CSCI 5408

Data Management, Warehousing, And Analytics

Assignment 3 - Problem 1

Perform research on NoSQL and data processing.

Prepared By

Bhavisha Oza (B00935827)

Problem-1:

Problem 1A: Reuter News Data Reading & Transformation and storing in MogoDb

1. Summary:

The task is to create a MongoDB database named "ReuterDb" using a Java program called "ReutRead.java" that processes two given news files (reut2-009.sgm and reut2-014.sgm). Each document in the database will represent a news article and must contain the extracted text between <REUTERS></REUTERS>, <TEXT></TEXT>, and <TITLE></TITLE> tags.

The program first reads the content of the two input files and merges them into a single string. It then uses regular expressions to extract data between <REUTERS> tags and further extracts text between <TITLE> and <TEXT> tags for each Reuters article. For each article, if both title and text content are present, it creates a MongoDB document and inserts it into the "ReuterDb" collection.

To visualize the program's logic, a flowchart and algorithm is included in the PDF file. The flowchart illustrates the sequence of steps, decision points, and data processing, while the algorithm provides a high-level outline of the program's functioning.

Requirements & Tasks fulfilled:

1. Create an account of MongoDB Atlas [3, 7]

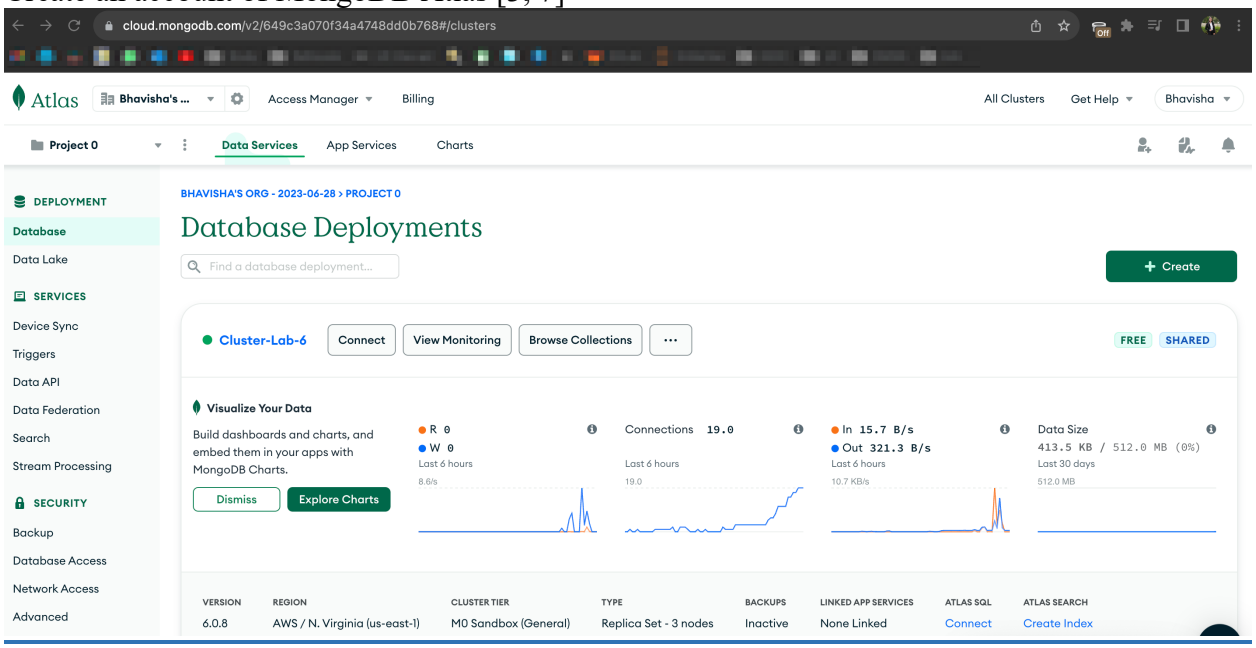

Figure 1: MongoDB atlas account creation.

2. Create a new cluster [7].
3. Cloud Provider & Region.
   Enter the name which you want to give to your cluster [7].

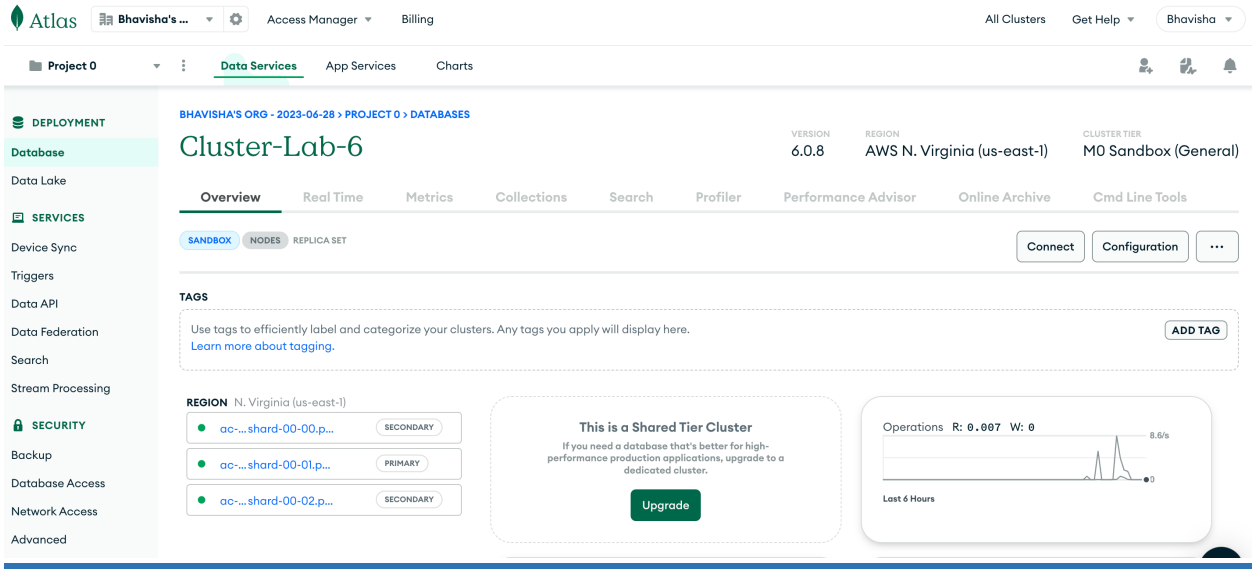

Figure 2: Cluster creation.

4. Create a java program, which creates MongoDb Database – ReuterDb from the two given news files (reut2-009.sgm, and reut2-014.sgm), where each Document contains a news article [1].

