Practical Journal
Big Data Analytics

Submitted By: Mohammad Tabish Mukaddam
Submitted To: Prof. Swati Maurya

Seat: 31031523015
MSc CS-II
A.Y. 2024-25

Department of Computer Science
Somaiya Vidyavihar University
SK Somaiya College

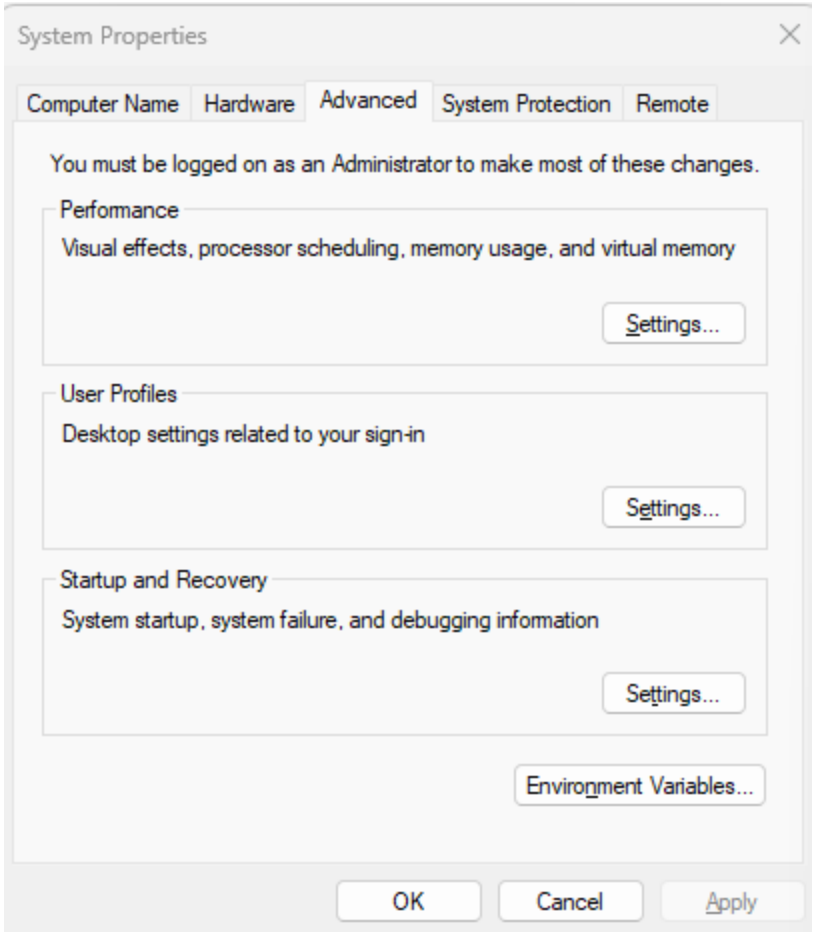
Sr. No.	Title
1	Installation of Hadoop
2	Spark SQL
3	Spark GraphX
4	PySpark
5	Download and installation of HBase

Practical 1:

Aim: Installation of Hadoop

Step 1: Download Binary File for Windows https://hadoop.apache.org/releases.html

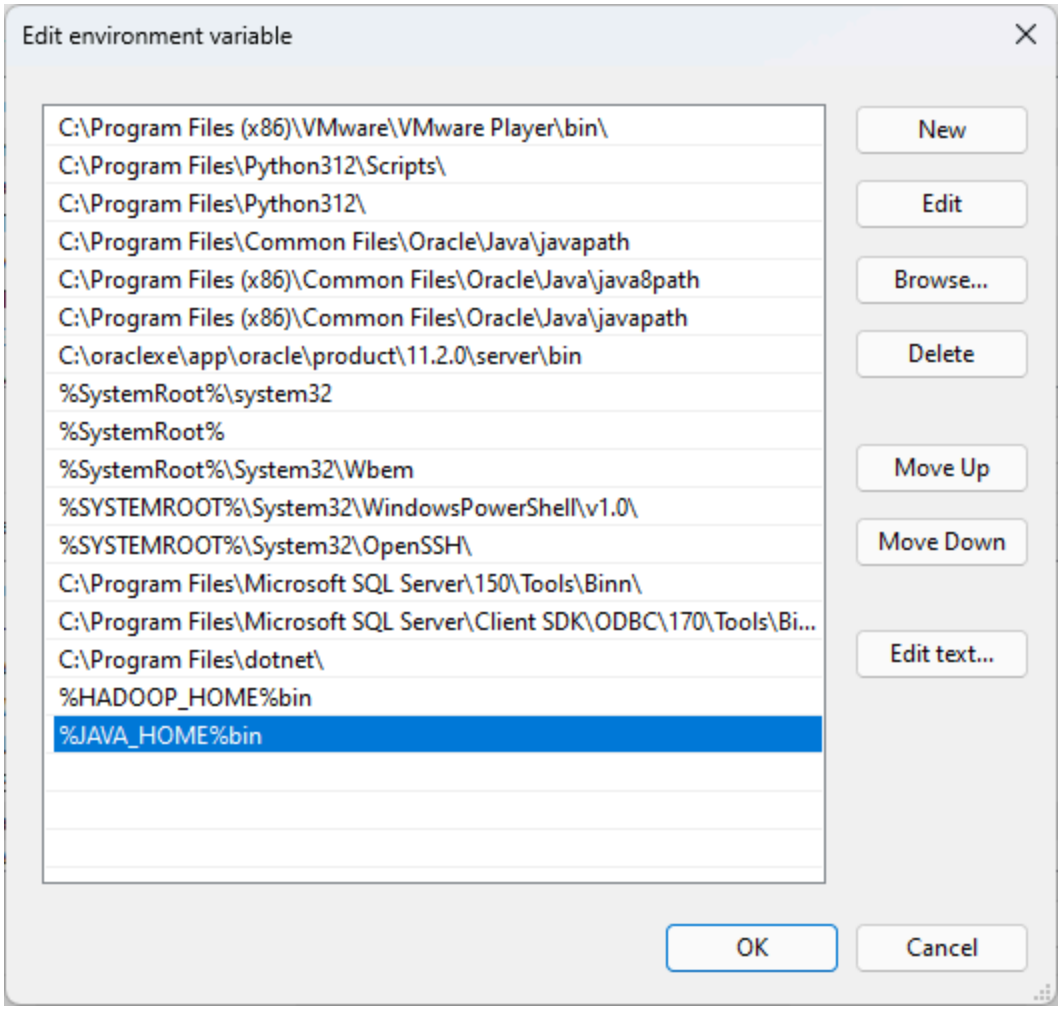
Step 2: Extract the files in C drive .
Step 3: Edit Environment Variables.



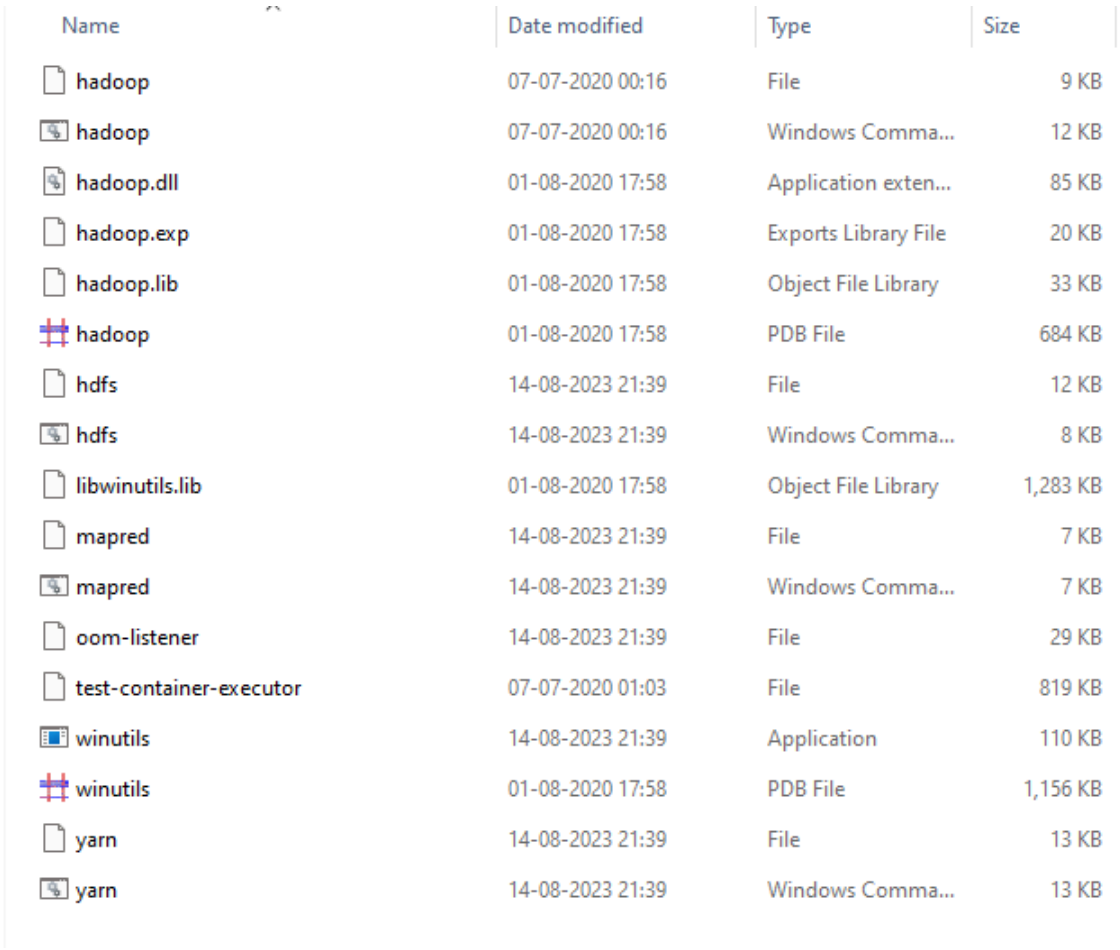
Step 4: Under System Variables click “New” and set “Variable name” as JAVA_HOME and “Variable value” as the path of your JAVA JDK.

Step 5: Similarly add “HADOOP_HOME” variable and download the bin folder from the below link

https://drive.google.com/drive/folders/1iURNbow2IglhAhSy3sfY5xxVfAg33NBW



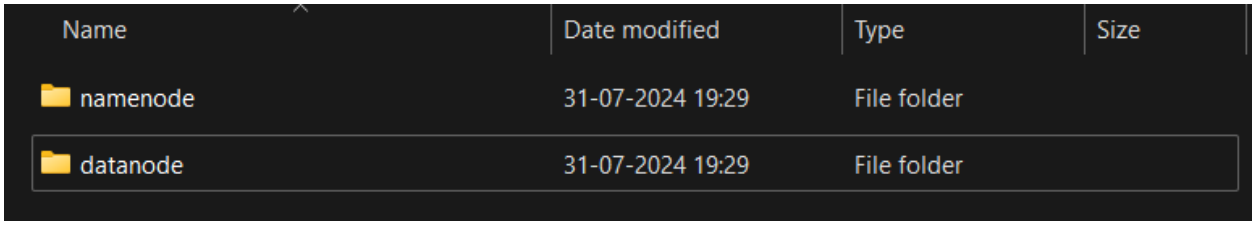
Step 6: Extract the bin archive and replace the bin folder in Hadoop folder with the bin folder in this archive.



Name	Date modified	Type	Size
hadoop	07-07-2020 00:16	File	9 KB
hadoop	07-07-2020 00:16	Windows Comma...	12 KB
hadoop.dll	01-08-2020 17:58	Application exten...	85 KB
hadoop.exp	01-08-2020 17:58	Exports Library File	20 KB
hadoop.lib	01-08-2020 17:58	Object File Library	33 KB
hadoop	01-08-2020 17:58	PDB File	684 KB
hdfs	14-08-2023 21:39	File	12 KB
hdfs	14-08-2023 21:39	Windows Comma...	8 KB
libwinutils.lib	01-08-2020 17:58	Object File Library	1,283 KB
mapred	14-08-2023 21:39	File	7 KB
mapred	14-08-2023 21:39	Windows Comma...	7 KB
oom-listener	14-08-2023 21:39	File	29 KB
test-container-executor	07-07-2020 01:03	File	819 KB
winutils	14-08-2023 21:39	Application	110 KB
winutils	01-08-2020 17:58	PDB File	1,156 KB
yarn	14-08-2023 21:39	File	13 KB
yarn	14-08-2023 21:39	Windows Comma...	13 KB

Step 7: Check if “winutils” is working. If you get any dll error then download that dll and paste in the Windows -> System32 folder.

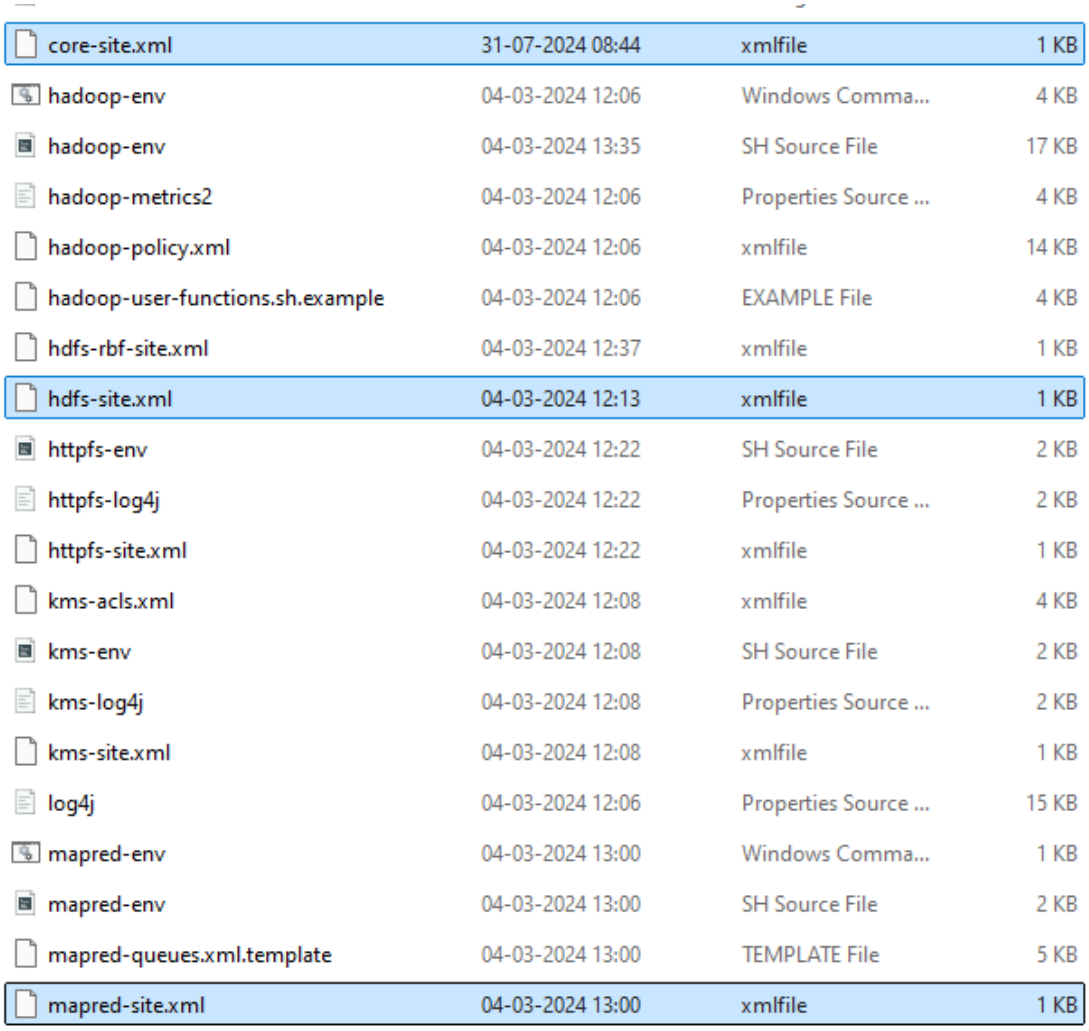
Step 8: Create a data folder in the hadoop home directory and add the folders datanode and namenode to it.



