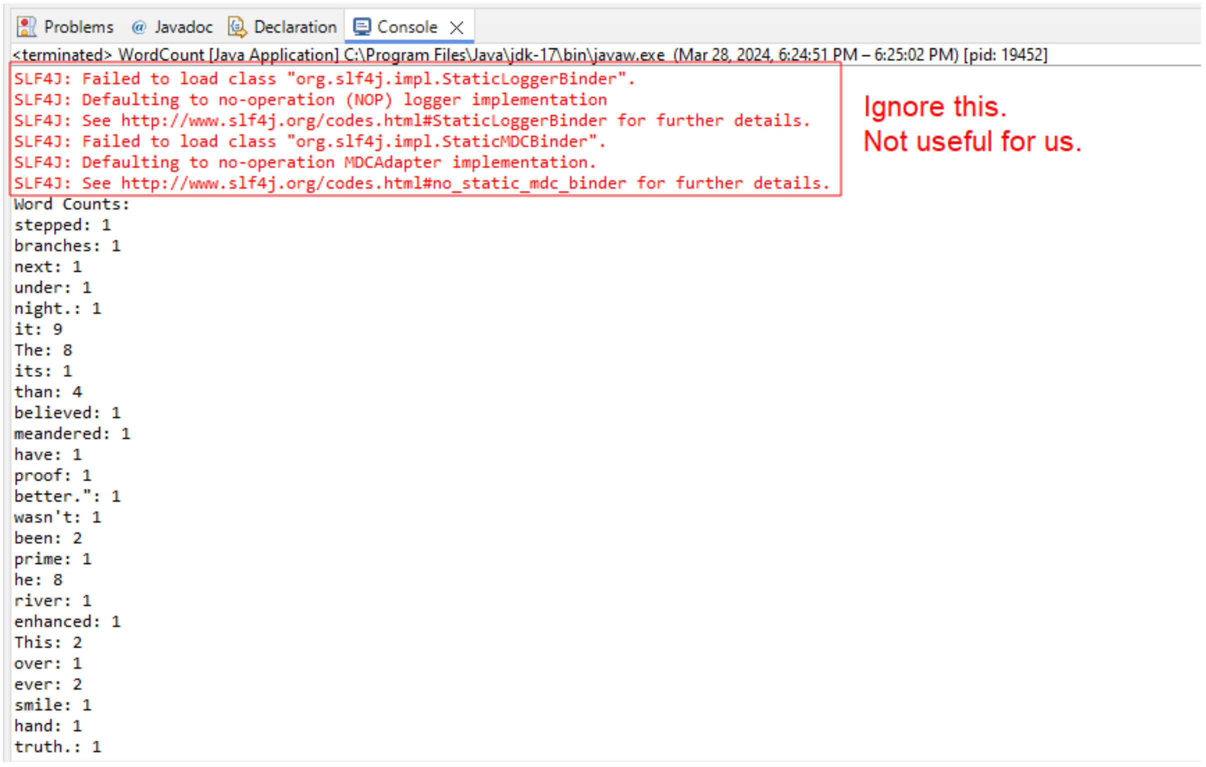


```
MySparkProject [5CS022_Distributed_Systems main ↑1]
  src/test/java
  src/test/resources
  > src/main/java
    > uk.ac.wlv
      WordCount.java
  src/main/resources
  Referenced Libraries
  JRE System Library [jdk-17]
  > src
  target
  input.txt
  pom.xml
```

5. Running the WordCount Spark program

Make sure that the file WordCount.java is opened as the current file in the Eclipse editor and click the run button on the Eclipse menu toolbar.