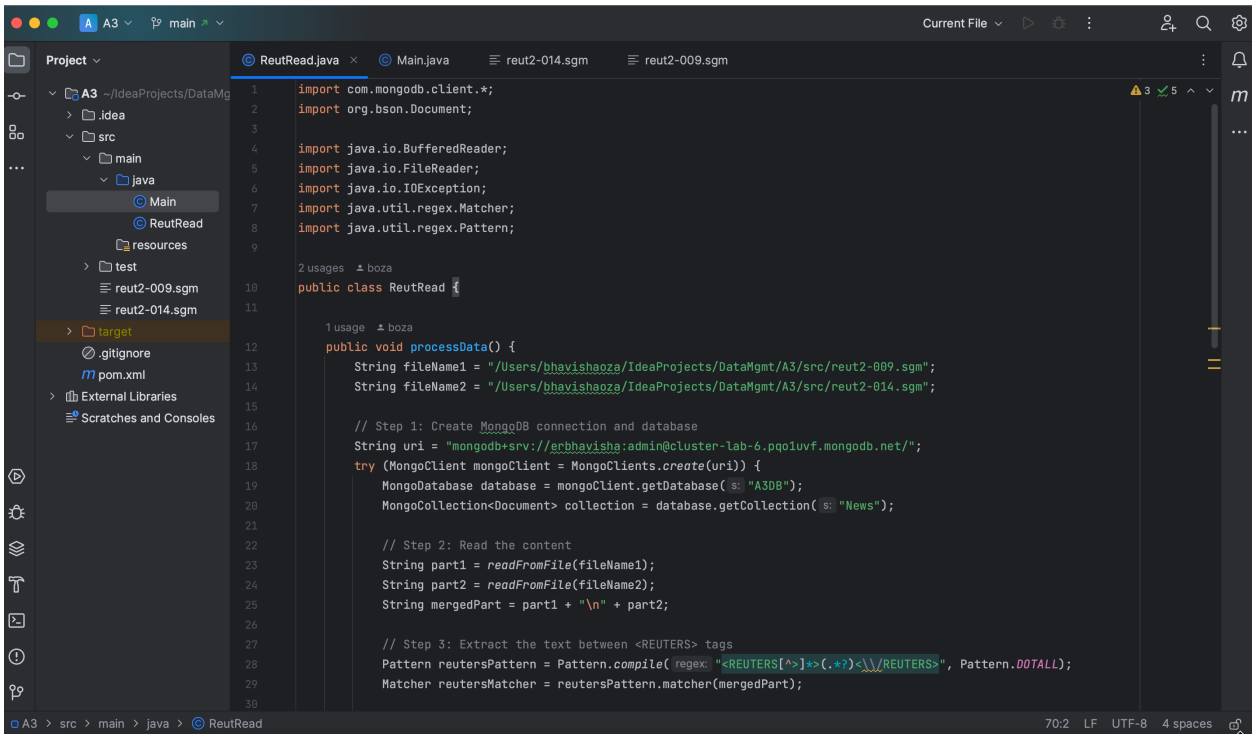

Figure 3: Java program for processes data from .sgm files and inserts article titles and texts into a MongoDB collection.

Program Output:

- IntelliJ

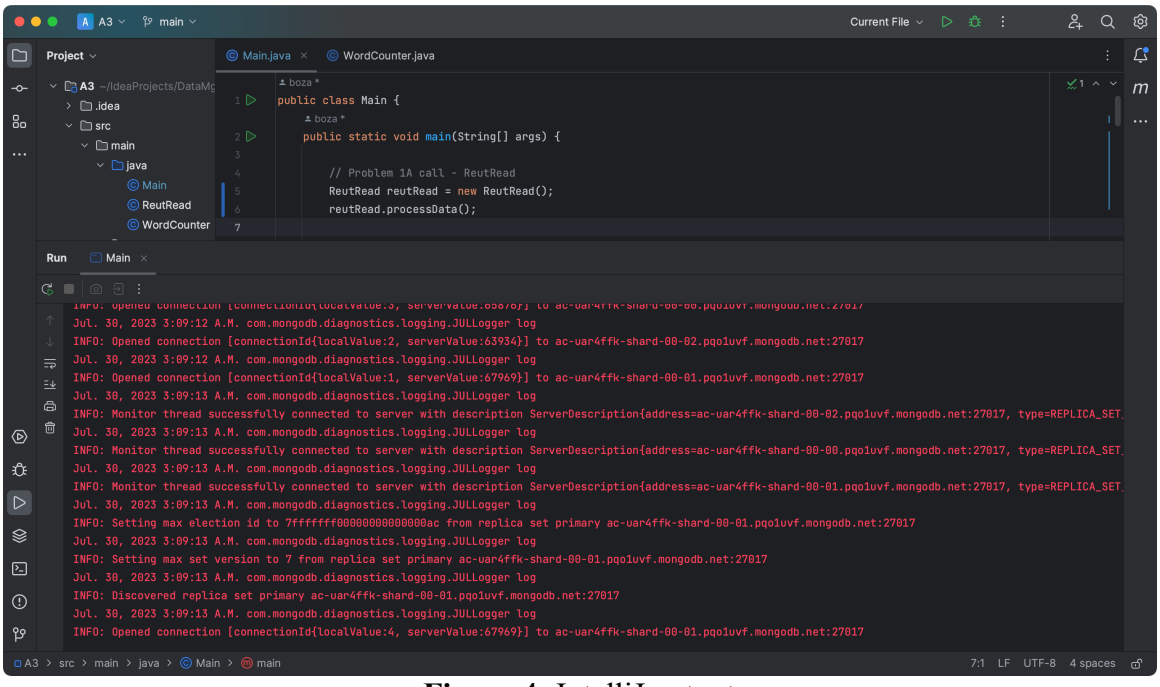

Figure 4: IntelliJ output

- MongoDB Compass

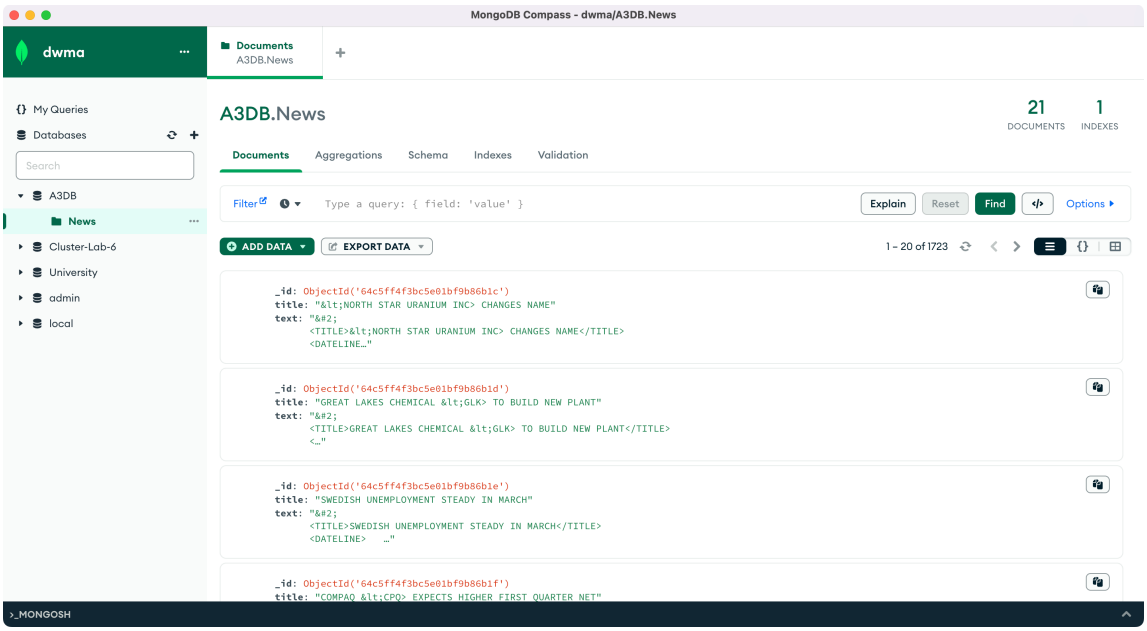

Figure 5: MongoDB Compass output

5. An algorithm of Reuters Data cleaning/transformation program

- Start the program execution.
- Define the `ReutRead` class with the following methods:
  - `processData()`: This method will handle the data cleaning and transformation process.
- Inside the `processData()` method:
  - Define two variables `fileName1` and `fileName2` to store the paths of the input .sgm files.
  - Create a MongoDB connection using the provided connection URI.
  - Get a reference to the database named "A3DB" and the collection named "News" in MongoDB.
  - Read the content of both input .sgm files (`fileName1` and `fileName2`) using the `readFromFile()` method and store the content in strings `part1` and `part2`.
  - Merge `part1` and `part2` to form `mergedPart`, which will contain the entire data from both files as a single string.
- Use regular expressions to extract data between the `<REUTERS>` tags in `mergedPart`.
  - Define a regex pattern `reutersPattern` to match content between `<REUTERS>` and `</REUTERS>` tags using `Pattern.compile()`.
  - Create a matcher `reutersMatcher` using `reutersPattern.matcher(mergedPart)`.
- Use regular expressions to extract text between `<TITLE>` and `<TEXT>` tags for each Reuters article in `reutersMatcher`.
  - Define a regex pattern `titlePattern` to match content between `<TITLE>` and `</TITLE>` tags using `Pattern.compile()`.
  - Define a regex pattern `textPattern` to match content between `<TEXT>` and `</TEXT>` tags using `Pattern.compile()`.

- Iterate through `reutersMatcher` and for each Reuters article:
    - Extract the article's content into a `reutersArticle` string.
    - Create a matcher `titleMatcher` using `titlePattern.matcher(reutersArticle)`.
    - Create a matcher `textMatcher` using `textPattern.matcher(reutersArticle)`.
- If both `titleMatcher` and `textMatcher` find a match (i.e., if there are both title and text content present), proceed to the next steps. Otherwise, skip the current article.
- Extract the `title` and `text` content using `titleMatcher.group(1)` and `textMatcher.group(1)` respectively.
- Create a MongoDB `Document` and add the extracted `title` and `text` as key-value pairs.
- Insert the created `Document` into the MongoDB collection "News" using `collection.insertOne(document)`.
- Repeat steps 6 to 10 until there are no more Reuters articles in `reutersMatcher`.
- Close the MongoDB connection.
- End the program execution.

6. The flowchart of Reuters Data cleaning/transformation program created using darw.io tool [8].

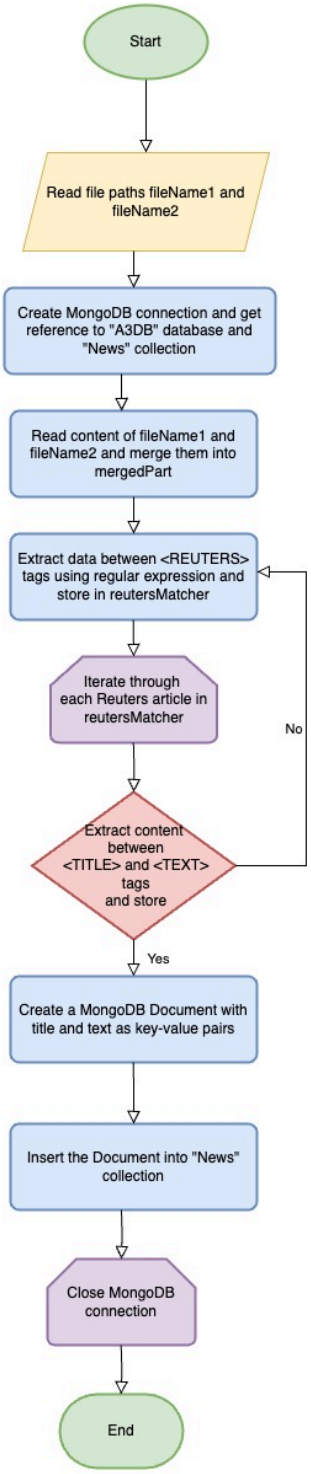

Figure 6: Flowchart of Reuters Data cleaning/transformation program

Problem 1B: Reuter News Data Processing using Spark.

1. Summary:

In this project, the main objective is to configure and initialize an Apache Spark cluster using the GCP (Google Cloud Platform) cloud account. The process involves following the provided tutorials in the lab session to set up the cluster successfully. The written explanation of the steps taken to complete the task are included in the description.

Once the cluster is set up, a MapReduce program is implemented using Java (WordCounter.java Engine). The program's purpose is to count the unique words and their frequencies in the file "reut2-009.sgm" and represent the words with the highest and lowest frequencies as an outcome of the MapReduce program.

TASK-1 Configure and initialize Apache Spark cluster using GCP cloud account [2].

Requirements & Tasks fulfilled to configure & initialize Apache Spark cluster using GCP:

1. Create an account on GCP, search for dataproc [2, 6]:

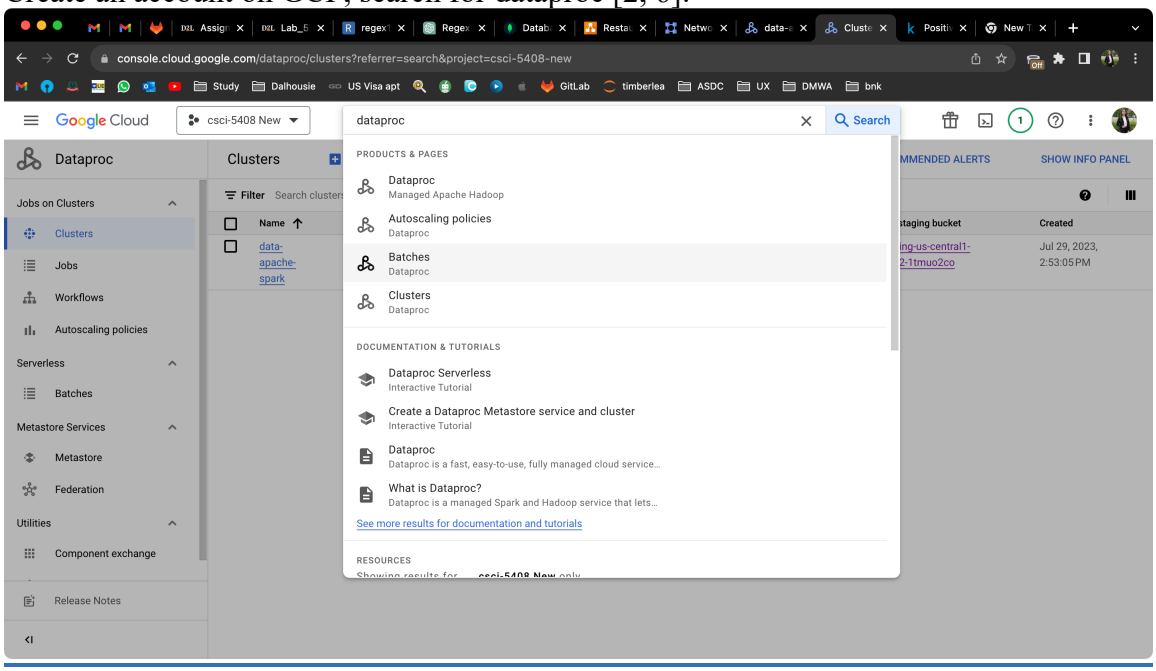

Figure 7: Searching dataproc.

2. Click on create cluster, select cluster on Compute Engine [6].

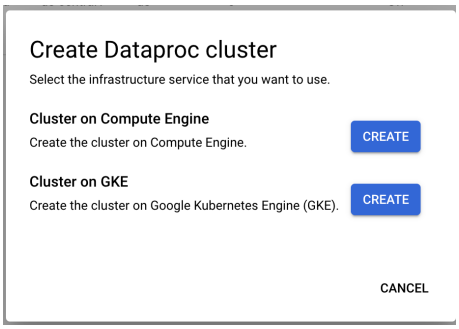

Figure 8: Creating cluster on Compute Engine

3. Create cluster by selecting following options:
   Enter cluster name as data-apache-spark
   In Location, select us-central1-c from zone dropdown
   select cluster type as Single node [6].

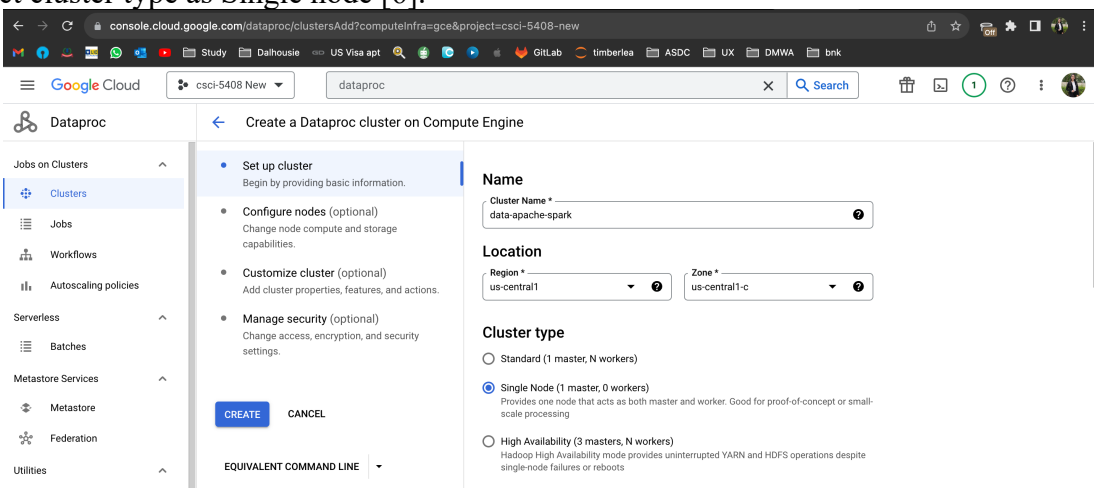

Figure 9: Setup selection for cluster

4. Start the cluster.

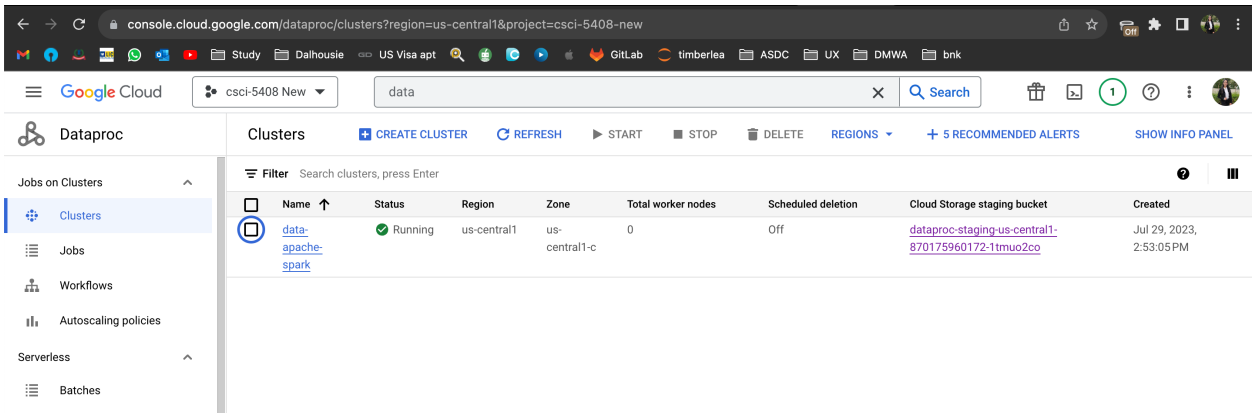

Figure 10: Apache spark cluster running.

5. Open the cluster and Go to VM instances and click on created instance.

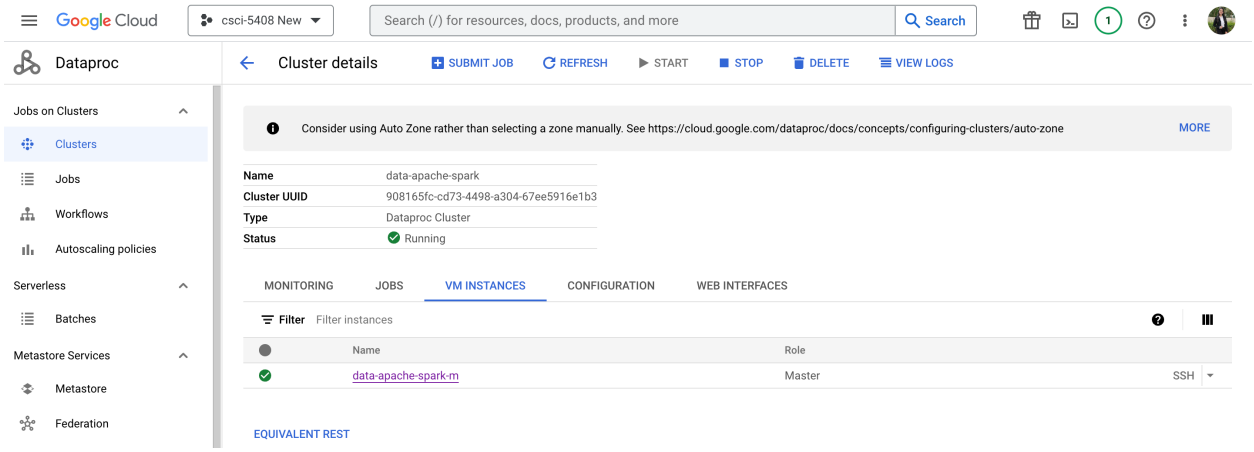

Figure 11: Checking created instance.