Name	Date modified	Type	Size
namenode	31-07-2024 19:29	File folder	
datanode	31-07-2024 19:29	File folder	

9: Add the following path to “Path” under “System Variables” in “Edit Environment Variables”
C:\hadoop-3.4.0\sbin

10: Make the changes to the following files as given, in “etc/hadoop” folder of hadoop home.



core-site.xml	31-07-2024 08:44	xmlfile	1 KB
hadoop-env	04-03-2024 12:06	Windows Comma...	4 KB
hadoop-env	04-03-2024 13:35	SH Source File	17 KB
hadoop-metrics2	04-03-2024 12:06	Properties Source ...	4 KB
hadoop-policy.xml	04-03-2024 12:06	xmlfile	14 KB
hadoop-user-functions.sh.example	04-03-2024 12:06	EXAMPLE File	4 KB
hdfs-rbf-site.xml	04-03-2024 12:37	xmlfile	1 KB
hdfs-site.xml	04-03-2024 12:13	xmlfile	1 KB
httpfs-env	04-03-2024 12:22	SH Source File	2 KB
httpfs-log4j	04-03-2024 12:22	Properties Source ...	2 KB
httpfs-site.xml	04-03-2024 12:22	xmlfile	1 KB
kms-acls.xml	04-03-2024 12:08	xmlfile	4 KB
kms-env	04-03-2024 12:08	SH Source File	2 KB
kms-log4j	04-03-2024 12:08	Properties Source ...	2 KB
kms-site.xml	04-03-2024 12:08	xmlfile	1 KB
log4j	04-03-2024 12:06	Properties Source ...	15 KB
mapred-env	04-03-2024 13:00	Windows Comma...	1 KB
mapred-env	04-03-2024 13:00	SH Source File	2 KB
mapred-queues.xml.template	04-03-2024 13:00	TEMPLATE File	5 KB
mapred-site.xml	04-03-2024 13:00	xmlfile	1 KB

core-site.xml

```
<configuration>
        <property>
                <name>fs.default.name</name>
                 <value>hdfs://localhost:9000</value>
        </property>
</configuration>
```

mapred-site.xml

```
<configuration>
        <property>
                <name>mapred.framework.name</name>
                <value>yarn</value>
```

```
        </property>
    </configuration>
```

hdfs-site.xml

```
    <configuration>
        <property>
            <name>dfs.replication</name>
            <value>1</value>
        </property>

        <property>
            <name>dfs.namenode.name.dir</name>
            <value>C:\hadoop-3.4.0\data\namenode</value>
        </property>

        <property>
            <name>dfs.datanode.data.dir</name>
            <value>C:\hadoop-3.4.0\data\datanode</value>
        </property>
    </configuration>
```

yarn-site.xml

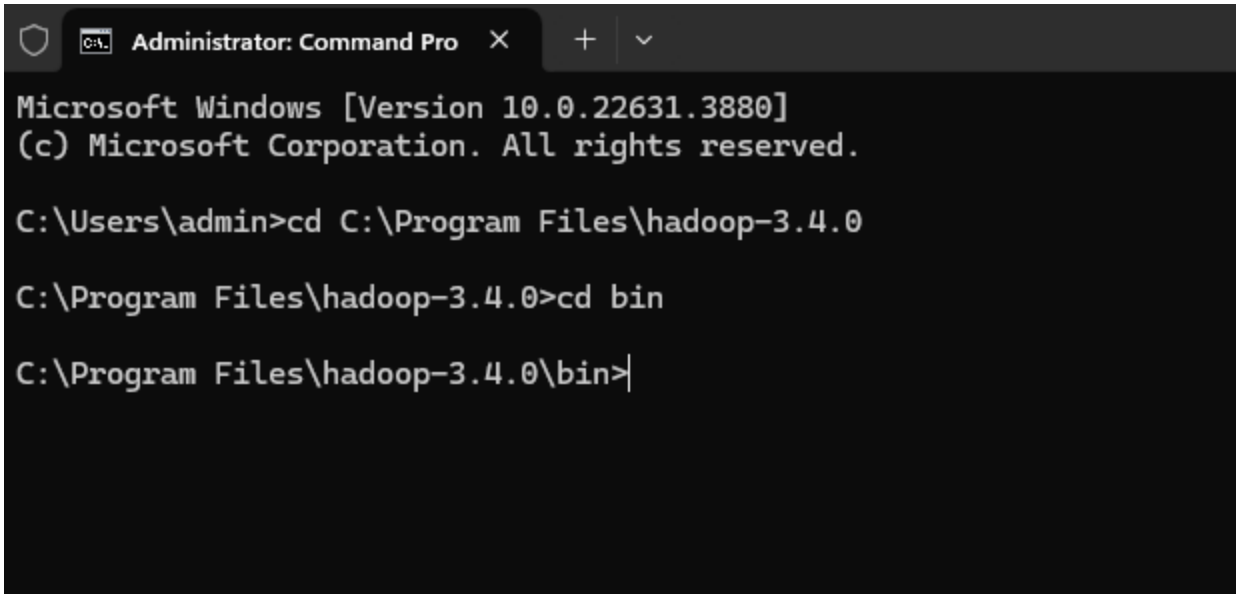
```
    <configuration>
        <property>
            <name>yarn.nodemanager.aux-services</name>
            <value>mapreduce_shuffle</value>
        </property>
        <property>
            <name>yarn.nodemanager.auxservice.mapreduce.shuffle.class</name>
            <value>org.apache.hadoop.mapred.shuffleHandler</value>
        </property>
    </configuration>
```

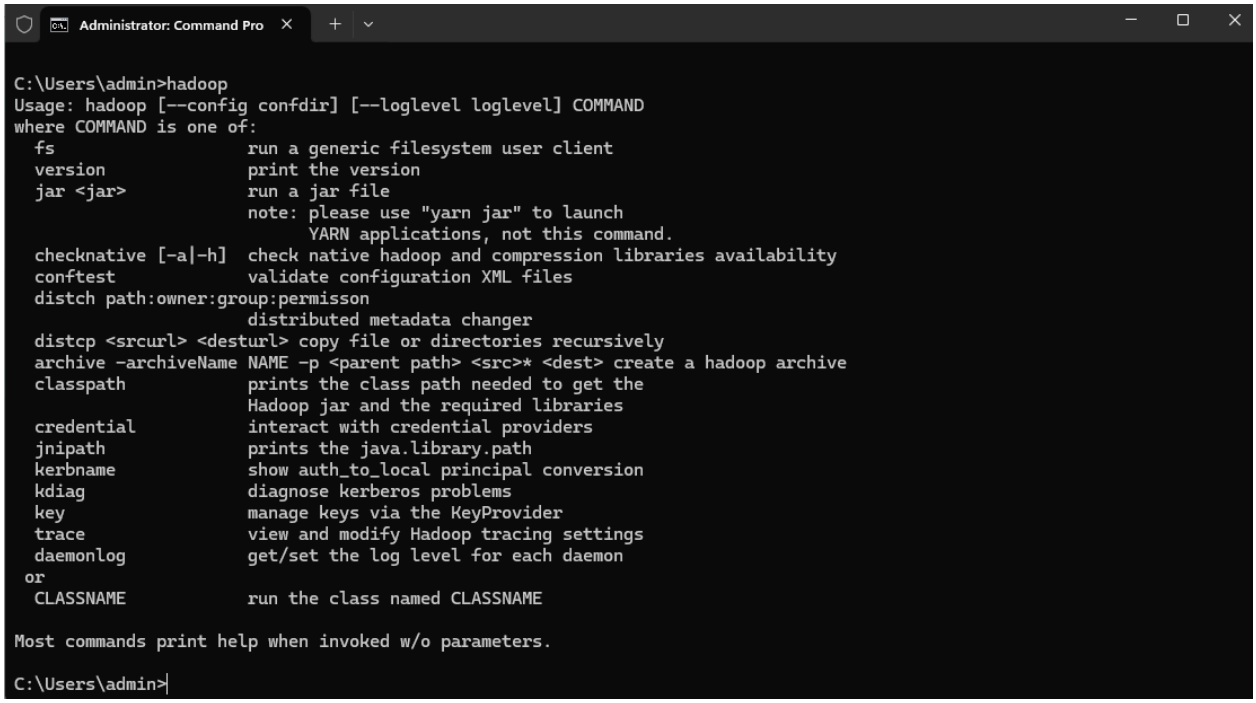
Step 11: Go to hadoop-env.cmd file in /etc/hadoop folder and replace the set JAVA_HOME=%JAVA_HOME% line with the following:

```
set JAVA_HOME=C:\Progra~1\Java\jdk-21
```

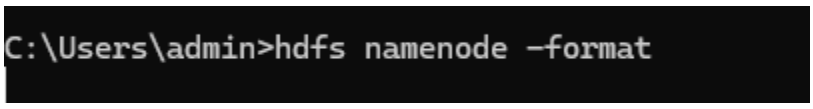
Step 12: Restart your PC for the changes to take effect.

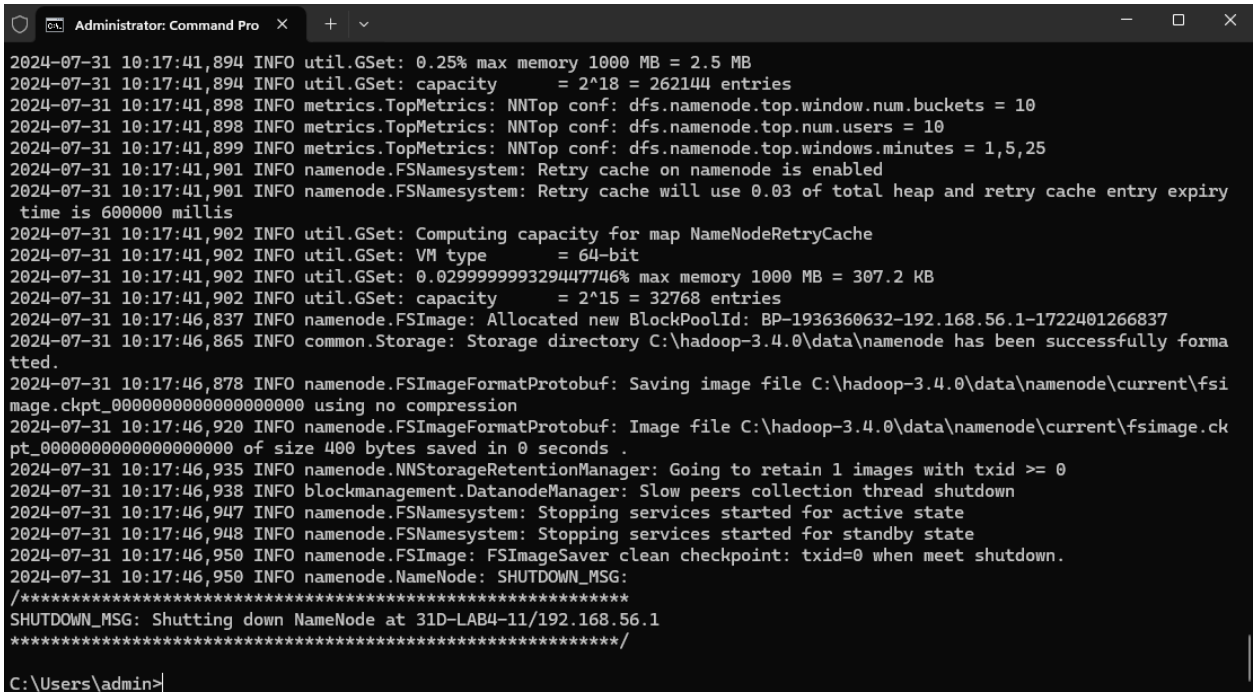
Step 13: Go to Admin Command prompt and type “hadoop” to see if the server is recognized.



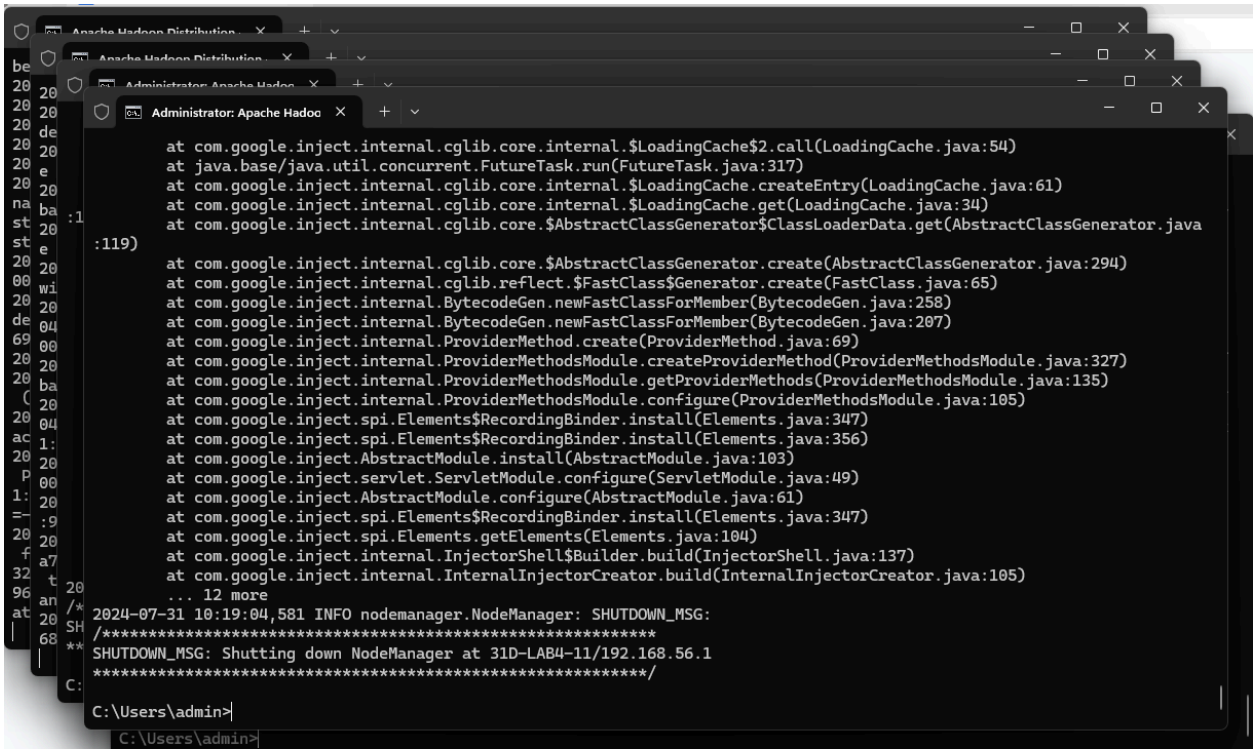


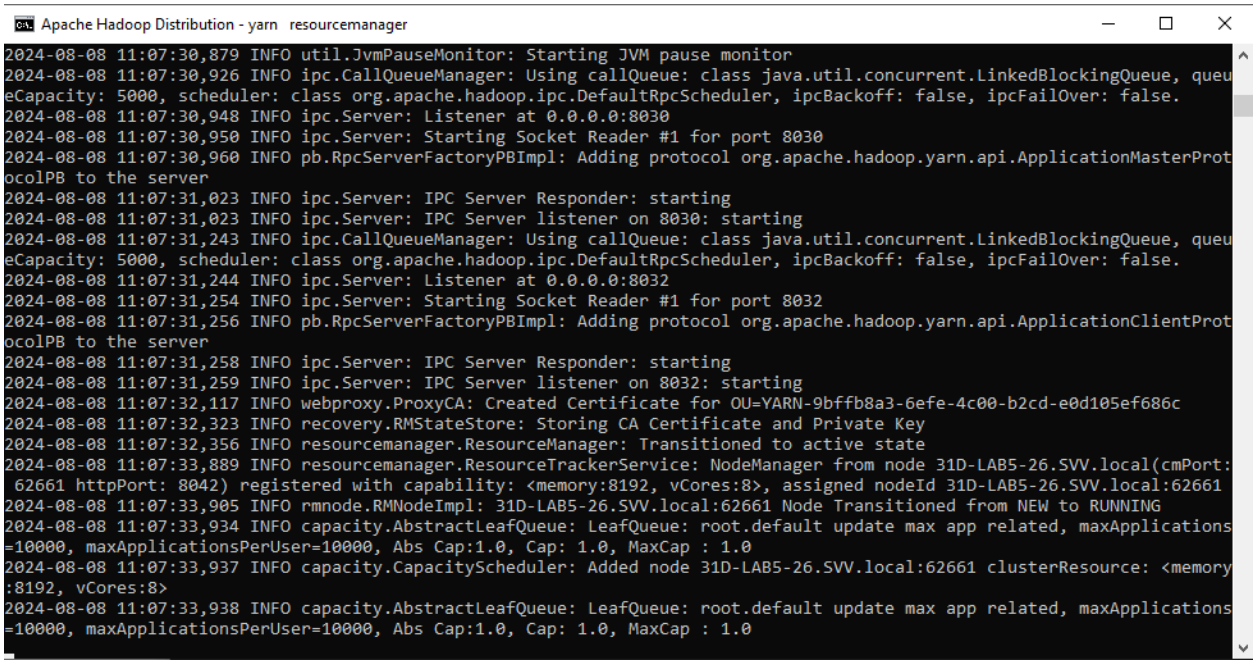
Step 14: Type “hdfs namenode -format” to format the namenode.

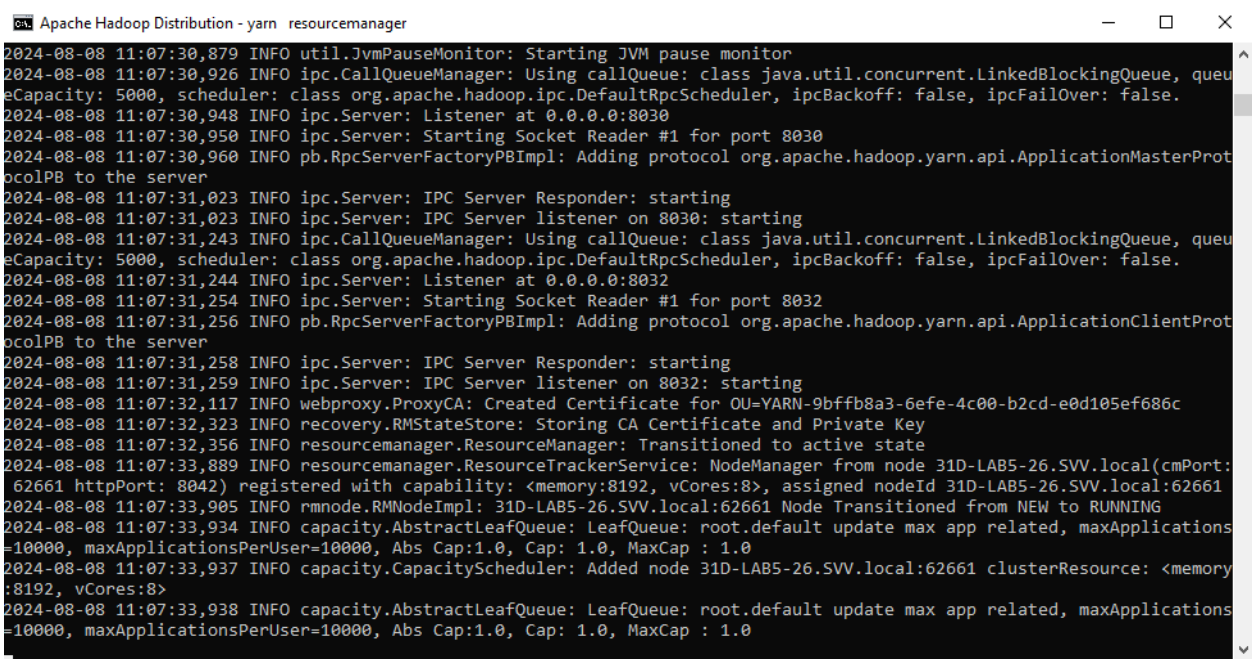




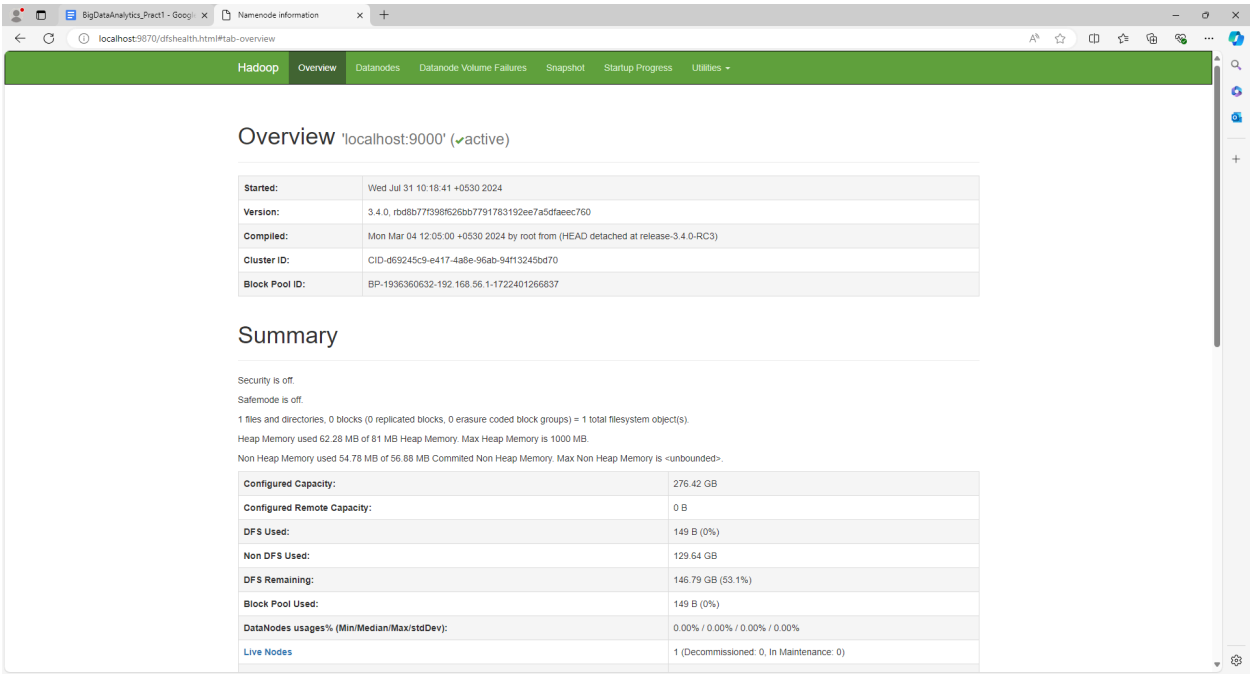
Step 15: Type start-all.cmd to start all hadoop processes (make sure you have added set JAVA_HOME=C:\Progra~1\Java\jdk-22)