If it runs successfully, you should see the Spark logging output in Eclipse:



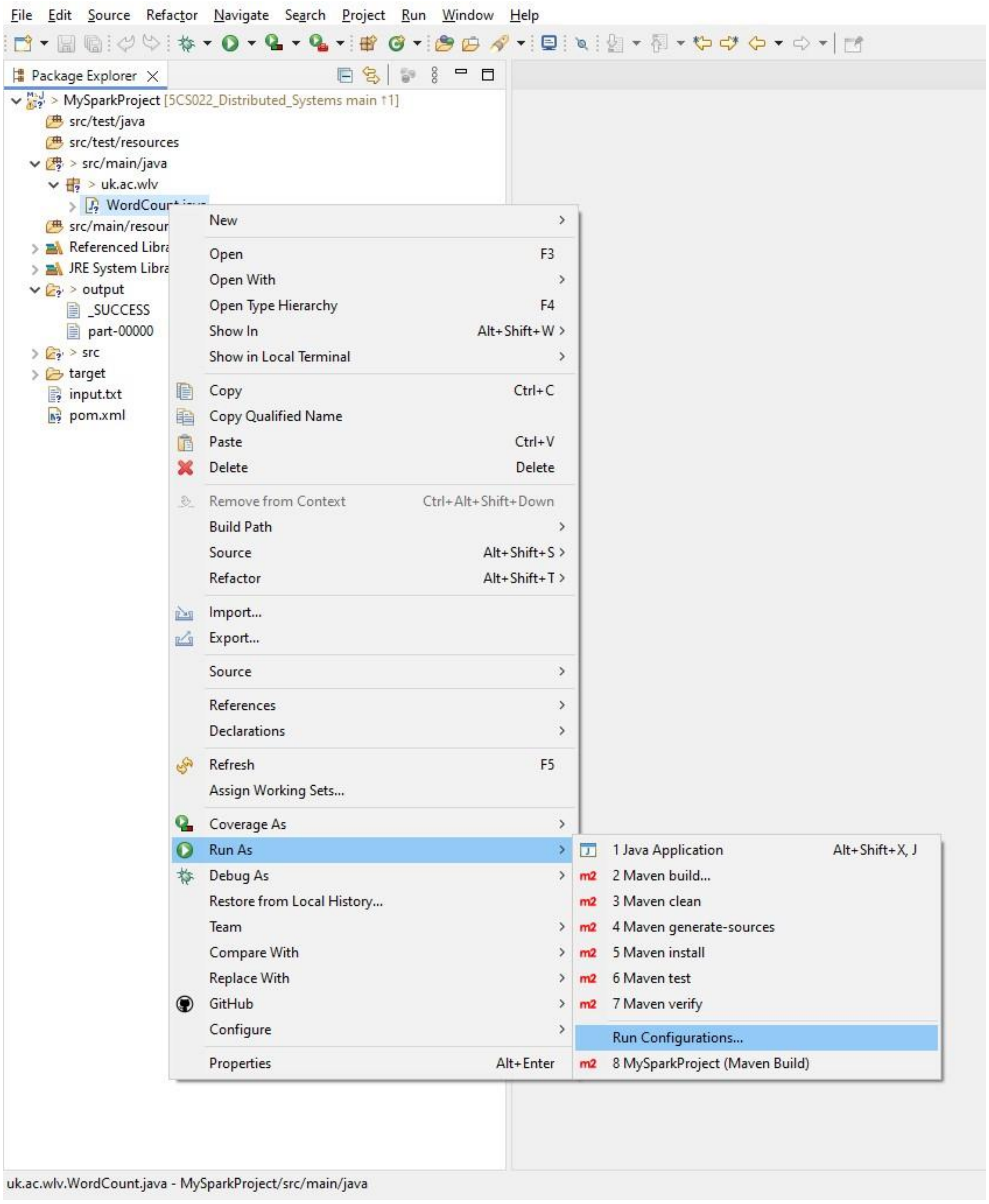
```
<terminated> WordCount [Java Application] C:\Program Files\Java\jdk-17\bin\javaw.exe (Mar 28, 2024, 6:24:51 PM – 6:25:02 PM) [pid: 19452]
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
SLF4J: Failed to load class "org.slf4j.impl.StaticMDCBinder".
SLF4J: Defaulting to no-operation MDCAdapter implementation.
SLF4J: See http://www.slf4j.org/codes.html#no_static_mdc_binder for further details.