6. From the details page, click on edit icon, go to the Firewalls section, and check the checkbox of Allow HTTP traffic and Allow HTTPs traffic and save [6].

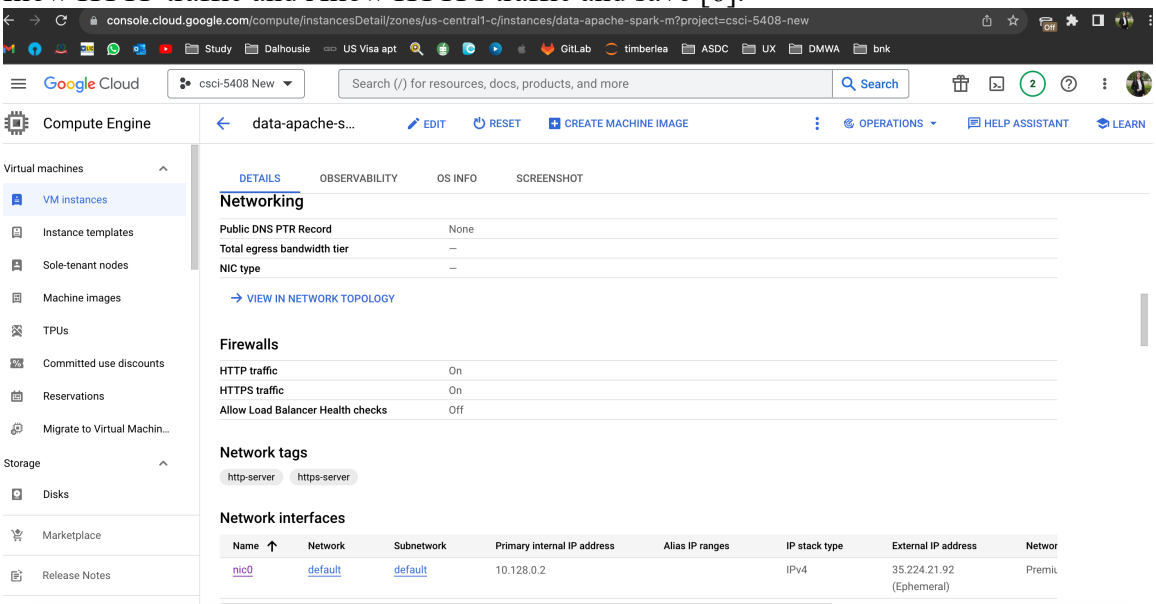

Figure 12: Configuring the instance.

➔ **Figure 11** displays the created virtual machine (VM) instance data-apache-spark under the project csci-5408 New [6].

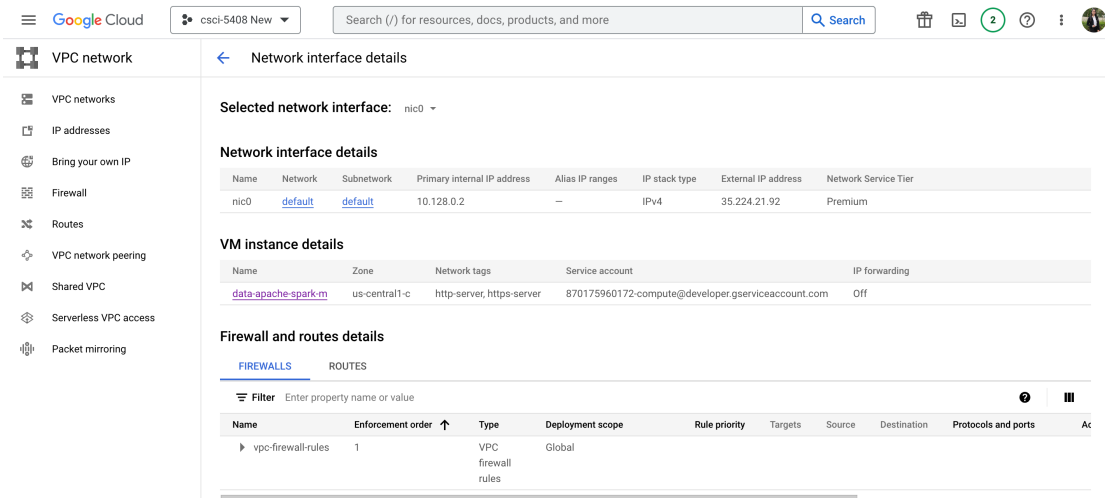

Figure 13: data-apache-spark VM hosted on GCP.

7. Click on SSH from the VM instance menu to connect to the shell [6].

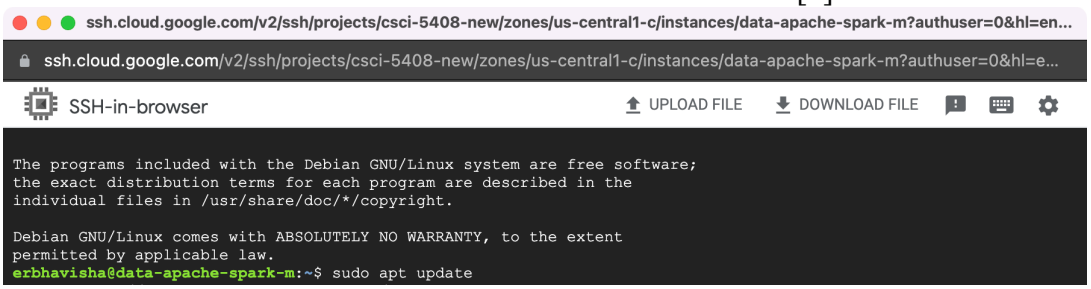

Figure 14: data-apache-spark VM connected to shell.

