Assisted Practice 20.1: Implementation of a Simple GraphX

Problem Scenario: Create a graph object with six friends from different age groups who are connected through social media

Objective: In this demonstration, you will learn to implement a graph.

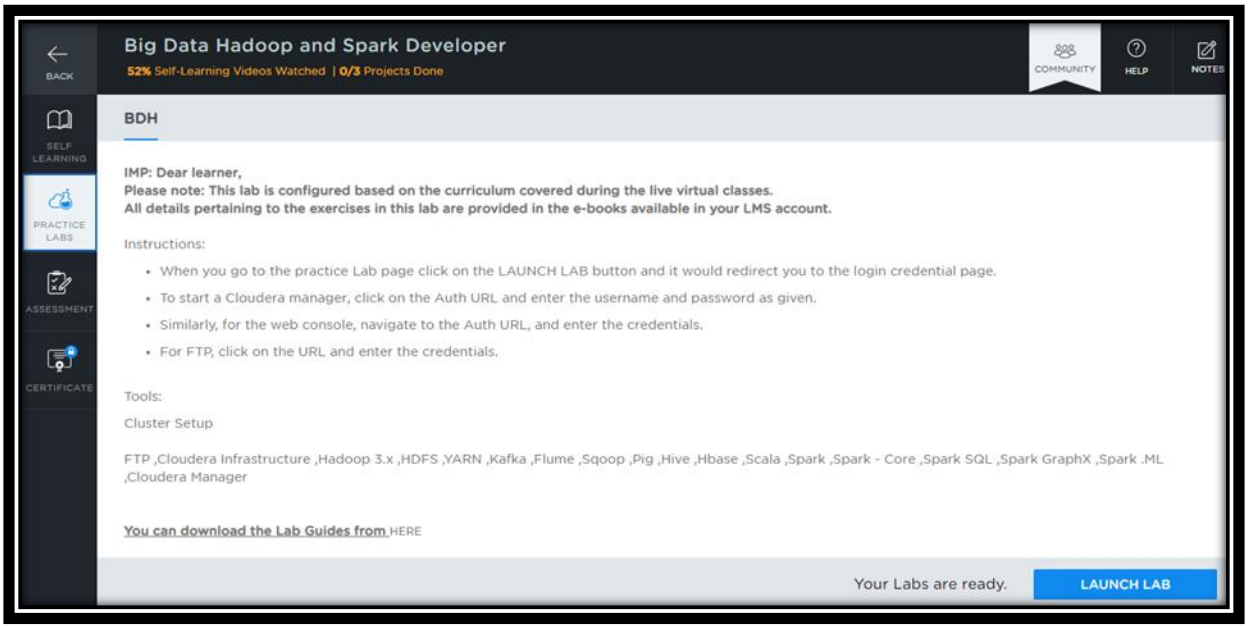
Steps to perform:

Step 1: Import the necessary libraries after logging in to the Spark environment

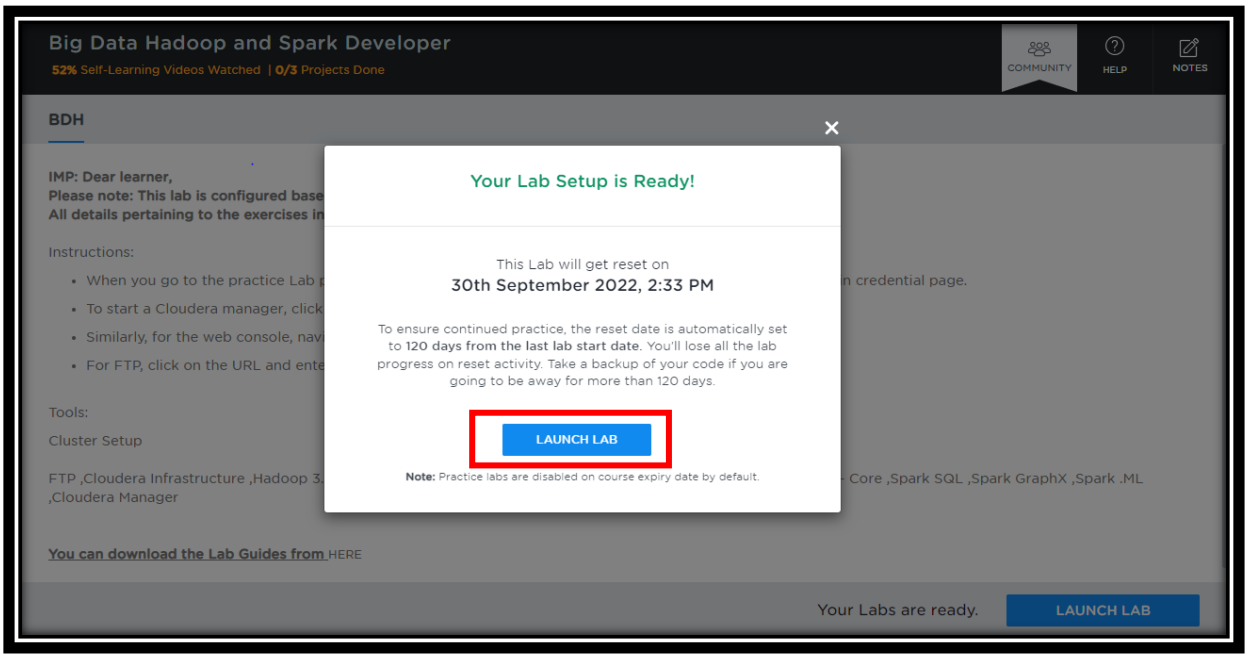
1.1 Login to your LMS account

1.2 Open the course **"Big Data Hadoop and Spark developer"**

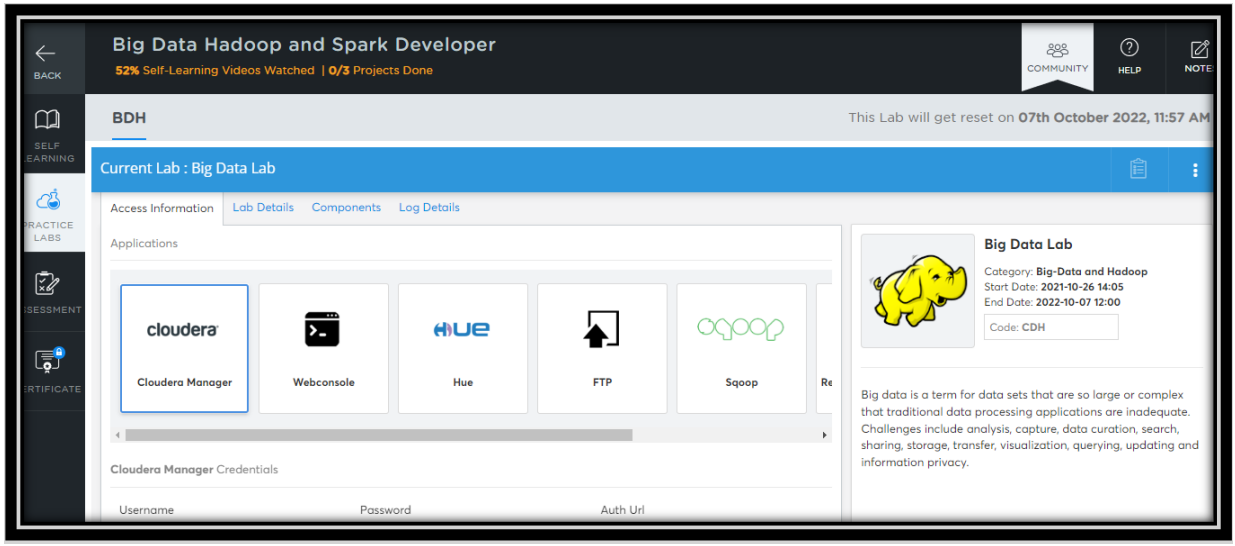
1.3 On the left side, click on the **"PRACTICE LABS"** tab and on the **"LAUNCH LAB"** button



1.4 Again, click on the **“LAUNCH LAB”** button



1.4 Click on **“Web console”** and on the **“Auth Url”** to upload the files and copy the **“Username”** and the **“Password”**



1.5 Login to the spark environment

Command:

Spark3-shell

```
[testdemomay1301mailinator@bdh-cluster2-edgenode10 ~]$ spark3-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/06/10 11:41:02 WARN util.Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
22/06/10 11:41:02 WARN util.Utils: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.
22/06/10 11:41:02 WARN util.Utils: Service 'SparkUI' could not bind on port 4042. Attempting port 4043.
22/06/10 11:41:02 WARN util.Utils: Service 'SparkUI' could not bind on port 4043. Attempting port 4044.
22/06/10 11:41:02 WARN util.Utils: Service 'SparkUI' could not bind on port 4044. Attempting port 4045.
Spark context Web UI available at http://bdh-cluster2-edgenode10.bdh-env.gne4-rutx.cloudera.site:4045
Spark context available as 'sc' (master = local[*], app id = local-1654861263023).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.1.2.7.2.12.4-1
      /_/

Using Scala version 2.12.10 (OpenJDK 64-Bit Server VM, Java 1.8.0_312)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

1.6 Import the necessary libraries

Command:

```
import org.apache.spark._

import org.apache.spark.rdd.RDD

import org.apache.spark.util.IntParam

import org.apache.spark.graphx._

import org.apache.spark.graphx.util.GraphGenerators
```

```
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.util.IntParam
import org.apache.spark.util.IntParam

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

scala> import org.apache.spark.graphx.util.GraphGenerators
import org.apache.spark.graphx.util.GraphGenerators
```

Step 2: Create a vertex array that contains the ID, name of the friend, and age

Command:

val vertexArray = Array((1L, ("Leonard", 28)),(2L, ("Penney", 29)),(3L, ("Sheldon", 25)),(4L, ("Amy", 46)),(5L, ("Howard", 55)),(6L, ("Bernie",50)))

```
scala> val vertexArray = Array((1L, ("Leonard", 28)),(2L, ("Penney", 29)),(3L, ("Sheldon", 25)),(4L, ("Amy", 46)),(5L, ("Howard", 55)),(6L, (
"Bernie",50)))
vertexArray: Array[(Long, (String, Int))] = Array((1,(Leonard,28)), (2,(Penney,29)), (3,(Sheldon,25)), (4,(Amy,46)), (5,(Howard,55)), (6,(Ber
nie,50)))
```

Command:

val vertexRDD: RDD[(Long, (String, Int))] = sc.parallelize(vertexArray)

```
scala> val vertexRDD: RDD[(Long, (String, Int))] = sc.parallelize(vertexArray)
vertexRDD: org.apache.spark.rdd.RDD[(Long, (String, Int))] = ParallelCollectionRDD[0] at parallelize at <console>:35
```

Step 3: Create an edge array

Command:

val edgeArray = Array(Edge(2L, 1L, 7),Edge(2L, 4L, 2),Edge(3L, 2L, 4),Edge(3L, 6L, 3),Edge(4L, 1L, 1),Edge(5L, 2L, 2),Edge(5L, 3L, 8),Edge(5L, 6L, 3))

```
scala> val edgeArray = Array(Edge(2L, 1L, 7),Edge(2L, 4L, 2),Edge(3L, 2L, 4),Edge(3L, 6L, 3),Edge(4L, 1L, 1),Edge(5L, 2L, 2),Edge(5L, 3L, 8))

edgeArray: Array[org.apache.spark.graphx.Edge[Int]] = Array(Edge(2,1,7), Edge(2,4,2), Edge(3,2,4), Edge(3,6,3), Edge(4,1,1), Edge(5,2,2), Edg
e(5,3,8))
```

Command:

val edgeRDD: RDD[Edge[Int]] = sc.parallelize(edgeArray)

```
scala> val edgeRDD: RDD[Edge[Int]] = sc.parallelize(edgeArray)
edgeRDD: org.apache.spark.rdd.RDD[org.apache.spark.graphx.Edge[Int]] = ParallelCollectionRDD[1] at parallelize at <console>:35
```

Step 4: Create a graph that contains vertices whose age is below 35

Command:

val graph: Graph[(String, Int), Int] = Graph(vertexRDD, edgeRDD)

graph.vertices.filter { case (id, (name, age)) => age < 35 }

.collect.foreach { case (id, (name, age)) => println(s"$name is $age")}

Step 5: Display the data

```
scala> graph.vertices.filter { case (id, (name, age)) => age < 35 }
res2: org.apache.spark.graphx.VertexRDD[(String, Int)] = VertexRDDImpl[31] at RDD at VertexRDD.scala:57

scala> .collect.foreach { case (id, (name, age)) => println(s"$name is $age")}
Leonard is 28
Penney is 29
Sheldon is 25
```