Word Counts:
stepped: 1
branches: 1
next: 1
under: 1
night.: 1
it: 9
The: 8
its: 1
than: 4
believed: 1
meandered: 1
have: 1
proof: 1
better.": 1
wasn't: 1
been: 2
prime: 1
he: 8
river: 1
enhanced: 1
This: 2
over: 1
ever: 2
smile: 1
hand: 1
truth.: 1
```

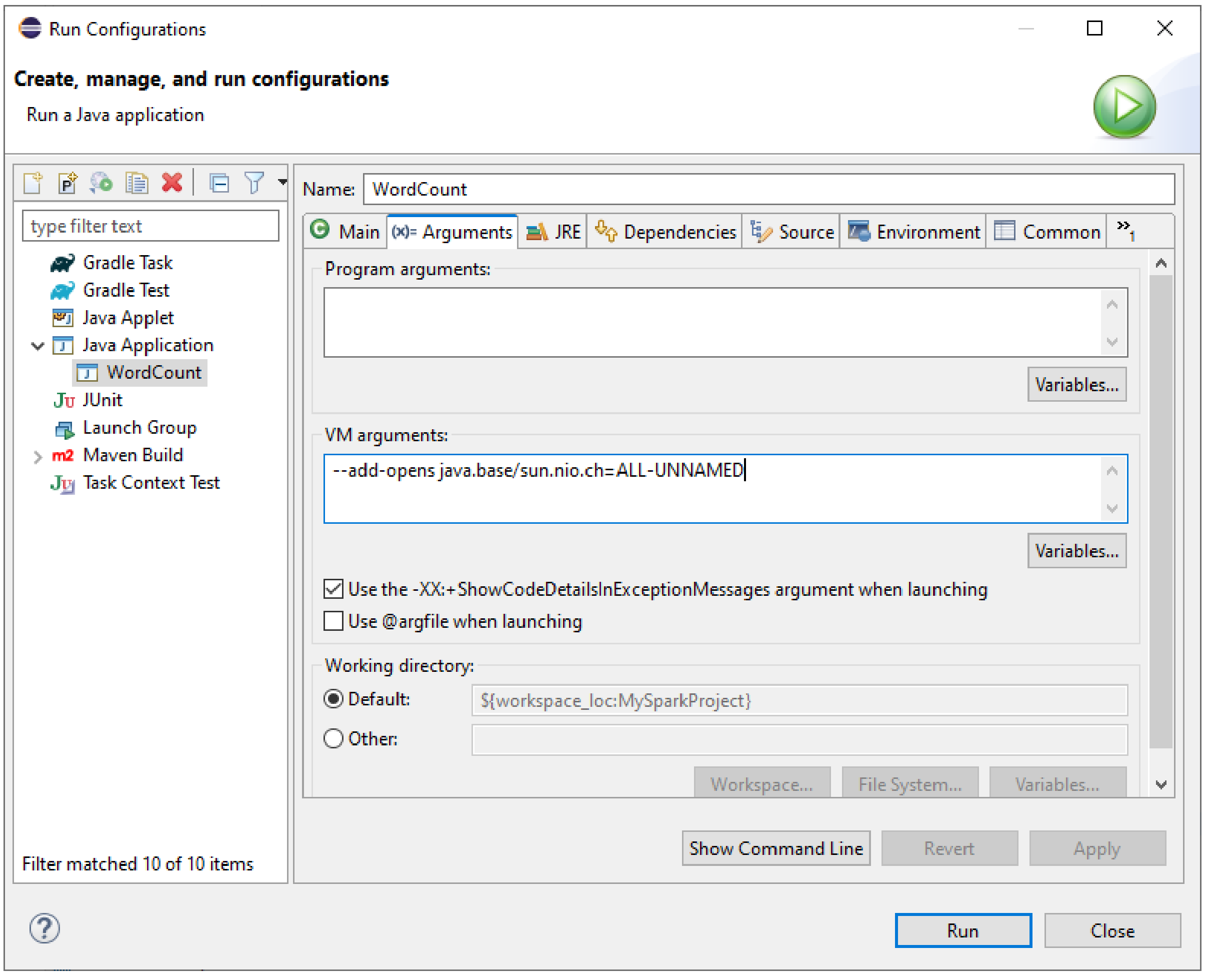
and there shouldn't be any error messages.

6. Debugging ERRORS

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further
details.