8. Update OS using command sudo apt update as it is new instance.

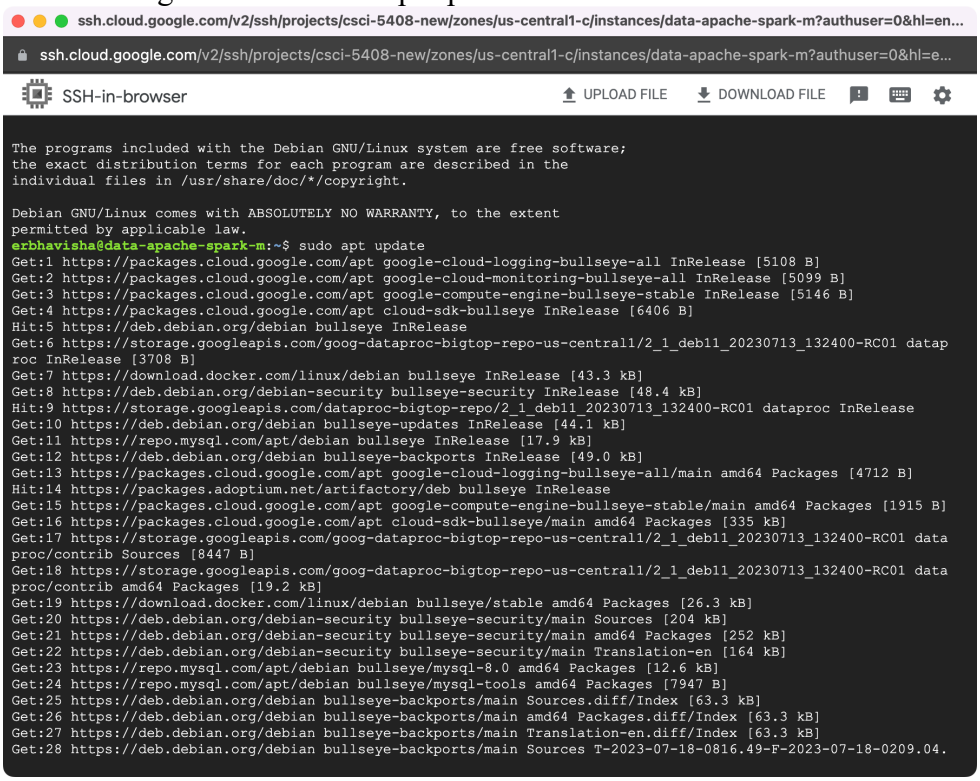

Figure 15: OS update for new instance

9. Install java and scala for the Apache Spark requirement [4]
   To install Java: sudo apt-get install default-jdk -y
   To install Scala: sudo apt-get install scala -y

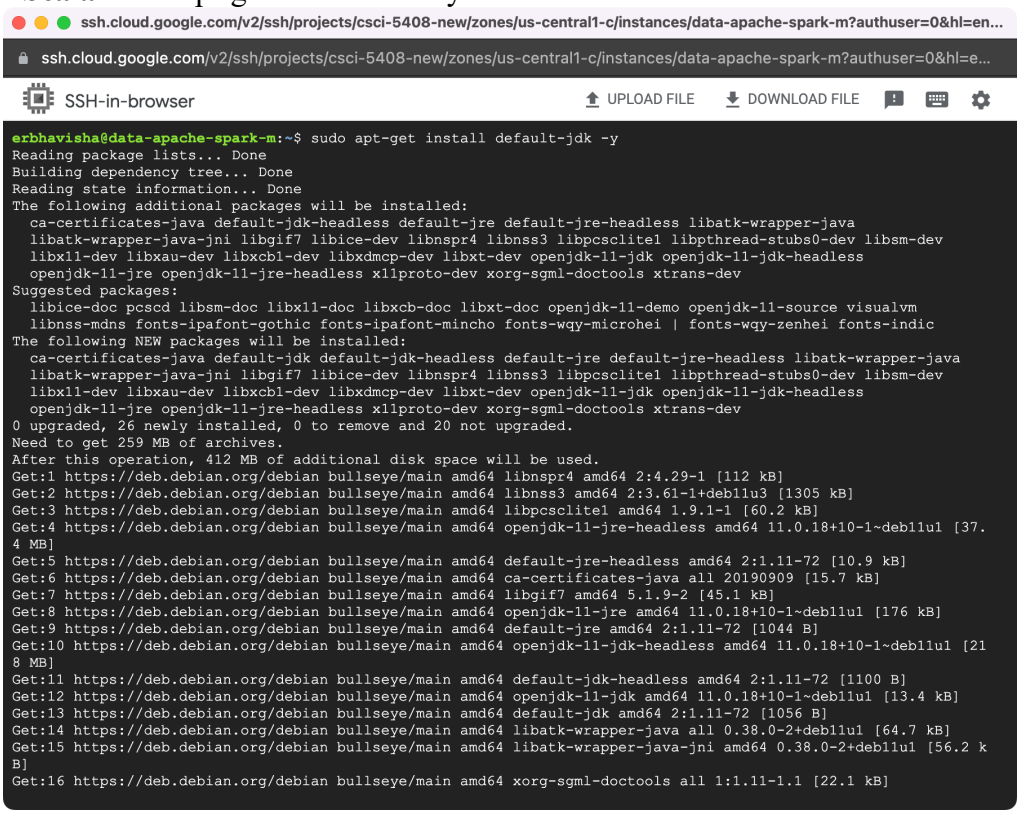

Figure 16: Installing Java

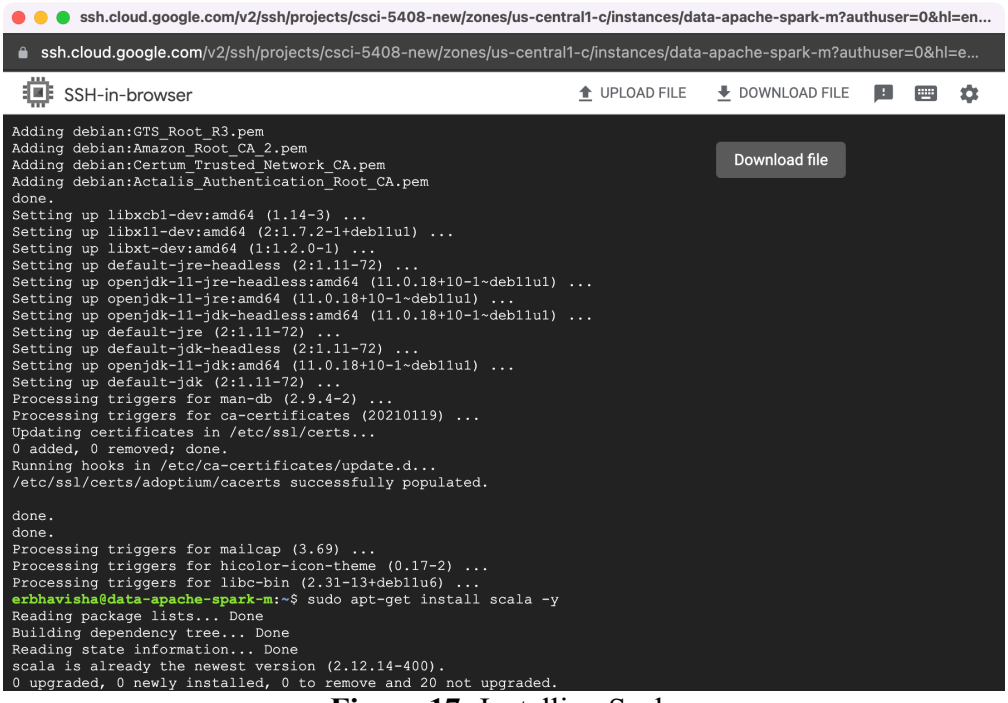

Figure 17: Installing Scala

10. After installation, check the version using below commands [4]
    java -version
    scala -version.

```
erbhavisha@data-apache-spark-m:~$ java -version
openjdk version "11.0.19" 2023-04-18
OpenJDK Runtime Environment Temurin-11.0.19+7 (build 11.0.19+7)
OpenJDK 64-Bit Server VM Temurin-11.0.19+7 (build 11.0.19+7, mixed mode)
erbhavisha@data-apache-spark-m:~$ scala -version
Scala code runner version 2.12.14 -- Copyright 2002-2021, LAMP/EPFL and Lightbend, Inc.
erbhavisha@data-apache-spark-m:~$
```