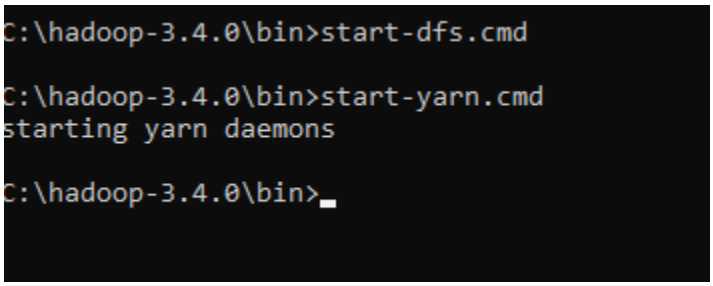




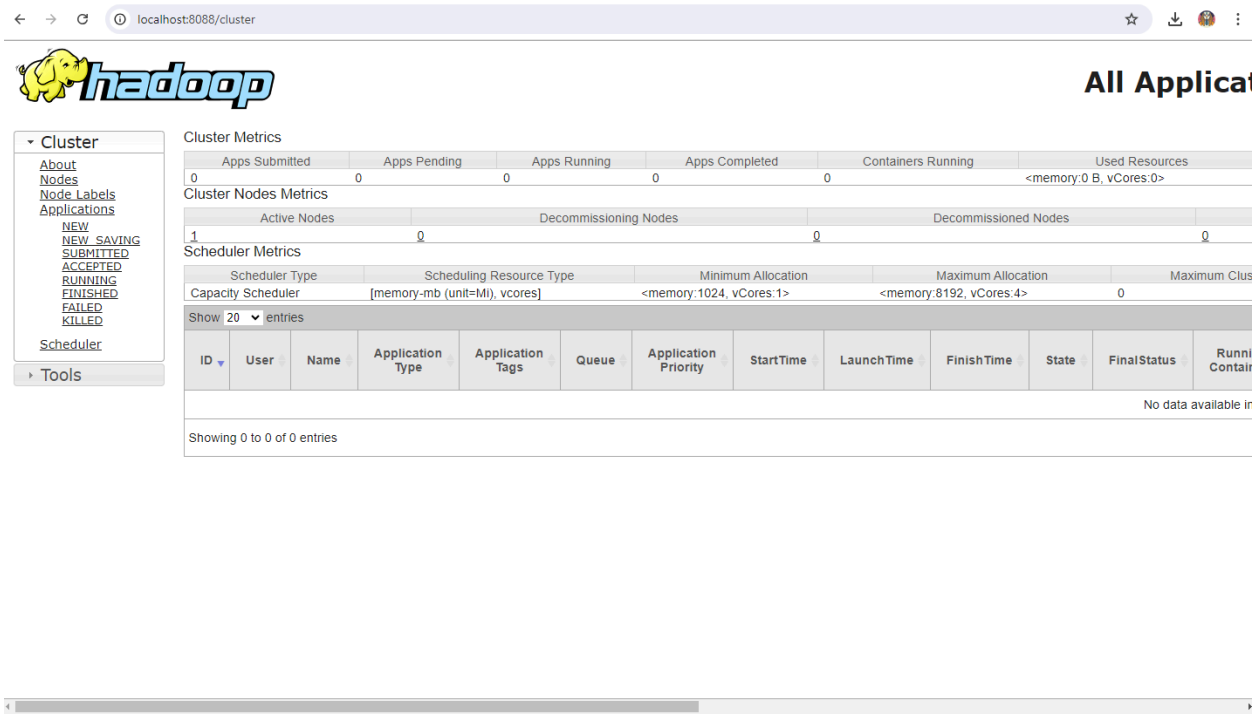
Step 16: Go to your browser and type localhost:9870 to view Hadoop Page.



Step 17: Now, go to cmd and type start-yarn.cmd:



Step 18: Now, go to localhost:8088 and observe accordingly:



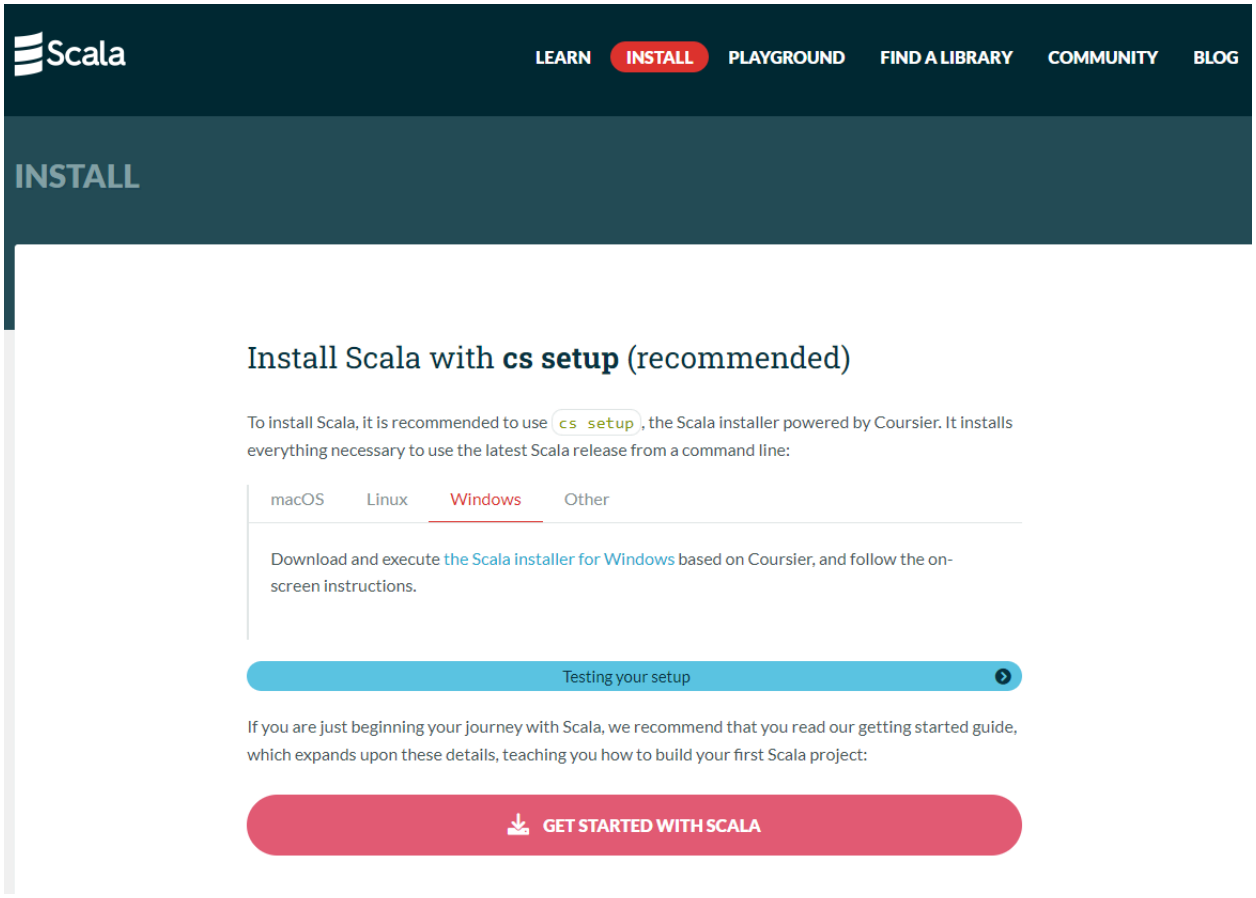
Practical 2:

Aim: Spark SQL

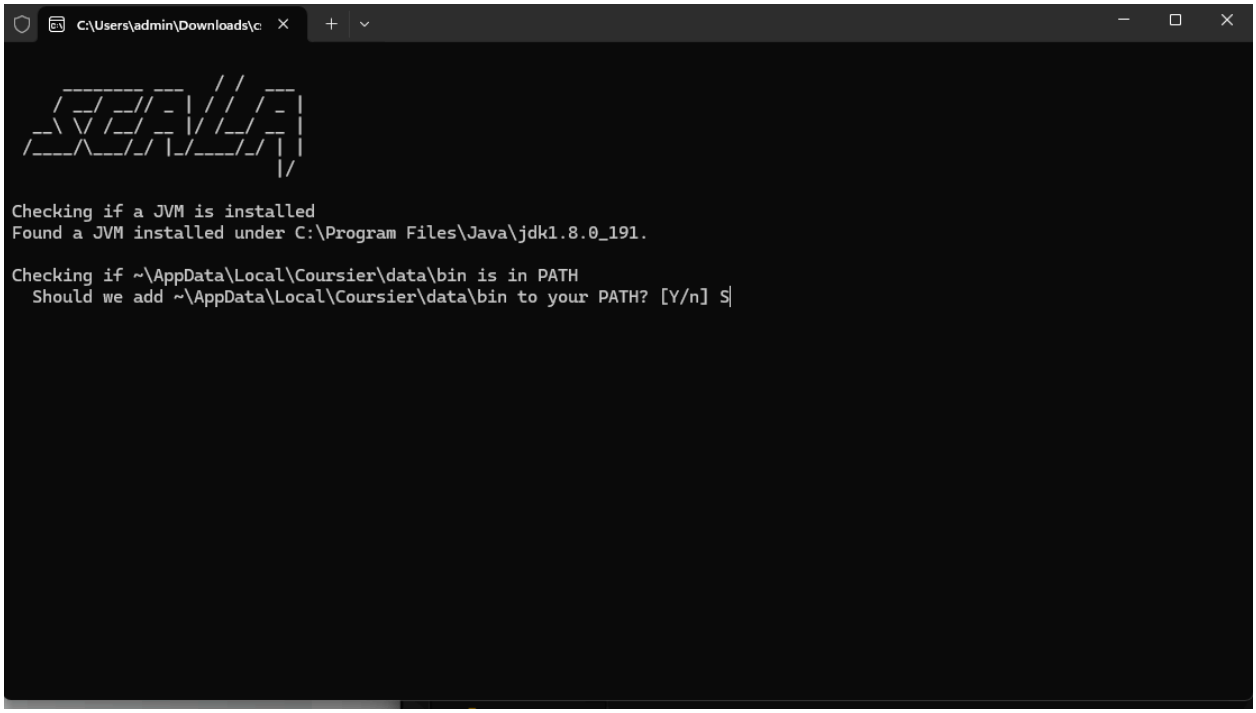
Steps:

A-Installation of Scala and Spark:

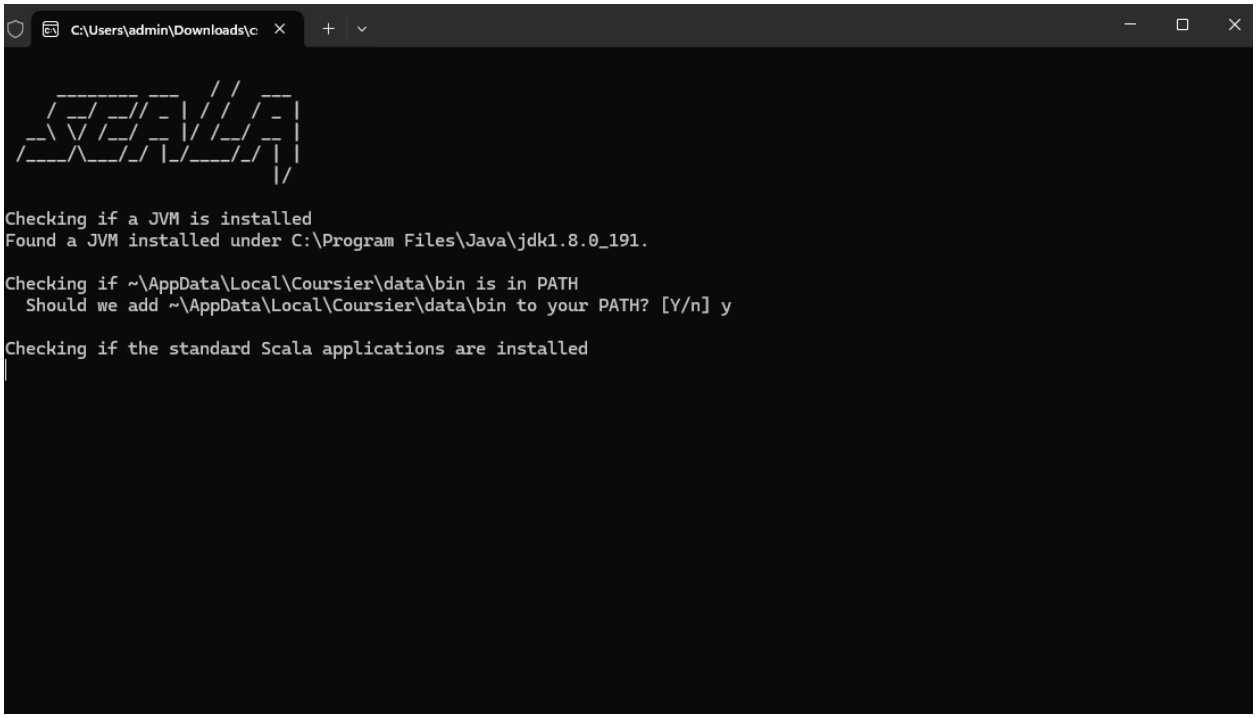
1-Install Scala and perform the following steps:


Scala
LEARN
INSTALL
PLAYGROUND
FIND A LIBRARY
COMMUNITY
BLOG
INSTALL
Install Scala with cs setup (recommended)
To install Scala, it is recommended to use cs setup, the Scala installer powered by Coursier. It installs everything necessary to use the latest Scala release from a command line:
macOS
Linux
Windows
Other
Download and execute the Scala installer for Windows based on Coursier, and follow the on-screen instructions.
Testing your setup
If you are just beginning your journey with Scala, we recommend that you read our getting started guide, which expands upon these details, teaching you how to build your first Scala project:
GET STARTED WITH SCALA

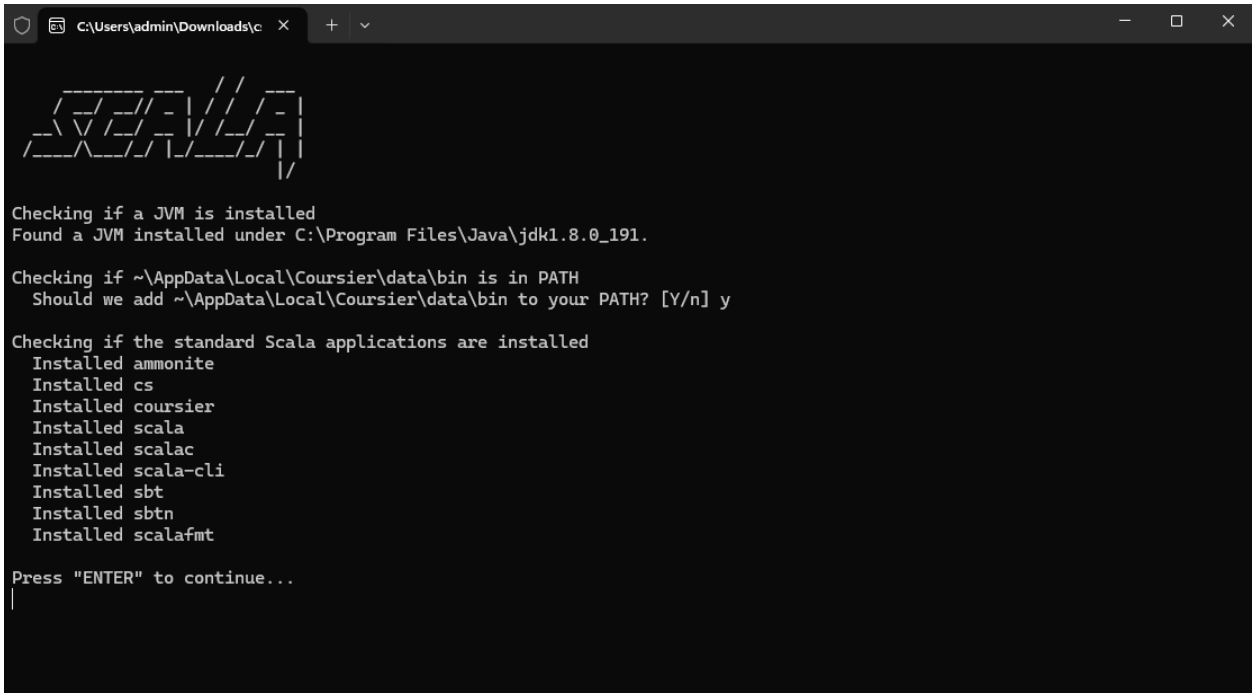
2-


C:\Users\admin\Downloads\c:
Checking if a JVM is installed
Found a JVM installed under C:\Program Files\Java\jdk1.8.0_191.
Checking if ~\AppData\Local\Coursier\data\bin is in PATH
Should we add ~\AppData\Local\Coursier\data\bin to your PATH? [Y/n] S