Exception in thread "main" java.lang.IllegalAccessError: class
org.apache.spark.storage.StorageUtils$ (in unnamed module @0x6a03bcb1) cannot
access class sun.nio.ch.DirectBuffer (in module java.base) because module
java.base does not export sun.nio.ch to unnamed module @0x6a03bcb1
	at org.apache.spark.storage.StorageUtils$.<clinit>(StorageUtils.scala:213)
	at
org.apache.spark.storage.BlockManagerMasterEndpoint.<init>(BlockManagerMasterEndp
oint.scala:121)
	at org.apache.spark.SparkEnv$.$anonfun$create$9(SparkEnv.scala:358)
	at
org.apache.spark.SparkEnv$.registerOrLookupEndpoint$1(SparkEnv.scala:295)
	at org.apache.spark.SparkEnv$.create(SparkEnv.scala:344)
	at org.apache.spark.SparkEnv$.createDriverEnv(SparkEnv.scala:196)
	at org.apache.spark.SparkContext.createSparkEnv(SparkContext.scala:284)
	at org.apache.spark.SparkContext.<init>(SparkContext.scala:483)
	at
org.apache.spark.api.java.JavaSparkContext.<init>(JavaSparkContext.scala:58)
	at uk.ac.wlv.WordCount.main(WordCount.java:21)
```

If you ran into this issue. Right-click on the Java file >> “Run As” >> click on the run configurations.


File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
MySparkProject [5CS022_Distributed_Systems main ↑1]
src/test/java
src/test/resources
src/main/java
uk.ac.wlv
WordCount.java
src/main/resour
Referenced Libra
JRE System Libra
output
_SUCCESS
part-00000
src
target
input.txt
pom.xml
New
Open F3
Open With
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Remove from Context Ctrl+Alt+Shift+Down
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Source
References
Declarations
Refresh F5
Assign Working Sets...
Coverage As
Run As
Debug As
Restore from Local History...
Team
Compare With
Replace With
GitHub
Configure
Properties Alt+Enter
1 Java Application Alt+Shift+X, J
2 Maven build...
3 Maven clean
4 Maven generate-sources
5 Maven install
6 Maven test
7 Maven verify
Run Configurations...
8 MySparkProject (Maven Build)
uk.ac.wlv.WordCount.java - MySparkProject/src/main/java


Run Configurations

Create, manage, and run configurations

Run a Java application

Name: WordCount

Program arguments:

VM arguments:

```
--add-opens java.base/sun.nio.ch=ALL-UNNAMED
```

[✓] Use the -XX:+ShowCodeDetailsInExceptionMessages argument when launching

[ ] Use @argfile when launching

Working directory:

(●) Default: ${workspace_loc:MySparkProject}

( ) Other:

Filter matched 10 of 10 items

On the Arguments tab, add the following to the VM arguments:

--add-opens java.base/sun.nio.ch=ALL-UNNAMED

If you still run into an issue:

Right-click on **winutils.exe** file in the C:/hadoop/bin directory and "Run as Administrator". You may come across this issue.

The code execution cannot proceed because MSVCR100.dll was not found. Reinstalling the program may fix this problem.

If error appears, follow the link below to download the Microsoft Visual C++ 2010 Service Pack 1 Redistributable Package. msvcr100.dll is a part of Microsoft Visual C++ and is required to run programs developed with Visual C++.

https://www.microsoft.com/en-us/download/details.aspx?id=26999

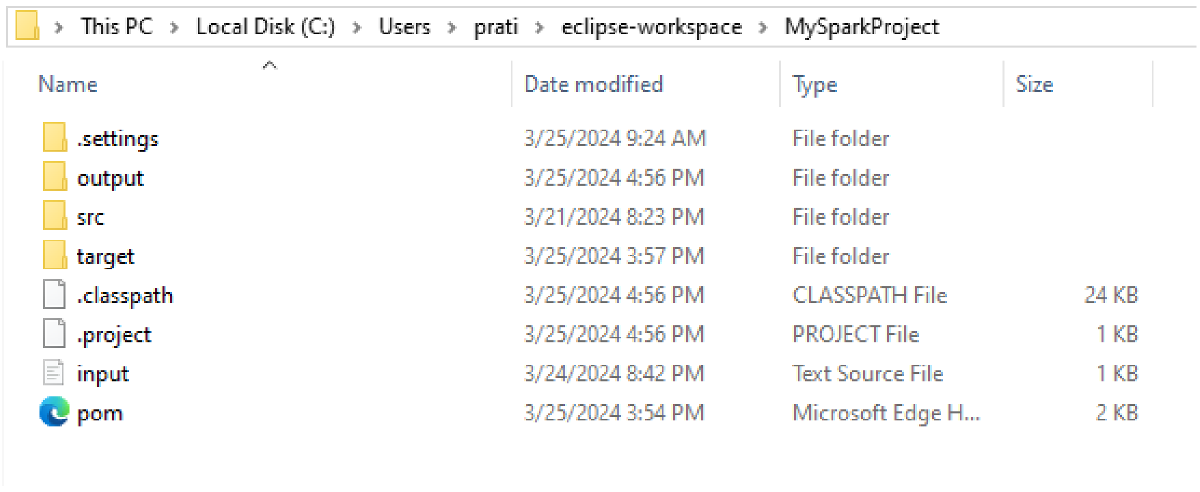
Even if you still receive the error message related to the Hadoop home not found or doesn't exist, add the following to the VM arguments of your Java file.