Figure 18: Java and Scala version check

11. Test Spark Shell [4]

```
erbhavisha@data-apache-spark-m:~$ sudo /opt/spark/bin/spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/07/30 04:46:17 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builti
n-java classes where applicable
Spark context Web UI available at http://data-apache-spark-m.us-central1-c.c.csci-5408-new.internal:4040
Spark context available as 'sc' (master = local[*], app id = local-1690692378575).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.4.1
      /_/

Using Scala version 2.12.17 (OpenJDK 64-Bit Server VM, Java 11.0.19)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

Figure 19: Spark shell check

12. Write a MapReduce program using Java (WordCounter.java Engine) to count (frequency count) the unique words found in "reut2-009.sgm".

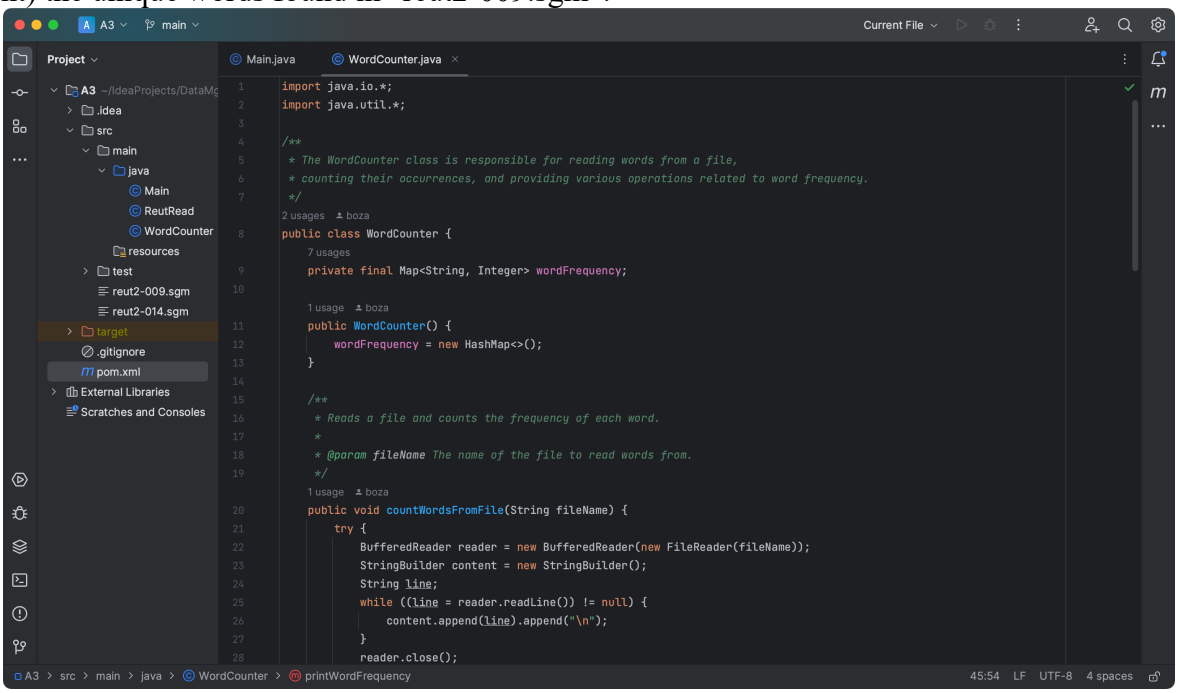

Figure 20: Java program for reading words from a file, counting their occurrences, and providing various operations related to word frequency.

13. Program output in IntelliJ

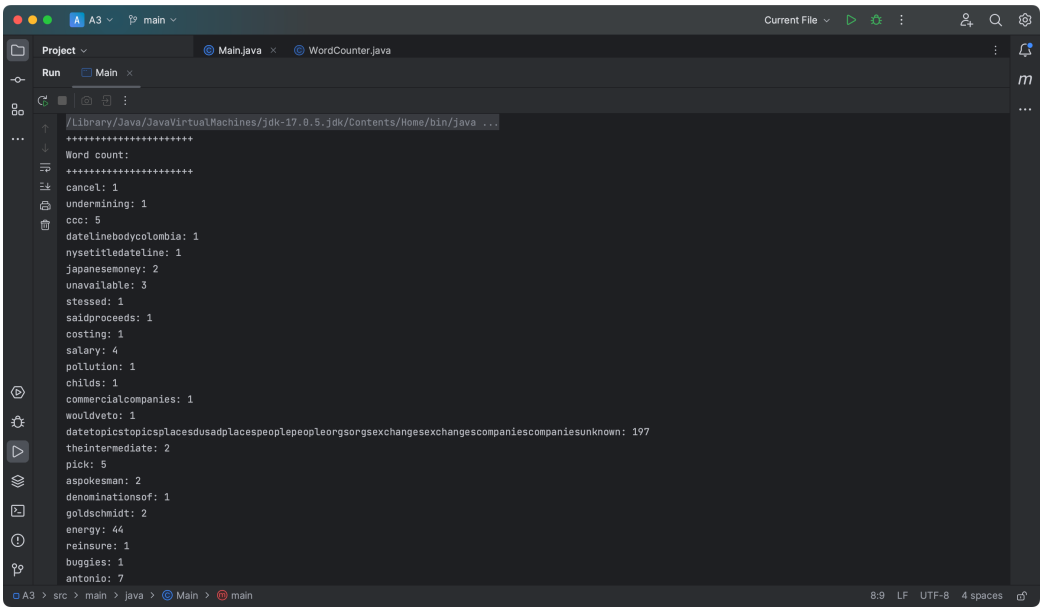

Figure 21: IntelliJ output -1.

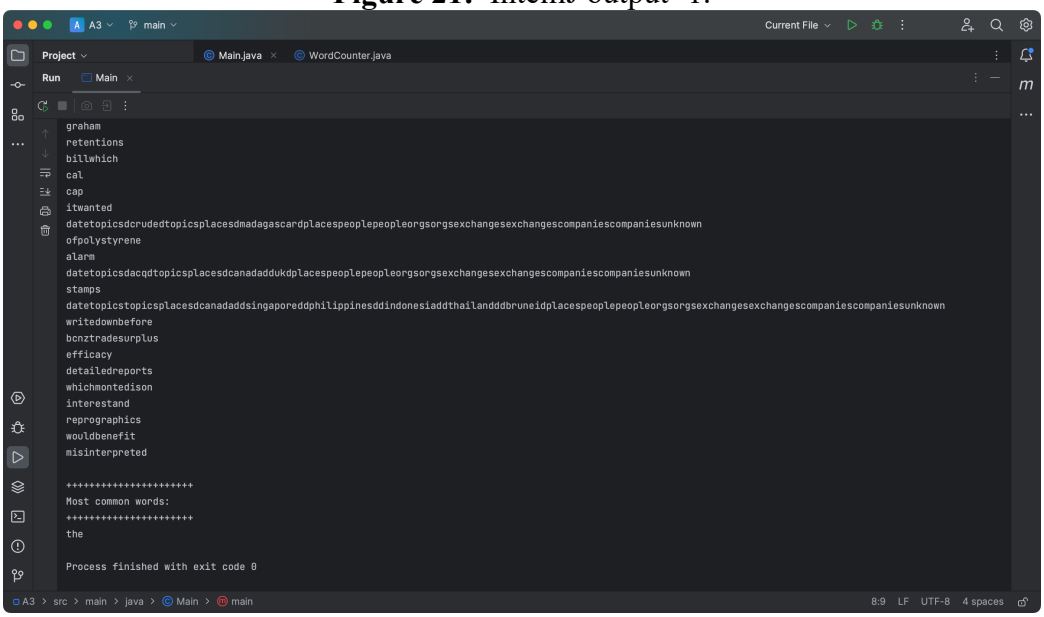

Figure 22: IntelliJ output -2

14. Program output in GCP

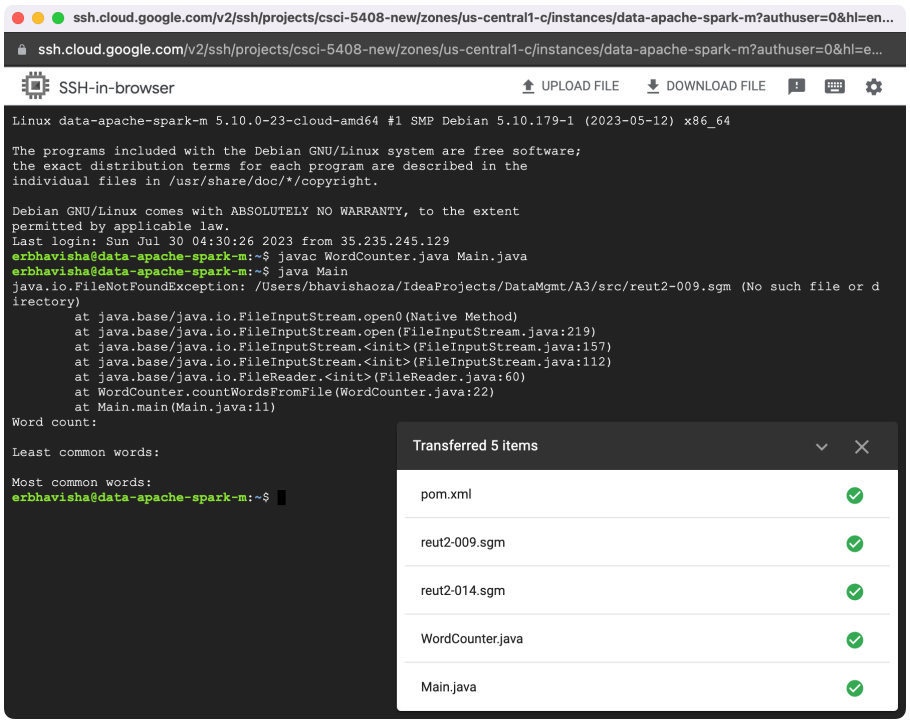

Figure 23: GCP output

15. An algorithm of MapReduce program:

1. Start the WordCounter class.
2. Declare a private instance variable `wordFrequency` of type `Map<String, Integer>` to store the frequency count of each word.
3. Implement the constructor `WordCounter()` to initialize the `wordFrequency` map.
4. Implement the method `countWordsFromFile(String fileName)` to read the content of the file and count the frequency of each word. The method takes the file name as input.
   - Open the file using a BufferedReader.
   - Read the content line by line, appending it to a StringBuilder.
   - Close the BufferedReader.
   - Remove all non-alphabetic characters and convert the content to lowercase.
   - Split the content into individual words using whitespace as the delimiter.
   - For each word, update its frequency in the `wordFrequency` map.
5. Implement the method `printWordFrequency()` to print the word count.
   - Iterate through the `wordFrequency` map and print each word along with its frequency.
6. Implement the method `getLeastCommonWords()` to find the least common words.
   - Initialize a variable `minCount` to Integer.MAX_VALUE.