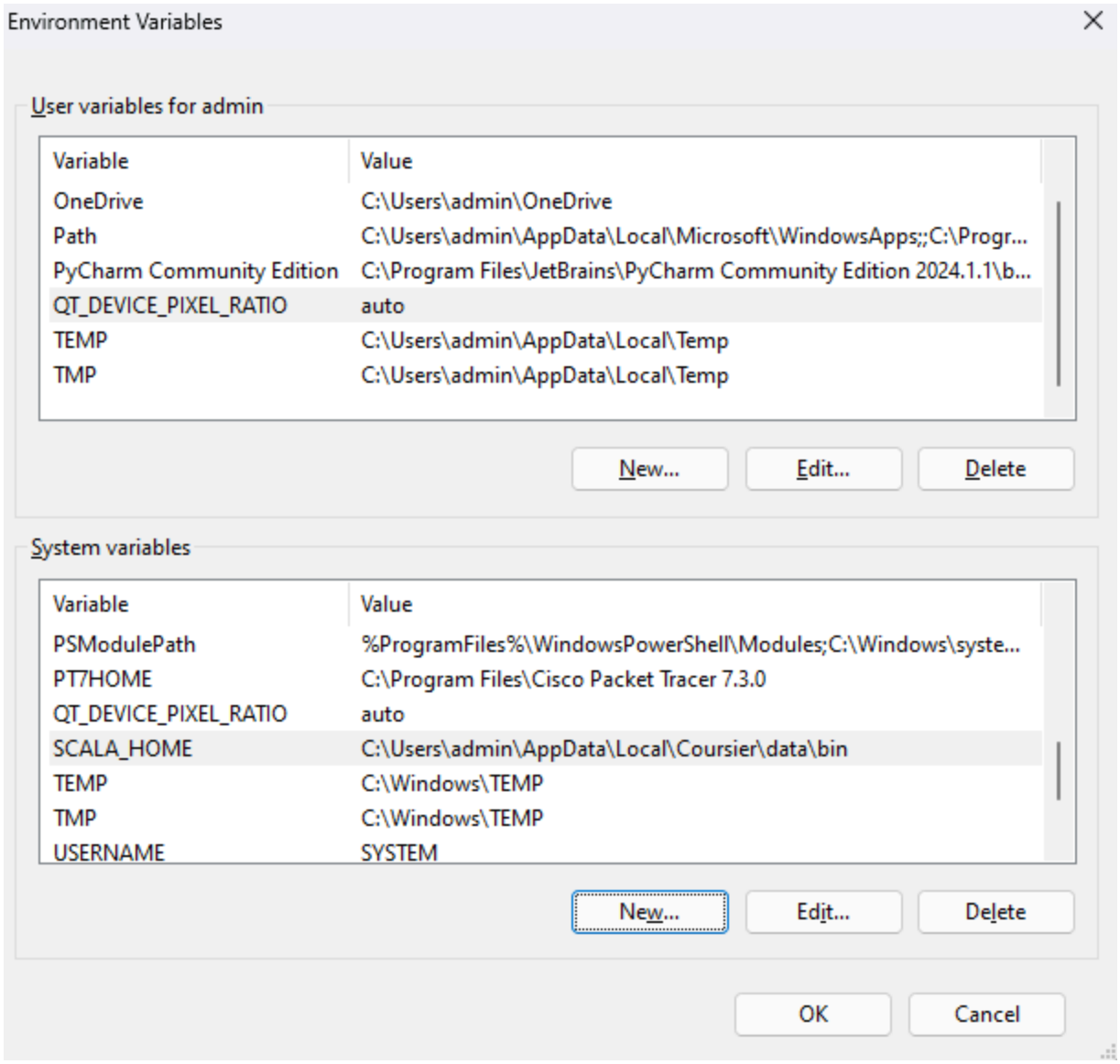
3-



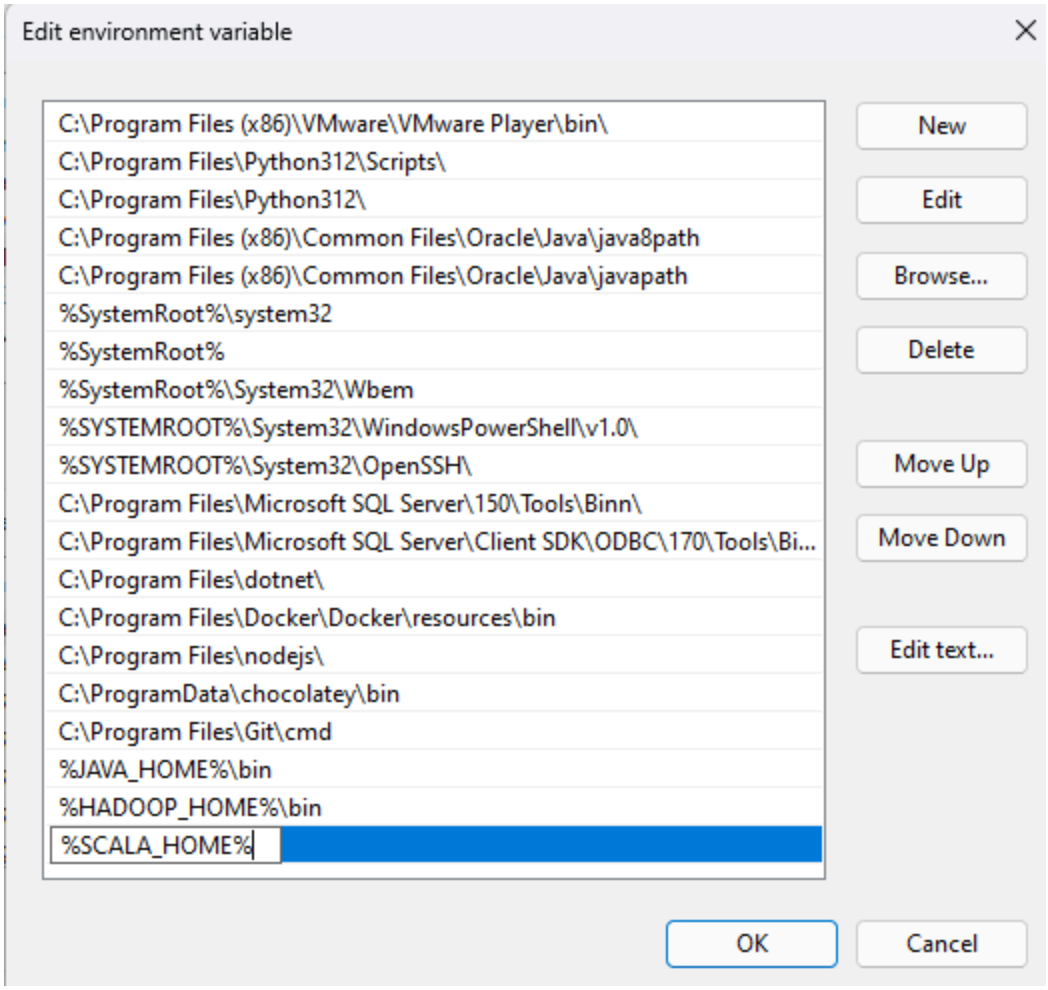
4-



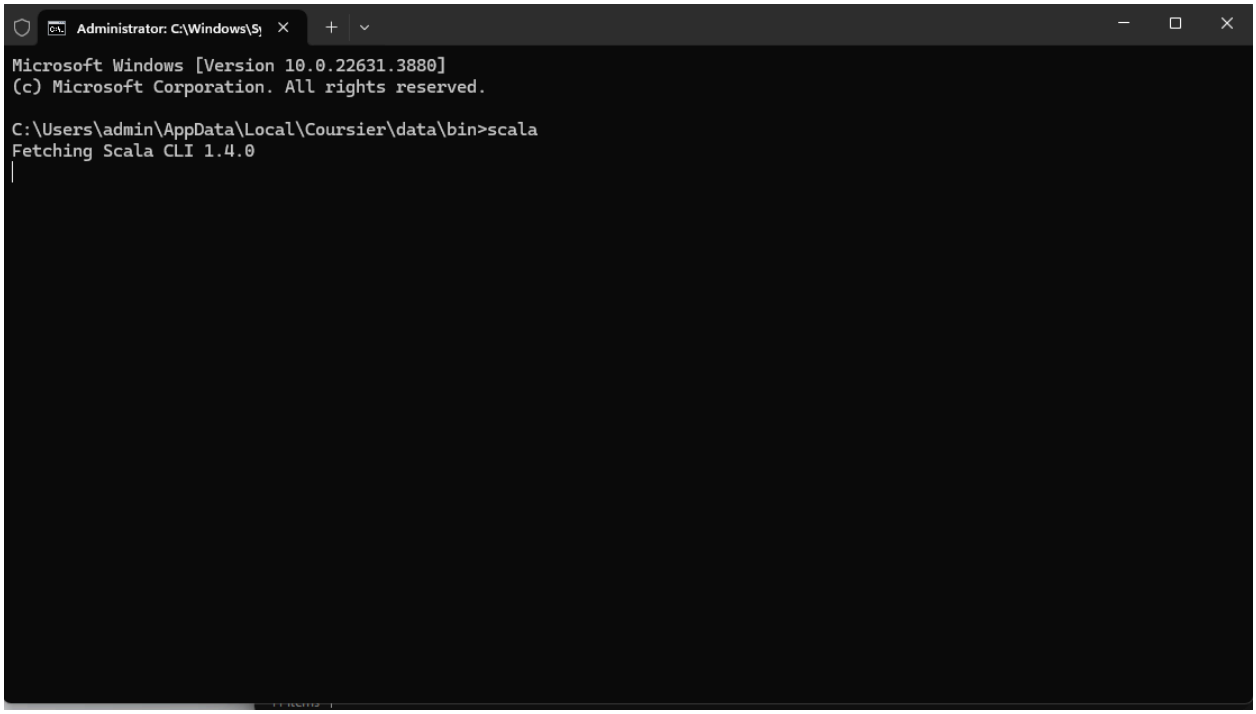
5-Now, make sure to set the environmental path variables


Environment Variables
User variables for admin
Variable
Value
OneDrive
C:\Users\admin\OneDrive
Path
C:\Users\admin\AppData\Local\Microsoft\WindowsApps;;C:\Progr...
PyCharm Community Edition
C:\Program Files\JetBrains\PyCharm Community Edition 2024.1.1\b...
QT_DEVICE_PIXEL_RATIO
auto
TEMP
C:\Users\admin\AppData\Local\Temp
TMP
C:\Users\admin\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
PSModulePath
%ProgramFiles%\WindowsPowerShell\Modules;C:\Windows\syste...
PT7HOME
C:\Program Files\Cisco Packet Tracer 7.3.0
QT_DEVICE_PIXEL_RATIO
auto
SCALA_HOME
C:\Users\admin\AppData\Local\Coursier\data\bin
TEMP
C:\Windows\TEMP
TMP
C:\Windows\TEMP
USERNAME
SYSTEM
New...
Edit...
Delete
OK
Cancel

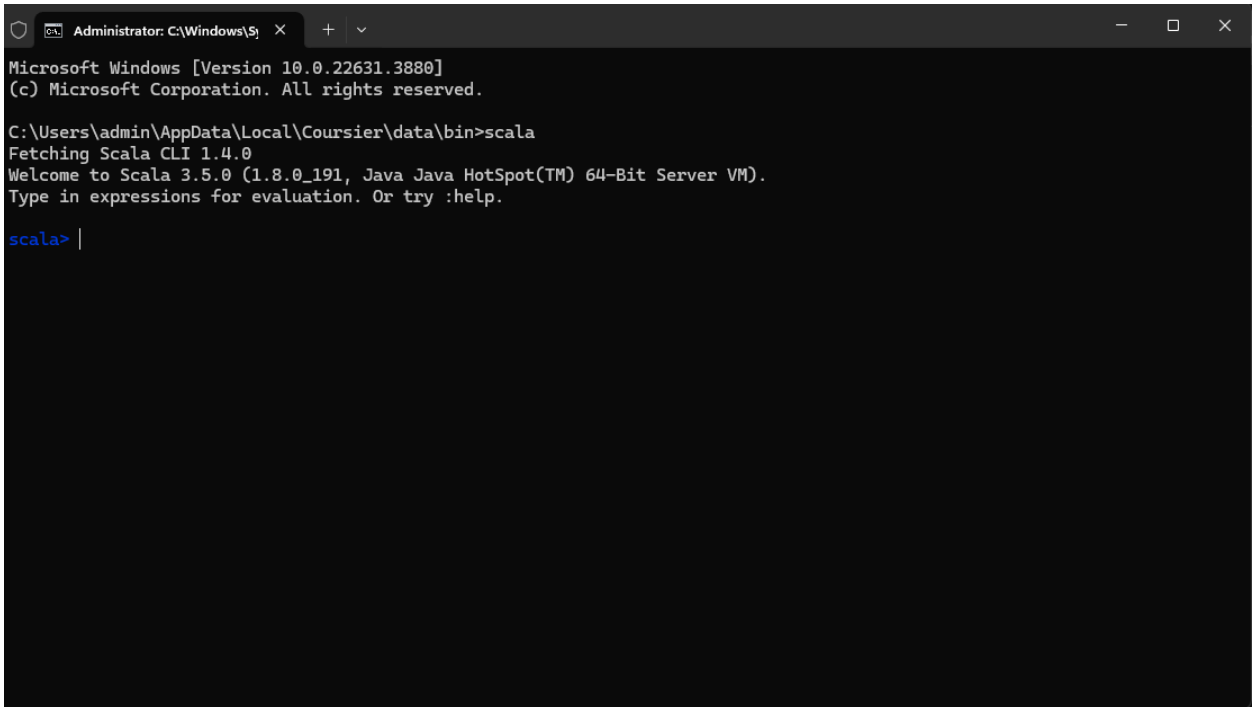
6-



7-Now, open the following with the help of cmd (enable hidden files if unable to find it as desired) and the following path C:\Users\admin\AppData\Local\Coursier\data\bin to access scala

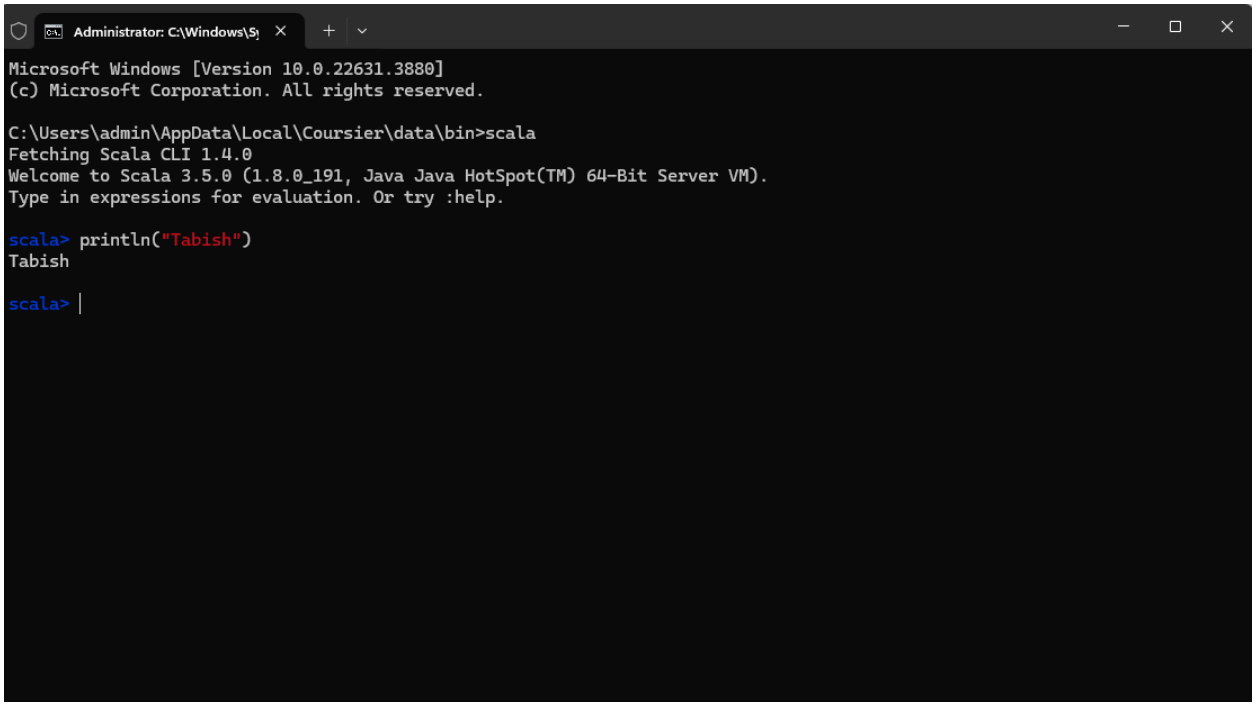


8-Successfully accessed the desired interface

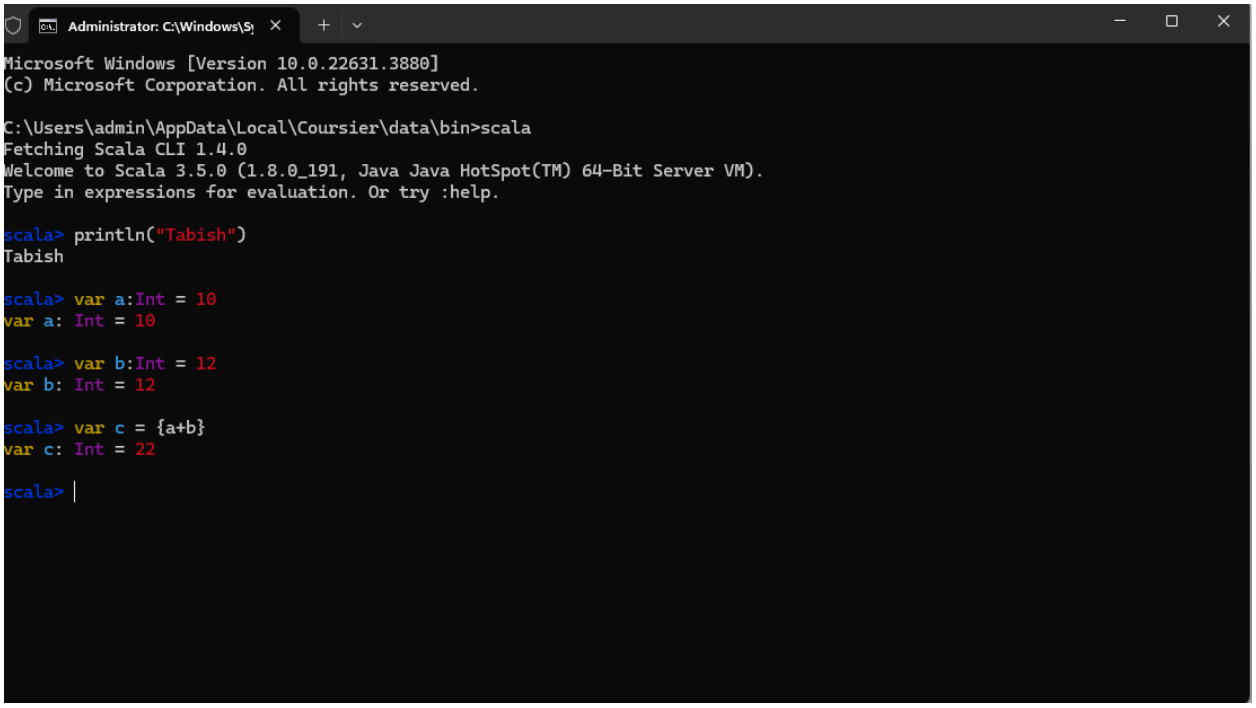


9-Perform the following programs:

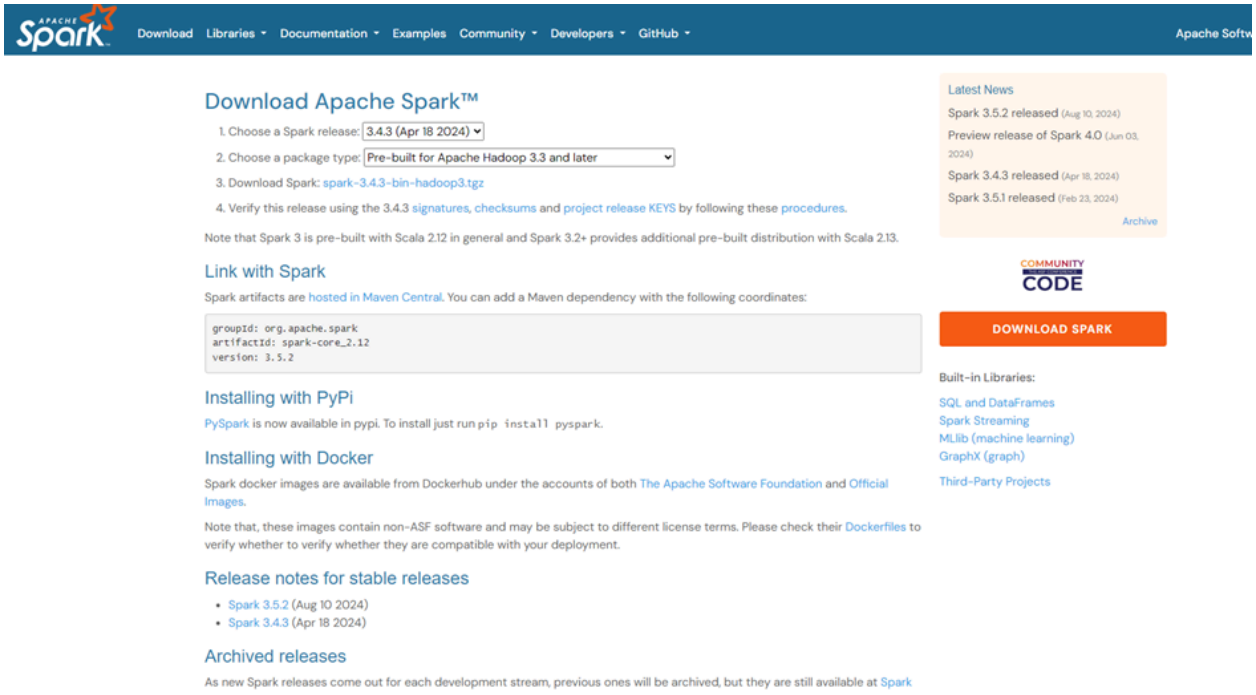
A-



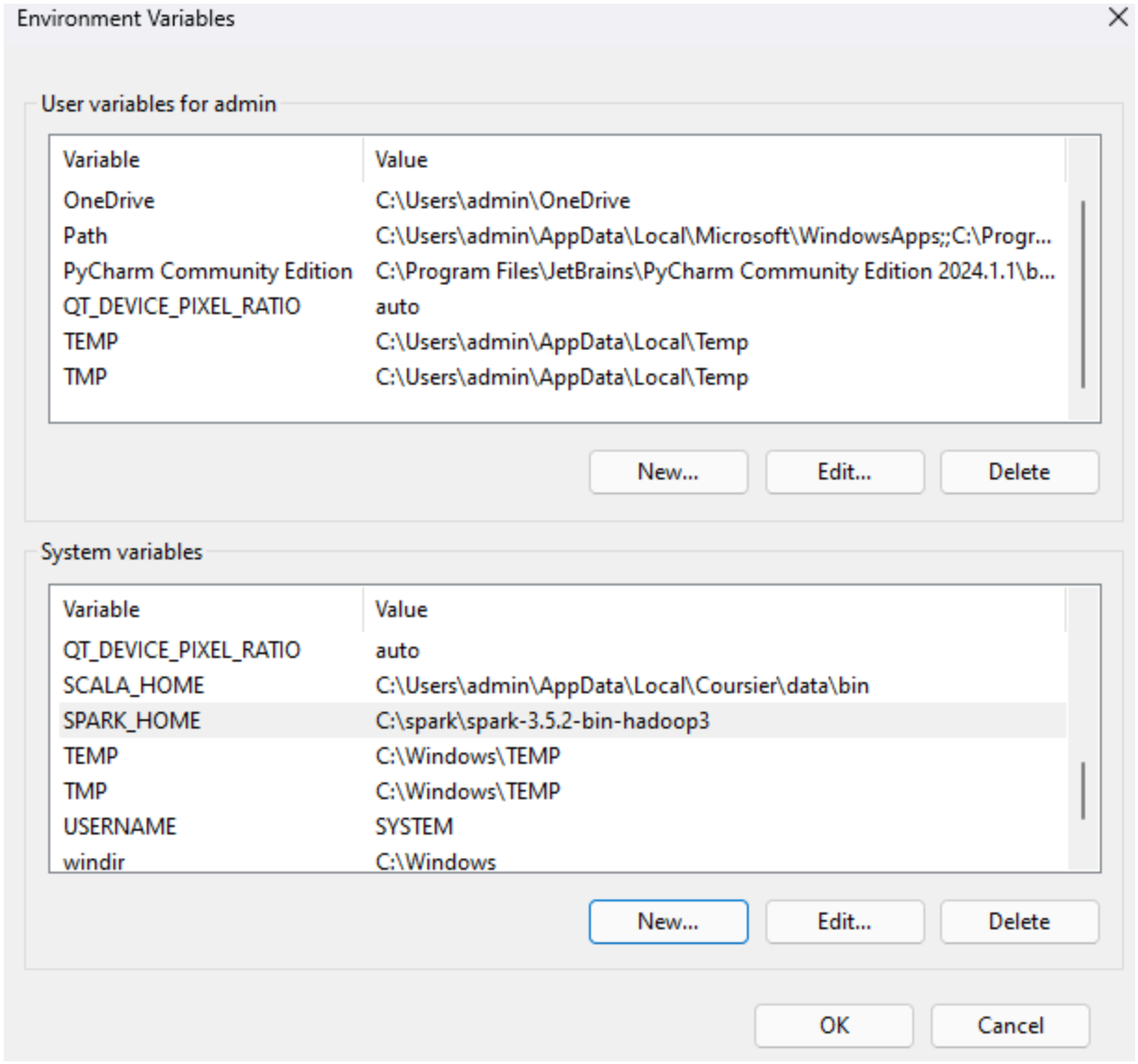
B-

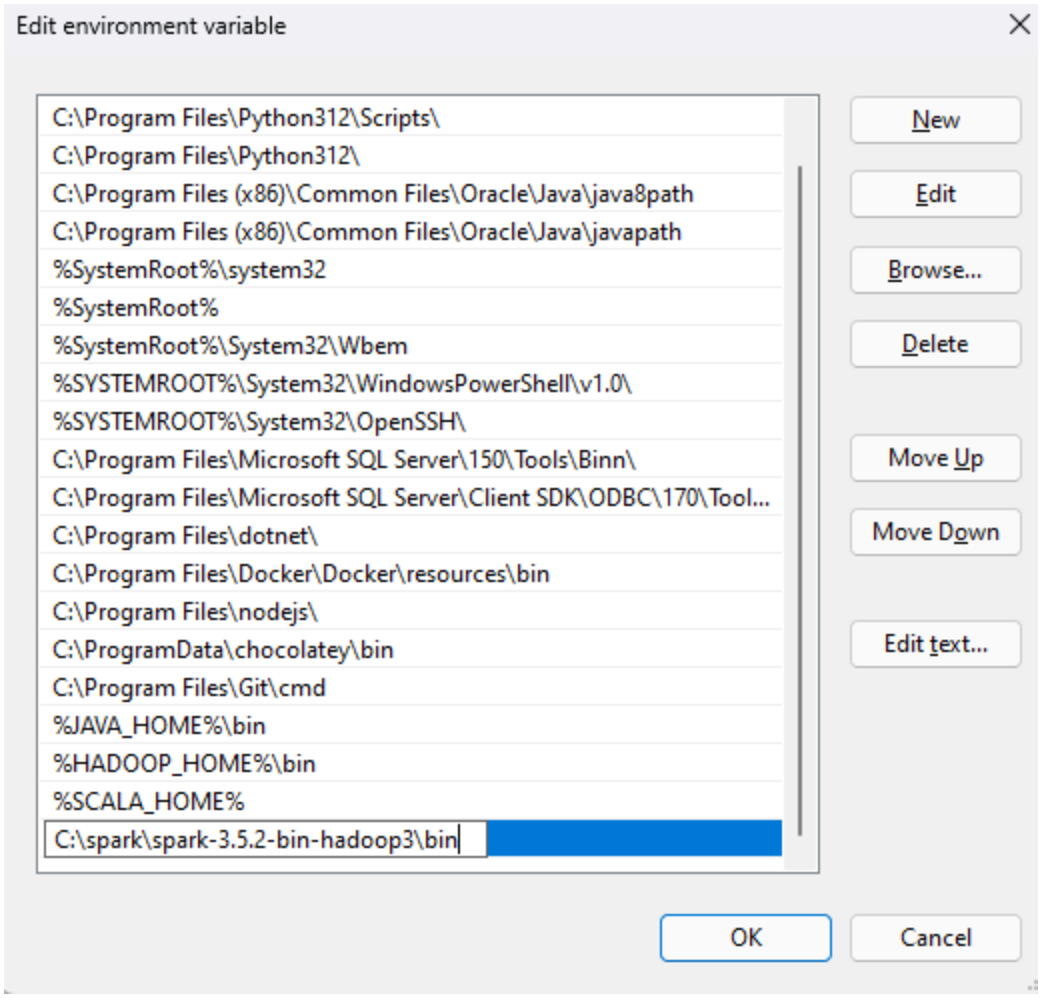


10-Install Spark



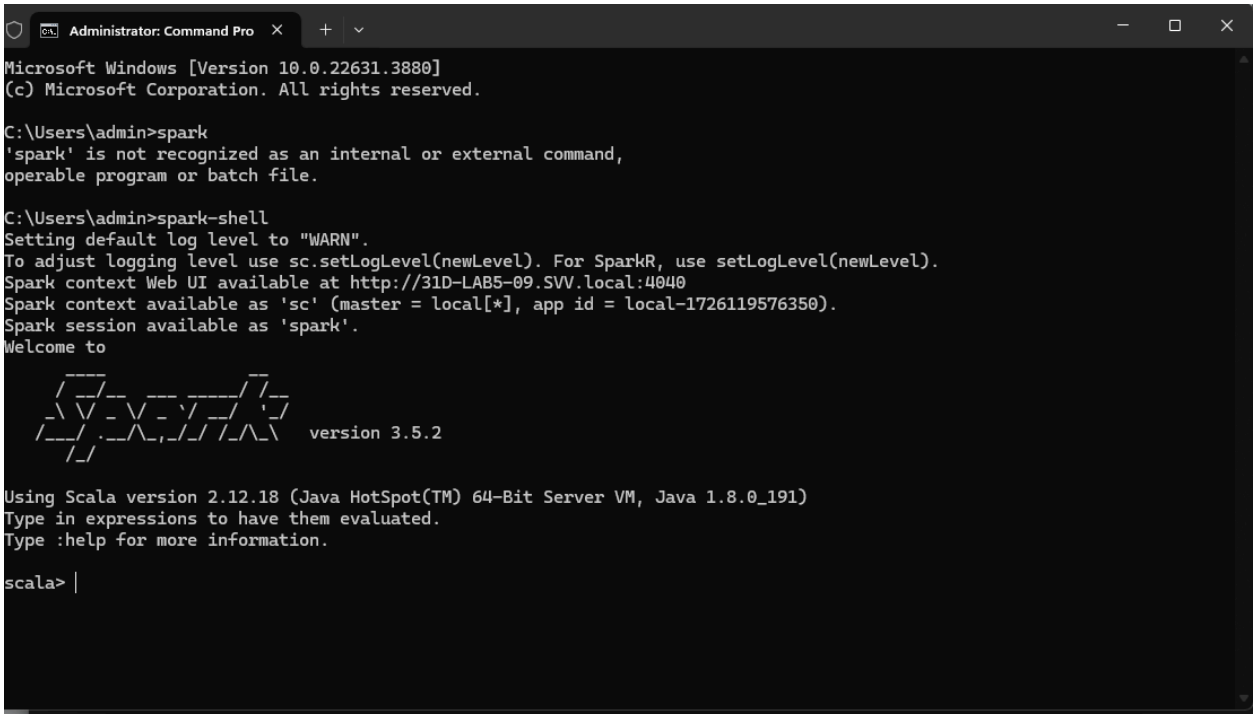
11-Set the environmental variables accordingly (using C:\spark\spark-3.5.2-bin-hadoop3\bin)


Environment Variables
User variables for admin
Variable
Value
OneDrive
C:\Users\admin\OneDrive
Path
C:\Users\admin\AppData\Local\Microsoft\WindowsApps;;C:\Progr...
PyCharm Community Edition
C:\Program Files\JetBrains\PyCharm Community Edition 2024.1.1\b...
QT_DEVICE_PIXEL_RATIO
auto
TEMP
C:\Users\admin\AppData\Local\Temp
TMP
C:\Users\admin\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
QT_DEVICE_PIXEL_RATIO
auto
SCALA_HOME
C:\Users\admin\AppData\Local\Coursier\data\bin
SPARK_HOME
C:\spark\spark-3.5.2-bin-hadoop3
TEMP
C:\Windows\TEMP
TMP
C:\Windows\TEMP
USERNAME
SYSTEM
windir
C:\Windows
New...
Edit...
Delete
OK
Cancel



12-Run spark (spark-shell) and perform the following:

A-



B-

```
scala> val Data = spark.read.json("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.json")
Data: org.apache.spark.sql.DataFrame = [age: bigint, name: string]
```

C-

```
scala> Data.show()
+----+-------+
| age|   name|
+----+-------+
|NULL|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+


scala>
```

D-

```
scala> Data.printSchema()
root
 |-- age: long (nullable = true)
 |-- name: string (nullable = true)


scala>
```

E-

```
scala> Data.select($"name",$"age").show()
+-------+----+
|   name| age|
+-------+----+
|Michael|NULL|
|   Andy|  30|
| Justin|  19|
+-------+----+


scala>
```

F-

```
scala> Data.filter($"age">20).show()
+---+----+
|age|name|
+---+----+
| 30|Andy|
+---+----+


scala>
```

G-

```
scala> Data.select($"age"+1).show()
+---------+
|(age + 1)|
+---------+
|     NULL|
|       31|
|       20|
+---------+


scala>
```

H-

```
scala> val Data2 = spark.read.csv("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.csv")
Data2: org.apache.spark.sql.DataFrame = [_c0: string]

scala> Data2.show()
+-----------------+
|              _c0|
+-----------------+
|      name;age;job|
|Jorge;30;Developer|
|  Bob;32;Developer|
+-----------------+


scala>
```

I-

```
scala> val Data2 = spark.read.option("header", "true").csv("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources
/people.csv")
Data2: org.apache.spark.sql.DataFrame = [name;age;job: string]

scala> Data2.show()
+-----------------+
|      name;age;job|
+-----------------+
|Jorge;30;Developer|
|  Bob;32;Developer|
+-----------------+


scala>
```

B-Now, further perform the following (refer: [Getting Started - Spark 3.5.2 Documentation (apache.org)](apache.org)):

A-

```
scala> val Data=spark.read.json("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.json")
Data: org.apache.spark.sql.DataFrame = [age: bigint, name: string]

scala> Data.createOrReplaceTempView("people")

scala> val sqlDF=spark.sql("SELECT * FROM people")
sqlDF: org.apache.spark.sql.DataFrame = [age: bigint, name: string]

scala> sqlDF.show()
+----+-------+
| age|   name|
+----+-------+
|NULL|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+


scala>
```

B-

```
scala> Data.createGlobalTempView("people")
24/09/19 10:34:09 WARN HiveConf: HiveConf of name hive.stats.jdbc.timeout does not exist
24/09/19 10:34:09 WARN HiveConf: HiveConf of name hive.stats.retries.wait does not exist
24/09/19 10:34:11 WARN ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 2.3.0
24/09/19 10:34:11 WARN ObjectStore: setMetaStoreSchemaVersion called but recording version is disabled: version = 2.3.0, comment = Set by MetaStore UNKNOWN@172.23.1.154
24/09/19 10:34:11 WARN ObjectStore: Failed to get database default, returning NoSuchObjectException
24/09/19 10:34:12 WARN ObjectStore: Failed to get database global_temp, returning NoSuchObjectException

scala> spark.sql("SELECT * FROM global_temp.people").show()
+----+-------+
| age|   name|
+----+-------+
|NULL|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+


scala> spark.newSession().sql("SELECT * FROM global_temp.people").show()
+----+-------+
| age|   name|
+----+-------+
|NULL|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+


scala>
```