```
--add-opens java.base/sun.nio.ch=ALL-UNNAMED
-Dhadoop.home.dir=C:/hadoop
-Djava.library.path=C:/hadoop/bin
```

The highlighted code is required to be added only if you have the error related to the Hadoop home path.

6. View the output results

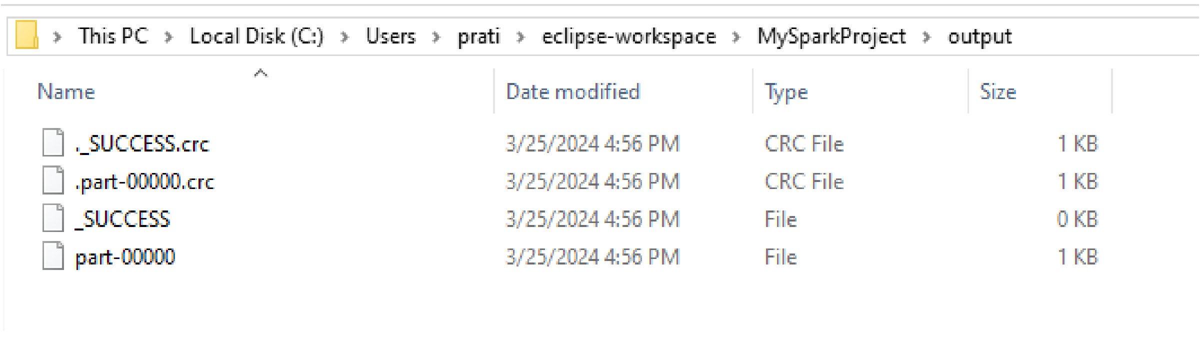
Assuming that your Spark program ran correctly, it would have created an output directory in your project directory:



This PC > Local Disk (C:) > Users > prati > eclipse-workspace > MySparkProject

Name	Date modified	Type	Size
.settings	3/25/2024 9:24 AM	File folder	
output	3/25/2024 4:56 PM	File folder	
src	3/21/2024 8:23 PM	File folder	
target	3/25/2024 3:57 PM	File folder	
.classpath	3/25/2024 4:56 PM	CLASSPATH File	24 KB
.project	3/25/2024 4:56 PM	PROJECT File	1 KB
input	3/24/2024 8:42 PM	Text Source File	1 KB
pom	3/25/2024 3:54 PM	Microsoft Edge H...	2 KB

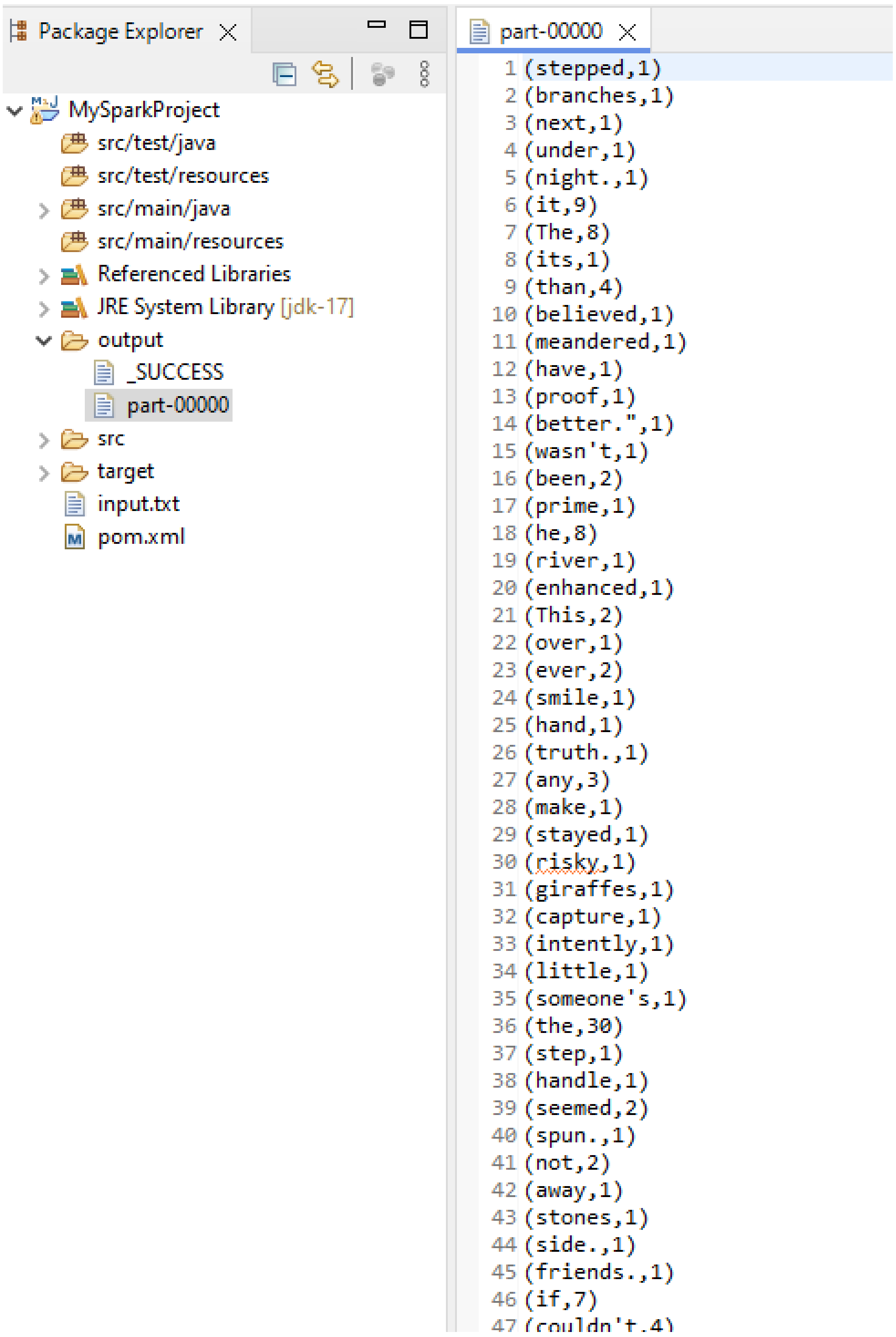
Opening the "output" directory should give you the following files:



This PC > Local Disk (C:) > Users > prati > eclipse-workspace > MySparkProject > output

Name	Date modified	Type	Size
._SUCCESS.crc	3/25/2024 4:56 PM	CRC File	1 KB
.part-00000.crc	3/25/2024 4:56 PM	CRC File	1 KB
_SUCCESS	3/25/2024 4:56 PM	File	0 KB
part-00000	3/25/2024 4:56 PM	File	1 KB

The results that we are looking for will be in the file "part-0000". Open that file in Notepad or any other text editors, and you should see a list of words and their counts.



```
(stepped,1)
(branches,1)
(next,1)
(under,1)
(night.,1)
(it,9)
(The,8)
(its,1)
(than,4)
(believed,1)
(meandered,1)
(have,1)
(proof,1)
(better.",1)
(wasn't,1)
(been,2)
(prime,1)
(he,8)
(river,1)
(enhanced,1)
(This,2)
(over,1)
(ever,2)
(smile,1)
(hand,1)
(truth.,1)
(any,3)
(make,1)
(stayed,1)
(risky,1)
(giraffes,1)
(capture,1)
(intently,1)
(little,1)
(someone's,1)
(the,30)
(step,1)
(handle,1)
(seemed,2)
(spun.,1)
(not,2)
(away,1)
(stones,1)
(side.,1)
(friends.,1)
(if,7)
(couldn't,4)
```

Tasks

1. Create a Spark program to count letters instead of words.