C-

```
scala> case class Person(name: String, age: Long)
defined class Person

scala> val caseClassDS = Seq(Person("Andy", 32)).toDS()
caseClassDS: org.apache.spark.sql.Dataset[Person] = [name: string, age: bigint]

scala> caseClassDS.show()
+----+---+
|name|age|
+----+---+
|Andy| 32|
+----+---+


scala> val primitiveDS = Seq(1, 2, 3).toDS()
primitiveDS: org.apache.spark.sql.Dataset[Int] = [value: int]

scala> primitiveDS.map(_ + 1).collect() // Returns: Array(2, 3, 4)
res6: Array[Int] = Array(2, 3, 4)

scala> val path = "C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.json"
path: String = C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.json

scala> val peopleDS = spark.read.json(path).as[Person]
peopleDS: org.apache.spark.sql.Dataset[Person] = [age: bigint, name: string]

scala> peopleDS.show()
+----+-------+
| age|   name|
+----+-------+
|NULL|Michael|
|  30|   Andy|
|  19| Justin|
+----+-------+


scala>
```

D-

```
scala> import spark.implicits._
import spark.implicits._

scala> val peopleDF = spark.sparkContext
peopleDF: org.apache.spark.SparkContext = org.apache.spark.SparkContext@4599f1e0

scala> .textFile("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.txt")
res8: org.apache.spark.rdd.RDD[String] = C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.txt MapPartitionsRDD[28] at textFile at <console>:27

scala> .map(_.split(","))
res9: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[29] at map at <console>:27

scala> .map(attributes => Person(attributes(0), attributes(1).trim.toInt))
res10: org.apache.spark.rdd.RDD[Person] = MapPartitionsRDD[30] at map at <console>:29

scala> .toDF()
res11: org.apache.spark.sql.DataFrame = [name: string, age: bigint]

scala> peopleDF.createOrReplaceTempView("people")
```

```
scala> val teenagersDF = spark.sql("SELECT name, age FROM people WHERE age BETWEEN 13 AND 19")
teenagersDF: org.apache.spark.sql.DataFrame = [name: string, age: bigint]

scala> teenagersDF.map(teenager => "Name: " + teenager(0)).show()
+------------+
|       value|
+------------+
|Name: Justin|
+------------+


scala> teenagersDF.map(teenager => "Name: " + teenager.getAs[String]("name")).show()
+------------+
|       value|
+------------+
|Name: Justin|
+------------+


scala> implicit val mapEncoder = org.apache.spark.sql.Encoders.kryo[Map[String, Any]]
mapEncoder: org.apache.spark.sql.Encoder[Map[String,Any]] = class[value[0]: binary]

scala> teenagersDF.map(teenager => teenager.getValuesMap[Any](List("name", "age"))).collect()
res16: Array[Map[String,Any]] = Array(Map(name -> Justin, age -> 19))

scala>
```

E-

```
scala> import org.apache.spark.sql.Row
import org.apache.spark.sql.Row

scala> import org.apache.spark.sql.types._
import org.apache.spark.sql.types._

scala> val peopleRDD = spark.sparkContext.textFile("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.txt")
peopleRDD: org.apache.spark.rdd.RDD[String] = C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.txt MapPartitionsRDD[44] at textFile at <console>:30

scala> val schemaString = "name age"
schemaString: String = name age

scala> val fields = schemaString.split(" ")
fields: Array[String] = Array(name, age)

scala> .map(fieldName => StructField(fieldName, StringType, nullable = true))
res17: Array[org.apache.spark.sql.types.StructField] = Array(StructField(name,StringType,true), StructField(age,StringType,true))

scala> val schema = StructType(fields)
```

```
scala> val rowRDD = peopleRDD
rowRDD: org.apache.spark.rdd.RDD[String] = C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.txt MapPartitionsRDD[44] at textFile at <console>:30
```

```
scala> .map(_.split(","))
res18: org.apache.spark.rdd.RDD[Array[String]] = MapPartitionsRDD[45] at map at <console>:32

scala> .map(attributes => Row(attributes(0), attributes(1).trim))
res19: org.apache.spark.rdd.RDD[org.apache.spark.sql.Row] = MapPartitionsRDD[46] at map at <console>:32

scala> val peopleDF = spark.createDataFrame(rowRDD, schema)
```

```
scala> val schema = StructType(fields)
```

```
scala> val peopleDF = spark.createDataFrame(rowRDD, schema)
```

```
scala> val results = spark.sql("SELECT name FROM people")
results: org.apache.spark.sql.DataFrame = [name: string]

scala> results.map(attributes => "Name: " + attributes(0)).show()
+-------------+
|        value|
+-------------+
|Name: Michael|
|   Name: Andy|
| Name: Justin|
+-------------+


scala>
```

F-

```
scala> val Data = spark.createDataFrame(rowRDD, schema)
<console>:31: error: package schema is not a value
       val Data = spark.createDataFrame(rowRDD, schema)
                                        ^

scala> val results = spark.sql("SELECT name FROM people")
results: org.apache.spark.sql.DataFrame = [name: string]

scala> results.map(attributes => "Name: " + attributes(0)).show()
+-------------+
|        value|
+-------------+
|Name: Michael|
|   Name: Andy|
| Name: Justin|
+-------------+


scala> val mydata=spark.read.format("csv").option("inferschema","true").option("header","true").load("C:/spark/spark-3.5.2-bin-hadoop3/examples/src/main/resources/people.csv")
mydata: org.apache.spark.sql.DataFrame = [name;age;job: string]

scala> mydata.show()
+-----------------+
|      name;age;job|
+-----------------+
|Jorge;30;Developer|
|  Bob;32;Developer|
+-----------------+


scala> mydata.show(50)
+-----------------+
|      name;age;job|
+-----------------+
|Jorge;30;Developer|
|  Bob;32;Developer|
+-----------------+
```

```
scala> mydata.select($"name;age;job").show()
+------------------+
|      name;age;job|
+------------------+
|Jorge;30;Developer|
|  Bob;32;Developer|
+------------------+


scala> mydata.select($"name;age;job")
res36: org.apache.spark.sql.DataFrame = [name;age;job: string]

scala>
```

```
scala> mydata.count()
res28: Long = 2

scala> mydata.count.toDouble
res29: Double = 2.0

scala> val totalcount=mydata.count.toDouble
totalcount: Double = 2.0
```

G-

```
scala> val res=spark.sql("select* from people where age>20")
res: org.apache.spark.sql.DataFrame = [age: bigint, name: string]

scala> res.w
<console>:32: error: value w is not a member of org.apache.spark.sql.DataFrame
       res.w
           ^

scala> res.write.save("mynewdf")

scala> res.show()
+---+----+
|age|name|
+---+----+
| 30|Andy|
+---+----+


scala>
```

Practical 3:

Aim: Spark GraphX

Steps: Perform the following

1-

import org.apache.spark._
import org.apache.spark.rdd.RDD
import org.apache.spark.graphx._

```
scala> import org.apache.spark._
import org.apache.spark._

scala> import org.apache.spark.rdd.RDD
import org.apache.spark.rdd.RDD

scala> import org.apache.spark.graphx._
import org.apache.spark.graphx._
```

2-

val vertices = Array((1L,("A")),(2L,("B")),(3L,("C")))

```
scala> val vertices = Array((1L,("A")),(2L,("B")),(3L,("C")))
val vertices: Array[(Long, String)] = Array((1,A), (2,B), (3,C))
```

3-

val vRDD = sc.parallelize(vertices)

```
scala> val vRDD = sc.parallelize(vertices)
warning: 1 deprecation (since 2.13.0); for details, enable `:setting -deprecation` or `:replay -deprec
ation`
val vRDD: org.apache.spark.rdd.RDD[(Long, String)] = ParallelCollectionRDD[0] at parallelize at <conso
le>:1
```

4-

vRDD.take(1)
vRDD.take(2)

```
scala> vRDD.take(1)
val res0: Array[(Long, String)] = Array((1,A))

scala> vRDD.take(2)
val res1: Array[(Long, String)] = Array((1,A), (2,B))
```

5-

val edges = Array(Edge(1L,2L,1800),Edge(2L,3L,800),Edge(3L,1L,1400))

```
scala> val edges = Array(Edge(1L,2L,1800),Edge(2L,3L,800),Edge(3L,1L,1400))
val edges: Array[org.apache.spark.graphx.Edge[Int]] = Array(Edge(1,2,1800), Edge(2,3,800), Edge(3,1,14
00))
```

6-

val eRDD = sc.parallelize(edges)

```
scala> val eRDD = sc.parallelize(edges)
warning: 1 deprecation (since 2.13.0); for details, enable `:setting -deprecation` or `:replay -deprec
ation`
val eRDD: org.apache.spark.rdd.RDD[org.apache.spark.graphx.Edge[Int]] = ParallelCollectionRDD[1] at pa
rallelize at <console>:1
```

7-

eRDD.take(2)

```
scala> eRDD.take(2)
val res2: Array[org.apache.spark.graphx.Edge[Int]] = Array(Edge(1,2,1800), Edge(2,3,800))
```

8-

val nowhere = "nowhere"

```
scala> val nowhere = "nowhere"
val nowhere: String = nowhere
```

9-

val graph = Graph(vRDD,eRDD,nowhere)

```
scala> val graph = Graph(vRDD,eRDD,nowhere)
val graph: org.apache.spark.graphx.Graph[String,Int] = org.apache.spark.graphx.impl.GraphImpl@2e008502
```

10-

#To check number of Airports
val numairports = graph.numVertices

```
scala> val numairports = graph.numVertices
val numairports: Long = 3
```

11-

#To check routes
val numairports = graph.numEdges

```
scala> val numairports = graph.numEdges
val numairports: Long = 3
```

12-

#Route having distance > 1000
(graph.edges.filter{case Edge(src,dst,prop)=>prop>1000}.collect.foreach(println))

```
scala> (graph.edges.filter{case Edge(src,dst,prop)=>prop>1000}.collect.foreach(println))
warning: 1 deprecation (since 2.13.3); for details, enable `:setting -deprecation` or `:replay -deprec
ation`
Edge(1,2,1800)
Edge(3,1,1400)
```

13-

#Triplet Information
graph.triplets.take(3).foreach(println)

```
scala> graph.triplets.take(3).foreach(println)
((1,A),(2,B),1800)
((2,B),(3,C),800)
((3,C),(1,A),1400)
```

14-

#Indegree
val i = graph.inDegrees
i.collect()

```
scala> val i = graph.inDegrees
val i: org.apache.spark.graphx.VertexRDD[Int] = VertexRDDImpl[25] at RDD at VertexRDD.scala:57

scala> i.collect()
val res5: Array[(org.apache.spark.graphx.VertexId, Int)] = Array((1,1), (2,1), (3,1))
```

15-

#Outdegrees
val o = graph.outDegrees
o.collect()

```
scala> val o = graph.outDegrees
val o: org.apache.spark.graphx.VertexRDD[Int] = VertexRDDImpl[29] at RDD at VertexRDD.scala:57

scala> o.collect()
val res6: Array[(org.apache.spark.graphx.VertexId, Int)] = Array((1,1), (2,1), (3,1))
```

16-

#Total Degree
val t = graph.degrees
t.collect()

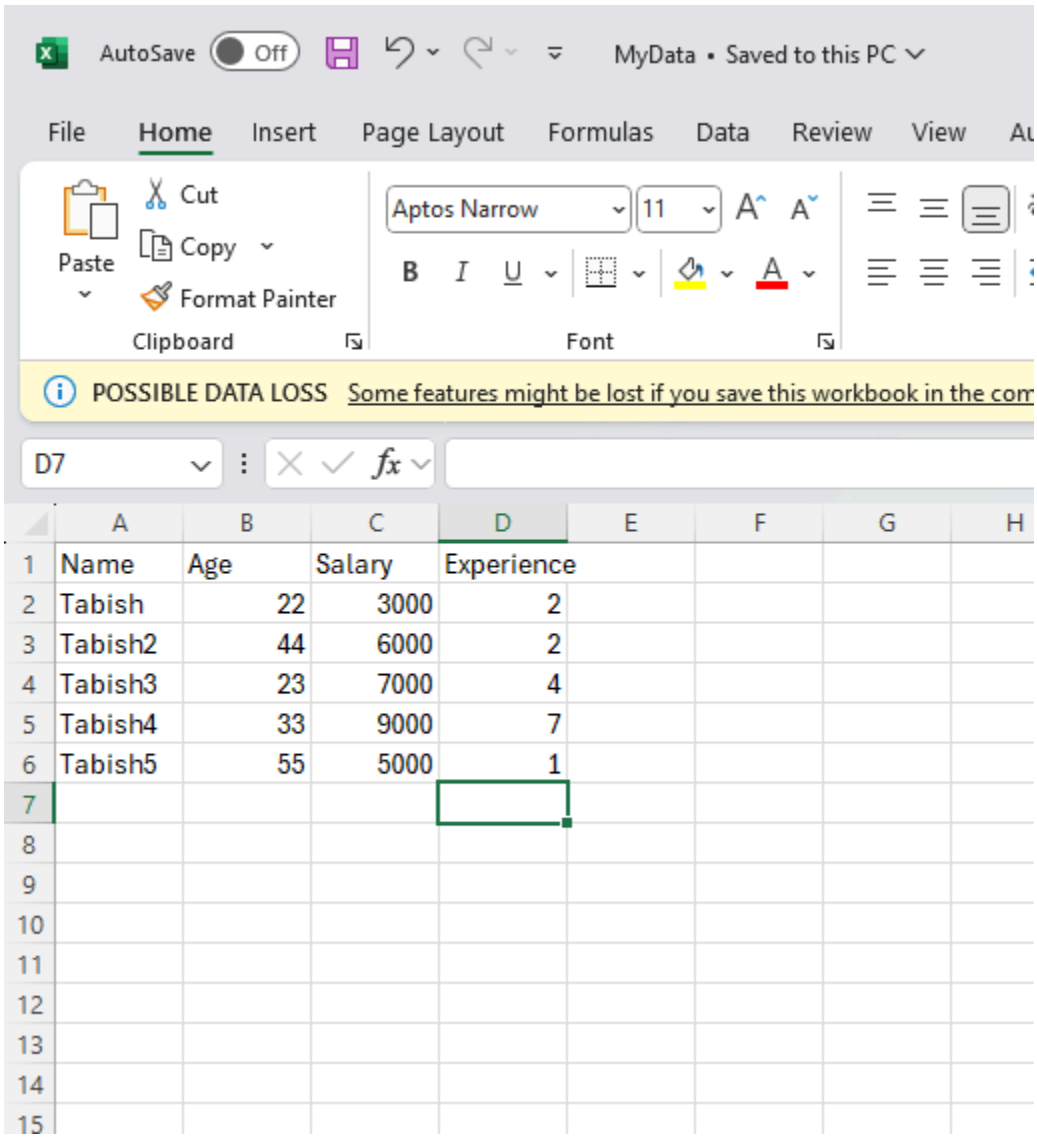
```
scala> val t = graph.degrees
val t: org.apache.spark.graphx.VertexRDD[Int] = VertexRDDImpl[33] at RDD at VertexRDD.scala:57

scala> t.collect()
val res8: Array[(org.apache.spark.graphx.VertexId, Int)] = Array((1,2), (2,2), (3,2))
```

Practical 4:

Aim: PySpark

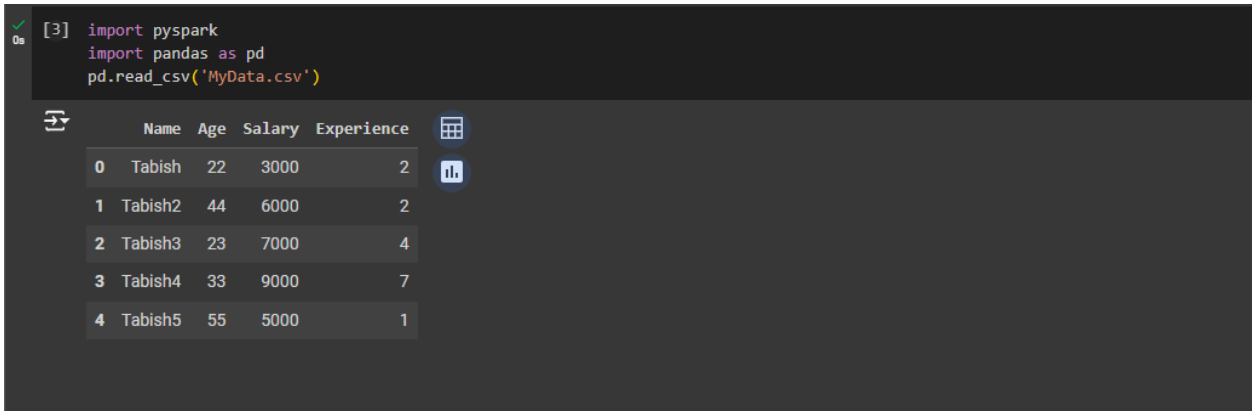
Step 1:



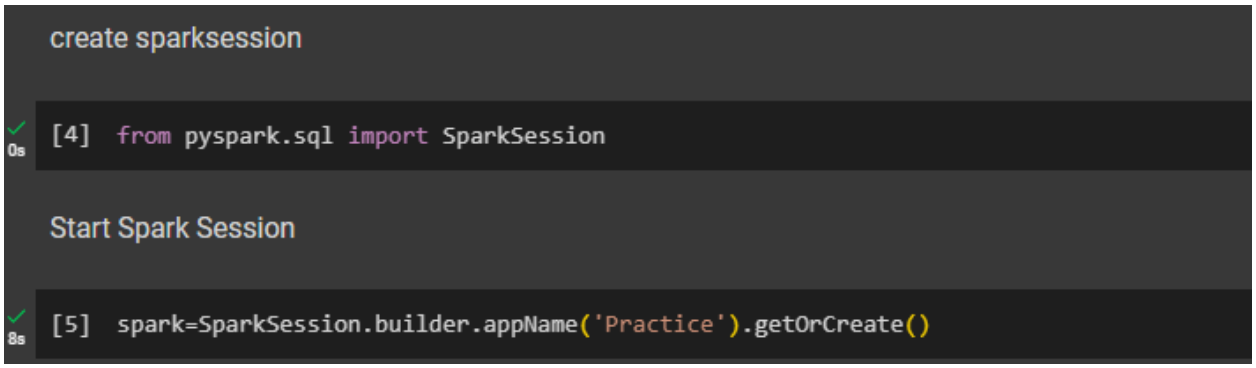
	A	B	C	D	E	F	G	H
1	Name	Age	Salary	Experience				
2	Tabish	22	3000	2				
3	Tabish2	44	6000	2				
4	Tabish3	23	7000	4				
5	Tabish4	33	9000	7				
6	Tabish5	55	5000	1				
7								
8								
9								
10								
11								
12								
13								
14								
15								

Step 2: Install PySpark

Step 2:



Step 3:



Step 4:

Read dataset and store in variable

```
df_pyspark=spark.read.csv('MyData.csv', header=True, inferSchema=True)
df_pyspark.show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
+-------+---+------+----------+
```

Step 5:

Type of the data

```
type(df_pyspark)
```

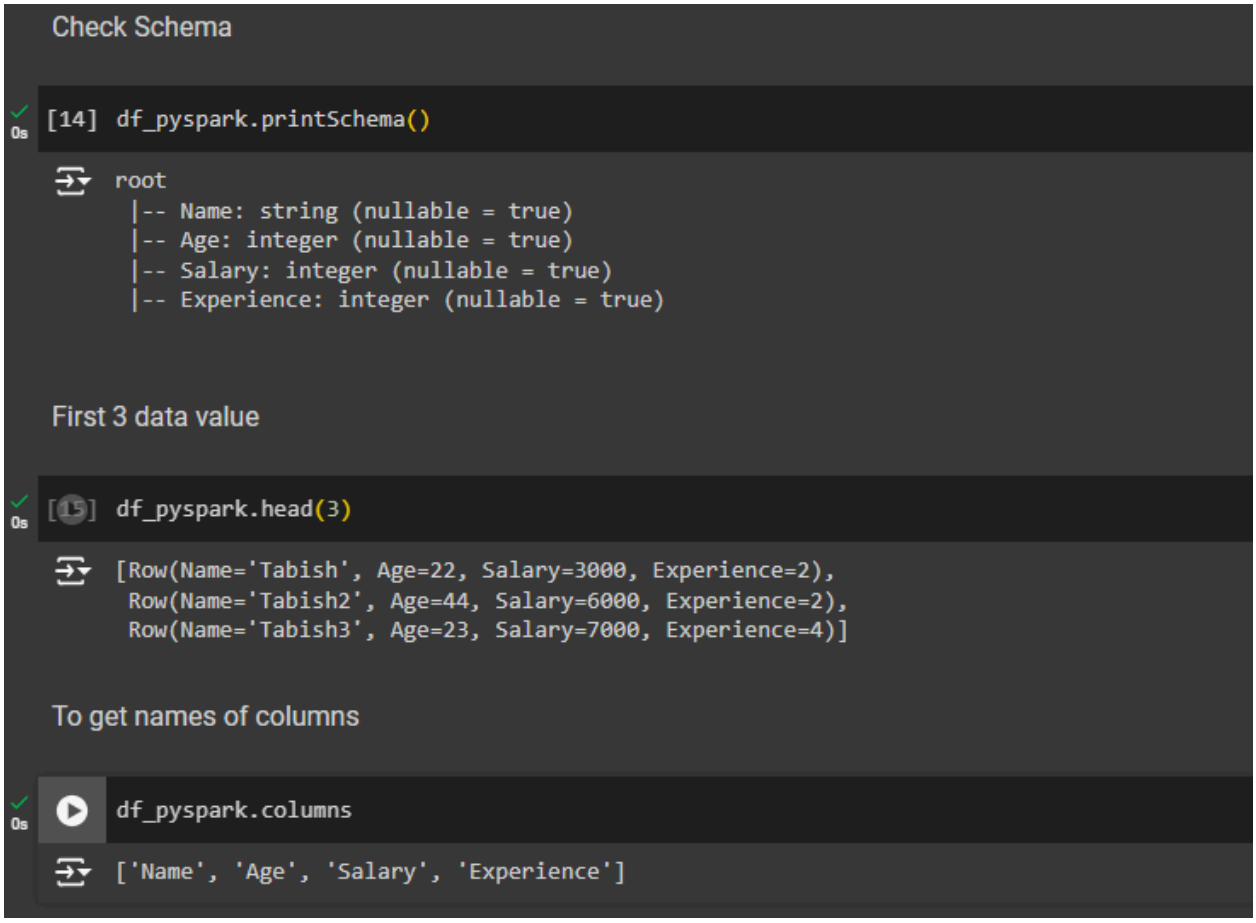
```
pyspark.sql.dataframe.DataFrame
def __init__(jdf: JavaObject, sql_ctx: Union['SQLContext', 'SparkSession'])

/usr/local/lib/python3.10/dist-packages/pyspark/sql/dataframe.py
A distributed collection of data grouped into named columns.

.. versionadded:: 1.3.0

.. versionchanged:: 3.4.0
```

Step 6:


Check Schema
[14] df_pyspark.printSchema()
root
|-- Name: string (nullable = true)
|-- Age: integer (nullable = true)
|-- Salary: integer (nullable = true)
|-- Experience: integer (nullable = true)
First 3 data value
[15] df_pyspark.head(3)
[Row(Name='Tabish', Age=22, Salary=3000, Experience=2),
Row(Name='Tabish2', Age=44, Salary=6000, Experience=2),
Row(Name='Tabish3', Age=23, Salary=7000, Experience=4)]
To get names of columns
df_pyspark.columns
['Name', 'Age', 'Salary', 'Experience']

Step 7:

Selecting Specific Columns

```
df_pyspark.select('Name').show()
df_pyspark.select(['Name','Experience']).show()
```

```
+-------+
|   Name|
+-------+
| Tabish|
|Tabish2|
|Tabish3|
|Tabish4|
|Tabish5|
+-------+

+-------+----------+
|   Name|Experience|
+-------+----------+
| Tabish|         2|
|Tabish2|         2|
|Tabish3|         4|
|Tabish4|         7|
|Tabish5|         1|
+-------+----------+
```

Step 8:

To check datatypes

```
[23] df_pyspark.dtypes
```

```
[('Name', 'string'), ('Age', 'int'), ('Salary', 'int'), ('Experience', 'int')]
```

To describe the dataset

```
df_pyspark.describe().show()
```

```
+-------+-------+------------------+----------------+------------------+
|summary|   Name|               Age|          Salary|        Experience|
+-------+-------+------------------+----------------+------------------+
|  count|      5|                 5|               5|                 5|
|   mean|   NULL|              35.4|          6000.0|               3.2|
| stddev|   NULL|14.117365193264641|2236.06797749979|2.387467277262665|
|    min| Tabish|                22|            3000|                 1|
|    max|Tabish5|                55|            9000|                 7|
+-------+-------+------------------+----------------+------------------+
```

Step 9:

Adding columns in data frame

```
[25] df_pyspark=df_pyspark.withColumn('Experience after 2 years', df_pyspark['Experience']+2)
```

Display the info

```
[26] df_pyspark.show()
```

```
+-------+---+------+----------+----------------------+
|   Name|Age|Salary|Experience|Experience after 2 years|
+-------+---+------+----------+----------------------+
| Tabish| 22|  3000|         2|                     4|
|Tabish2| 44|  6000|         2|                     4|
|Tabish3| 23|  7000|         4|                     6|
|Tabish4| 33|  9000|         7|                     9|
|Tabish5| 55|  5000|         1|                     3|
+-------+---+------+----------+----------------------+
```

Drop columns

```
df_pyspark=df_pyspark.drop('Experience after 2 years').show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
+-------+---+------+----------+
```

Step 10:

Renaming columns

```
df_pyspark.withColumnRenamed('Name','New Name')
df_pyspark.show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
+-------+---+------+----------+
```

```
[56] df_pyspark.na.drop().show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
+-------+---+------+----------+
```

```
[57] df_pyspark.na.drop(how='all').show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
+-------+---+------+----------+
```

Step 11:

Renaming columns

```
df_pyspark=df_pyspark.withColumnRenamed('Name','New Name')
df_pyspark.show()
```

```
+--------+---+------+----------+
|New Name|Age|Salary|Experience|
+--------+---+------+----------+
|  Tabish| 22|  3000|         2|
| Tabish2| 44|  6000|         2|
| Tabish3| 23|  7000|         4|
| Tabish4| 33|  9000|         7|
| Tabish5| 55|  5000|         1|
|  Tabish| 33|  NULL|          |
+--------+---+------+----------+
```

Step 12:

```
[129] df_pyspark=df_pyspark.na.drop(how='all')
      df_pyspark.show()
```

```
+---------+---+------+----------+
|New Name|Age|Salary|Experience|
+---------+---+------+----------+
|   Tabish| 22|  3000|         2|
| Tabish2| 44|  6000|         2|
| Tabish3| 23|  7000|         4|
| Tabish4| 33|  9000|         7|
| Tabish5| 55|  5000|         1|
| Tabish6| 33|  NULL|          |
| Tabish7| 35|  NULL|      NULL|
+---------+---+------+----------+
```

```
[130] df_pyspark.na.drop(how='any')
      df_pyspark.show()
```

```
+---------+---+------+----------+
|New Name|Age|Salary|Experience|
+---------+---+------+----------+
|   Tabish| 22|  3000|         2|
| Tabish2| 44|  6000|         2|
| Tabish3| 23|  7000|         4|
| Tabish4| 33|  9000|         7|
| Tabish5| 55|  5000|         1|
| Tabish6| 33|  NULL|          |
| Tabish7| 35|  NULL|      NULL|
+---------+---+------+----------+
```

Step 13:

threshold

```
df_pyspark.na.drop(how='any', thresh=2)
df_pyspark.show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
|Tabish6| 33|  NULL|          |
|Tabish7| 35|  NULL|      NULL|
+-------+---+------+----------+
```

subset

```
[184] df_pyspark=df_pyspark.na.drop(how='any', subset=['Experience']).show()
```

```
+-------+---+------+----------+
|   Name|Age|Salary|Experience|
+-------+---+------+----------+
| Tabish| 22|  3000|         2|
|Tabish2| 44|  6000|         2|
|Tabish3| 23|  7000|         4|
|Tabish4| 33|  9000|         7|
|Tabish5| 55|  5000|         1|
|Tabish6| 33|  NULL|          |
+-------+---+------+----------+
```

Step 14:

Filling the missing value

```
df_pyspark.na.fill('Missing Values').show()
```

```
+-------+---+------+--------------+
|   Name|Age|Salary|    Experience|
+-------+---+------+--------------+
| Tabish| 22|  3000|             2|
|Tabish2| 44|  6000|             2|
|Tabish3| 23|  7000|             4|
|Tabish4| 33|  9000|             7|
|Tabish5| 55|  5000|             1|
|Tabish6| 33|  NULL|              |
|Tabish7| 35|  NULL|Missing Values|
+-------+---+------+--------------+
```

Step 15:

Writing the function to calculate the value at the place of Null

```
[208] from pyspark.ml.feature import Imputer

      imputer= Imputer(
          inputCols=['Age','Salary'],
          outputCols=["{}_imputed".format(c) for c in ['Age','Salary']]).setStrategy('mean'
```

Add imputation cols to df

```
imputer.fit(df_pyspark).transform(df_pyspark).show()
```

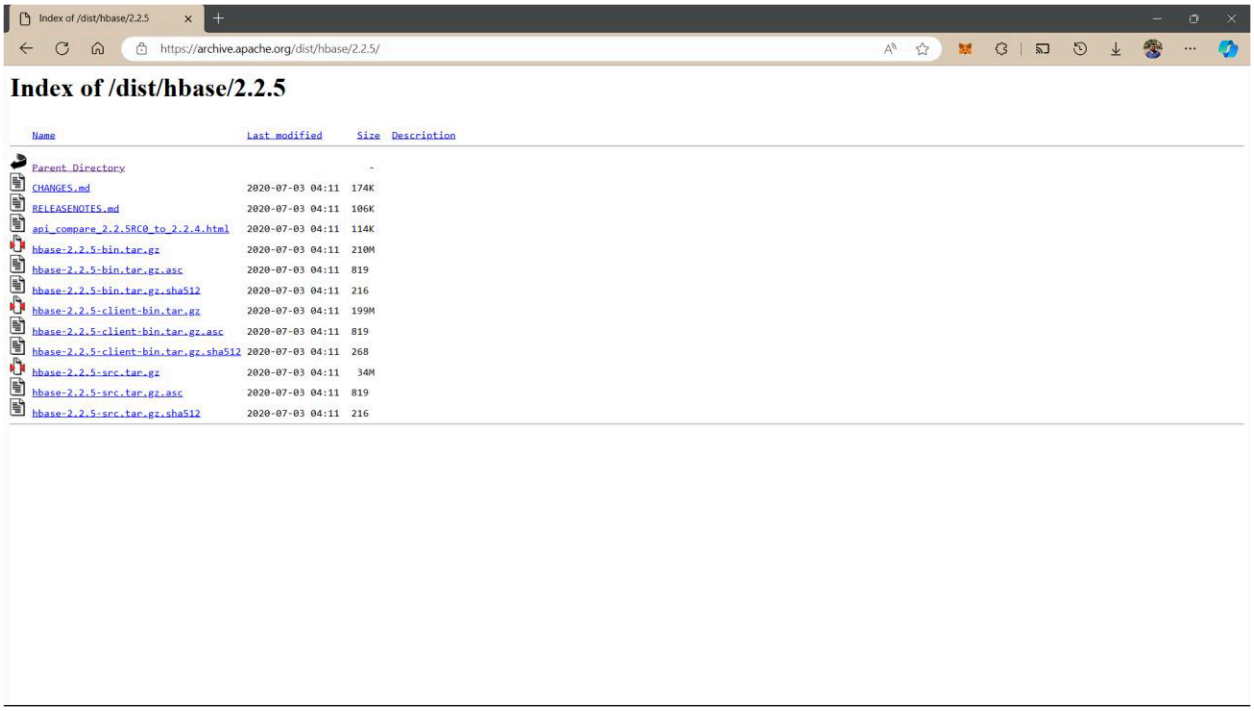
```
+-------+---+------+----------+-----------+--------------+
|   Name|Age|Salary|Experience|Age_imputed|Salary_imputed|
+-------+---+------+----------+-----------+--------------+
| Tabish| 22|  3000|         2|         22|          3000|
|Tabish2| 44|  6000|         2|         44|          6000|
|Tabish3| 23|  7000|         4|         23|          7000|
|Tabish4| 33|  9000|         7|         33|          9000|
|Tabish5| 55|  5000|         1|         55|          5000|
|Tabish6| 33|  NULL|          |         33|          6000|
|Tabish7| 35|  NULL|      NULL|         35|          6000|
+-------+---+------+----------+-----------+--------------+
```

Practical 5:

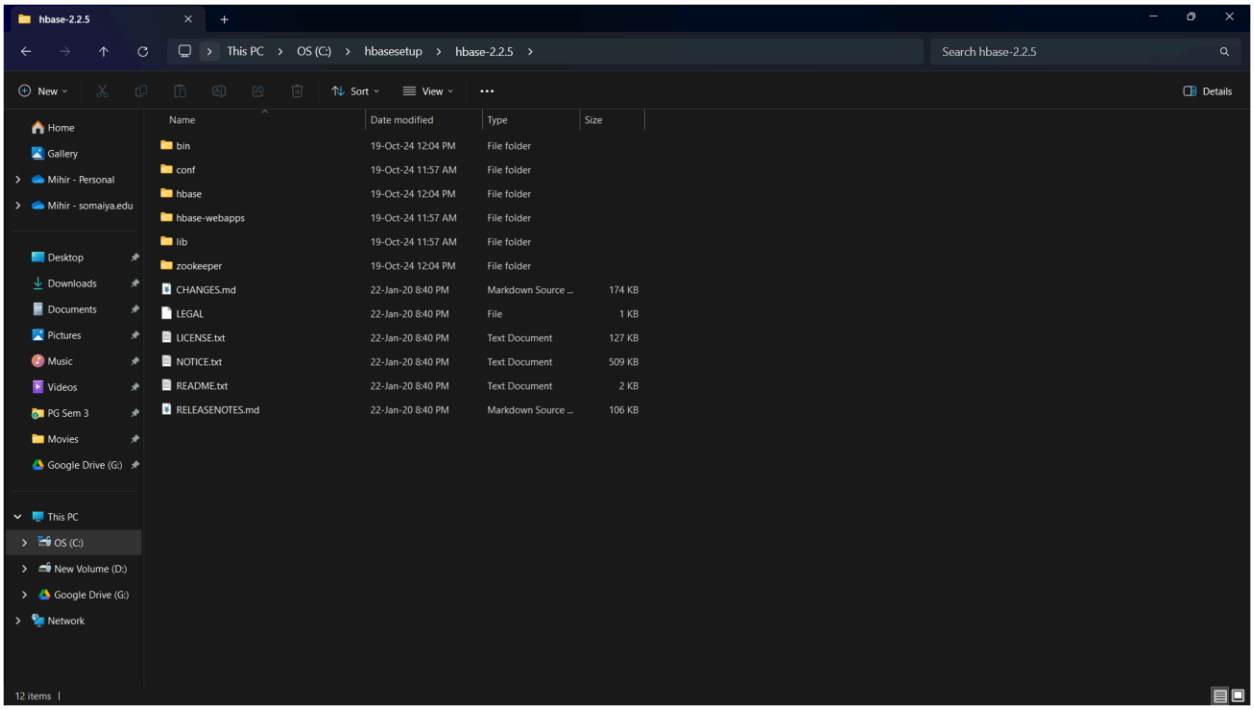
Aim: Download and installation of HBase

Steps:

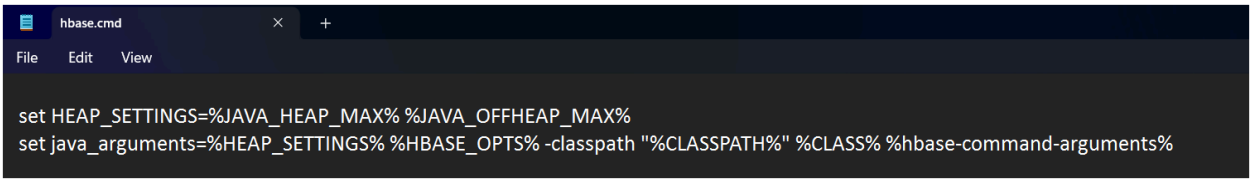
1- Download HBase bin file from here.

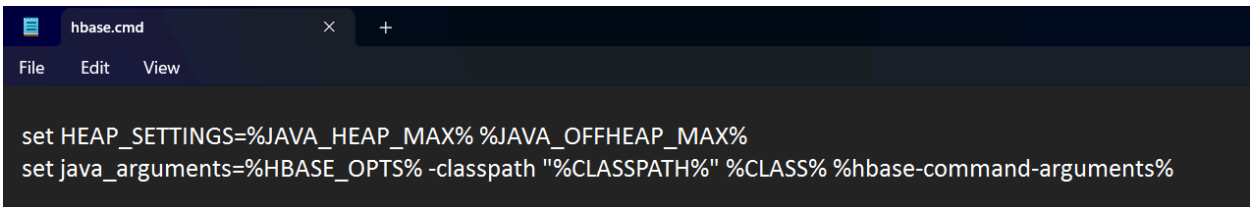


2- Create a new folder in C drive named hbasesetup. Extract the files and paste it in the hbasesetup. Create 2 new folders hbase and zookeeper inside.



3-Open the bin folder. Search for the file hbase.cmd and edit it in notepad. Search for java_arguments and remove %HEAP_SETTINGS%.

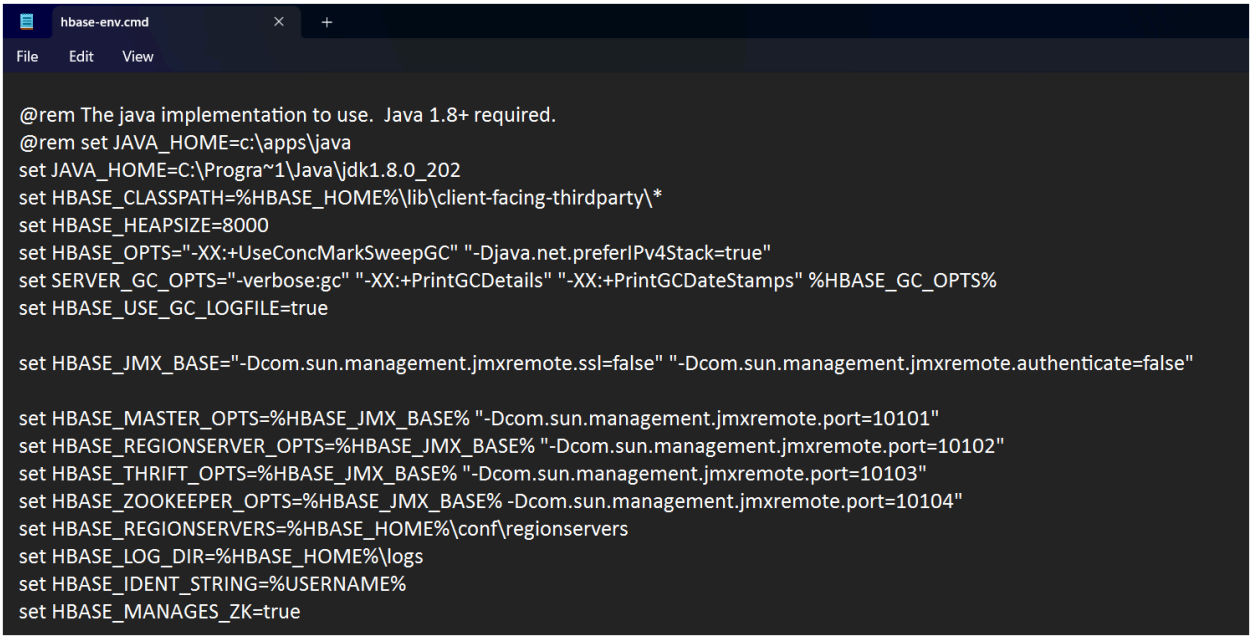




4-Open the conf folder and edit the file hbase-env.cmd in notepad. Add the following lines inside.

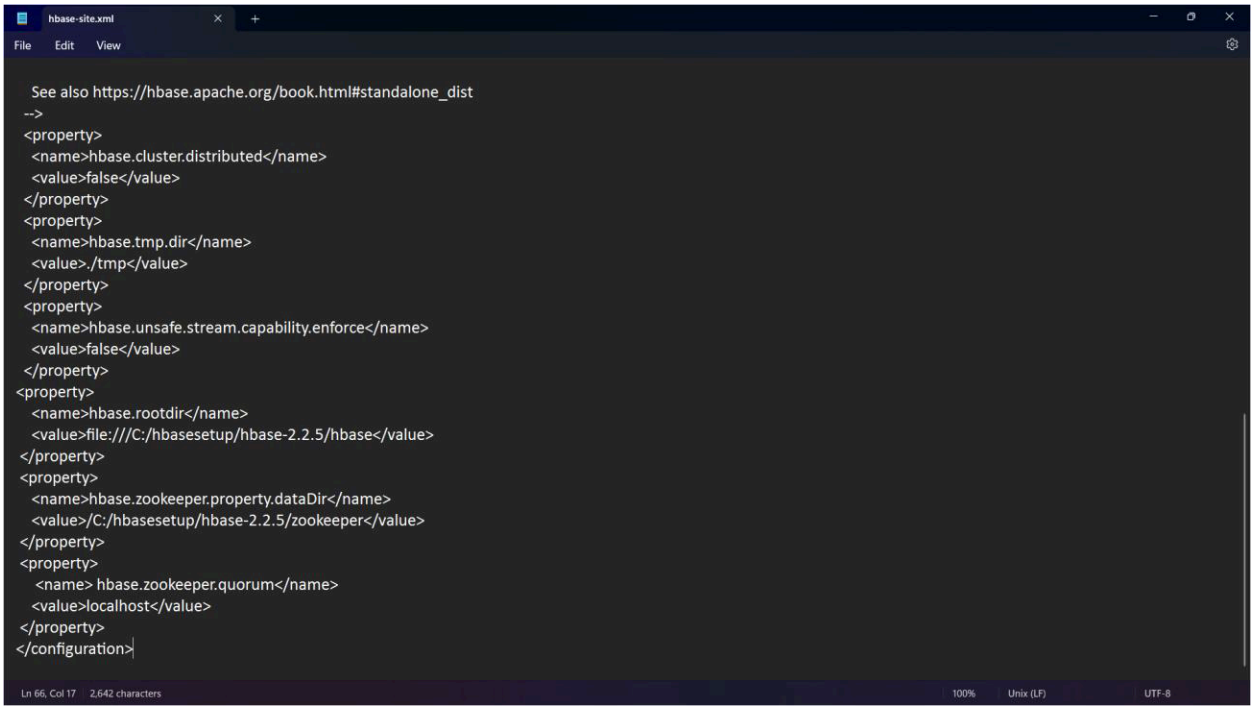
```
set JAVA_HOME=C:\Progra~1\Java\jdk1.8.0_202
set HBASE_CLASSPATH=%HBASE_HOME%\lib\client-facing-thirdparty\*
set HBASE_HEAPSIZE=8000
set HBASE_OPTS="-XX:+UseConcMarkSweepGC" "-Djava.net.preferIPv4Stack=true"
set SERVER_GC_OPTS="-verbose:gc" "-XX:+PrintGCDetails" "-XX:+PrintGCDateStamps"
%HBASE_GC_OPTS%
set HBASE_USE_GC_LOGFILE=true
```

```
set HBASE_JMX_BASE="-Dcom.sun.management.jmxremote.ssl=false"
"-Dcom.sun.management.jmxremote.authenticate=false"
set HBASE_MASTER_OPTS=%HBASE_JMX_BASE% "-Dcom.sun.management.jmxremote.port=10101"
set HBASE_REGIONSERVER_OPTS=%HBASE_JMX_BASE%
"-Dcom.sun.management.jmxremote.port=10102"
set HBASE_THRIFT_OPTS=%HBASE_JMX_BASE% "-Dcom.sun.management.jmxremote.port=10103"
set HBASE_ZOOKEEPER_OPTS=%HBASE_JMX_BASE%
-Dcom.sun.management.jmxremote.port=10104"
set HBASE_REGIONSERVERS=%HBASE_HOME%\conf\regionservers
set HBASE_LOG_DIR=%HBASE_HOME%\logs
set HBASE_IDENT_STRING=%USERNAME%
set HBASE_MANAGES_ZK=true
```

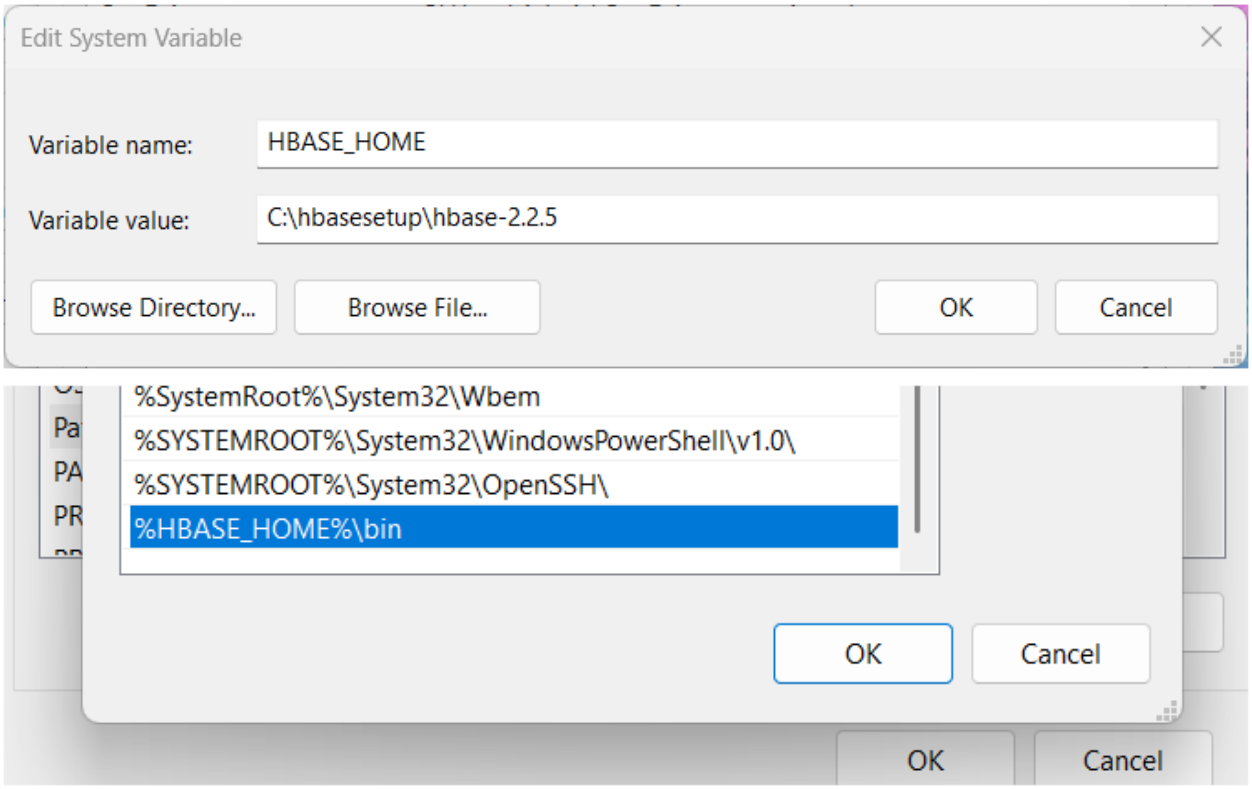


5-Open the conf folder and edit the file hbase-site.xml in notepad. Add the following lines after the last property tag.

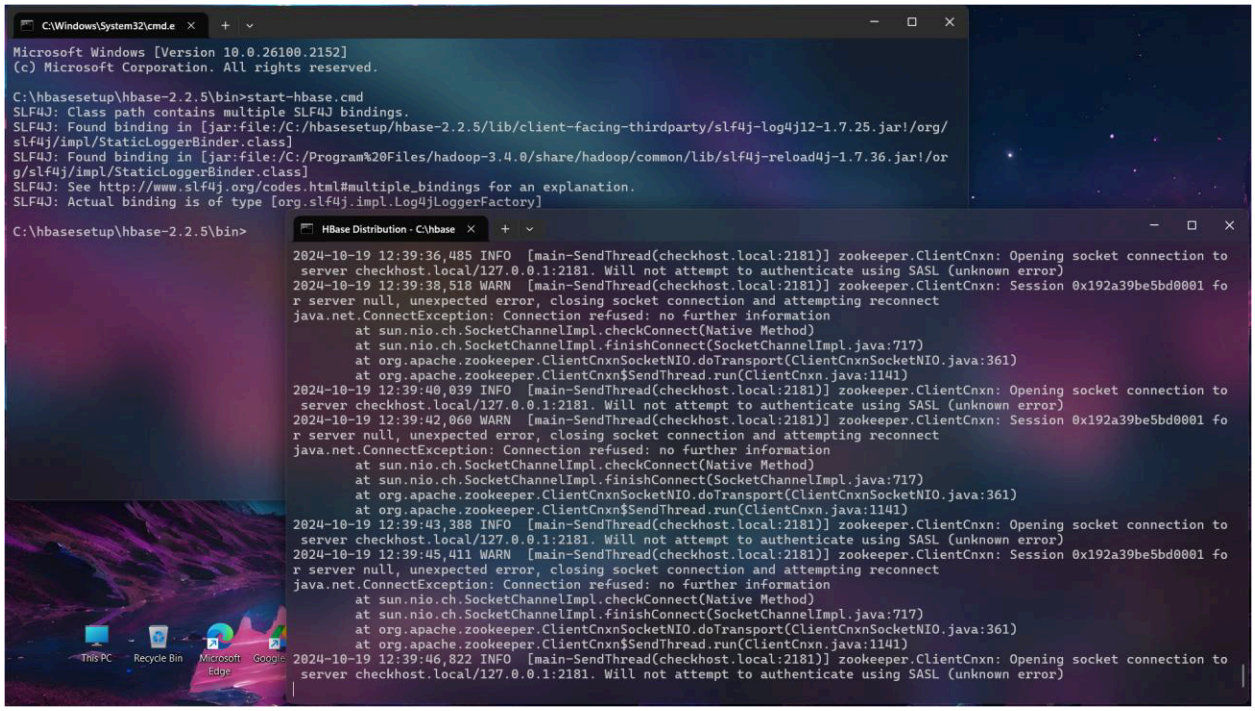
```
<property>
<name>hbase.rootdir</name>
<value>file:///C:/hbasesetup/hbase-2.2.5/hbase</value>
</property>
<property>
<name>hbase.zookeeper.property.dataDir</name>
<value>/C:/hbasesetup/hbase-2.2.5/zookeeper</value>
</property>
<property>
<name> hbase.zookeeper.quorum</name>
<value>localhost</value>
</property>
```



6-Set the HBase environment variables and it too path as well.

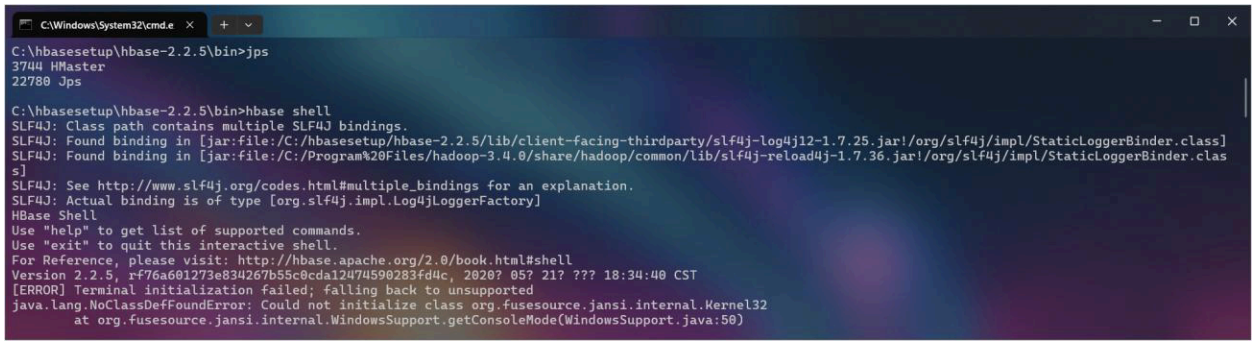


7-Open command prompt and navigate to the hbase bin folder. Run the command start-hbase.cmd to start hbase

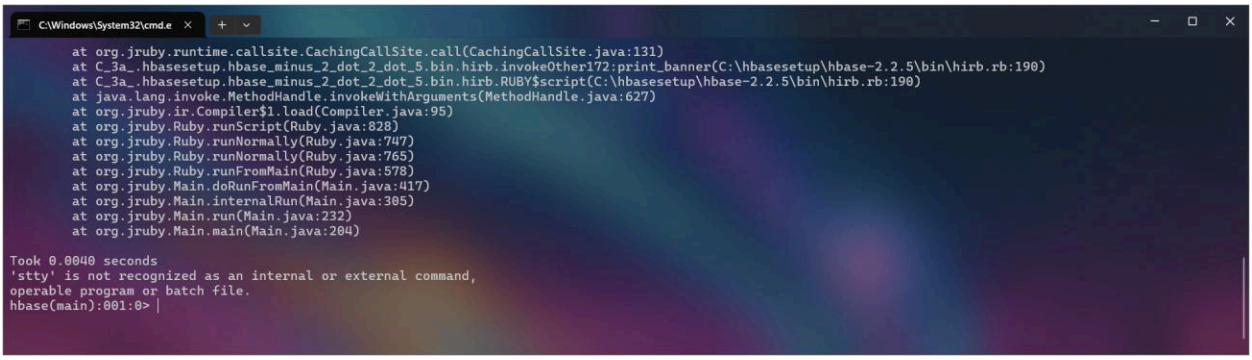


8-Using the command jps you can check that our HMaster is running.

9-Start the HBase Shell now with the command hbase shell. Initial startup may take some time.



10-Your HBase Shell has been started. Ignore the warnings received while starting the shell.


C:\Windows\System32\cmd.e
at org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:131)
at C_3a_.hbasesetup.hbase_minus_2_dot_2_dot_5.bin.hirb.invokeOther172:print_banner(C:\hbasesetup\hbase-2.2.5\bin\hirb.rb:190)
at C_3a_.hbasesetup.hbase_minus_2_dot_2_dot_5.bin.hirb.RUBY$script(C:\hbasesetup\hbase-2.2.5\bin\hirb.rb:190)
at java.lang.invoke.MethodHandle.invokeWithArguments(MethodHandle.java:627)
at org.jruby.ir.Compiler$1.load(Compiler.java:95)
at org.jruby.Ruby.runScript(Ruby.java:828)
at org.jruby.Ruby.runNormally(Ruby.java:747)
at org.jruby.Ruby.runNormally(Ruby.java:765)
at org.jruby.Ruby.runFromMain(Ruby.java:578)
at org.jruby.Main.doRunFromMain(Main.java:417)
at org.jruby.Main.internalRun(Main.java:305)
at org.jruby.Main.run(Main.java:232)
at org.jruby.Main.main(Main.java:204)
Took 0.0040 seconds
'stty' is not recognized as an internal or external command,
operable program or batch file.
hbase(main):001:0>