   - Iterate through the values of the `wordFrequency` map and update `minCount` with the minimum frequency found.
   - Return an array of words that have a frequency equal to `minCount`.
7. Implement the method `getMostCommonWords()` to find the most common words.
   - Initialize a variable `maxCount` to Integer.MIN_VALUE.

- Iterate through the values of the `wordFrequency` map and update `maxCount` with the maximum frequency found.
- Return an array of words that have a frequency equal to `maxCount`.

8. Implement the helper method `getWordsWithFrequency(int frequency)` to retrieve words with a specific frequency.
   - Use stream operations to filter the `wordFrequency` map and collect words with the given `frequency`.
9. In the `main` method:
   - Create an instance of the WordCounter class.
   - Call the `countWordsFromFile(fileName)` method to read the file and count words.
   - Call the `printWordFrequency()` method to print the word count.
   - Call the `getLeastCommonWords()` method to find the least common words and print them.
   - Call the `getMostCommonWords()` method to find the most common words and print them.

16. The words that have highest frequencies:
    ++++++++++++++++++++++
    Most common words:
    ++++++++++++++++++++++
    the: 6294
17. The words that have lowest frequencies:
    ++++++++++++++++++++++
    Least common words:
    ++++++++++++++++++++++
    cancel: 1
    undermining: 1
    datelinebodycolombia: 1
    nysetitledateline: 1
    stessed: 1
    saidproceeds: 1
    costing: 1
    pollution: 1
    childs: 1
    commercialcompanies: 1
    wouldveto: 1
    denominationsof: 1
    reinsure: 1
    buggies: 1
    spreading: 1
    astudebaker: 1
    carriers: 1
    bcpanterasltpanttob: 1
    packagetitleauthor: 1
    averagedaily: 1
    newemployment: 1

orcombination: 1
nonrecurring: 1
roll: 1
unknowntexttitlepenobscot: 1
wilsonis: 1
bbbminus: 1
athreefortwo: 1
fromtermination: 1
postings: 1
planpreviously: 1
thedepartments: 1
bcnmspharmaceuticalltn: 1
boostdemand: 1
broadcastingcorp: 1
ormillion: 1
posture: 1
pastdue: 1
throughlloyds: 1
morningcut: 1
o: 1
r: 1
bcmalaysiancentralban: 1
givetime: 1
datelinebodycalifornia: 1
ltchina: 1
x: 1
butbrokers: 1
beturned: 1
reportof: 1
raisedits: 1
empty: 1
amounted: 1
toaplus: 1
divergence: 1
datelinebodywaste: 1
categorically: 1
cie: 1
graincommodity: 1
bcgartnergroupltgart: 1
sevenyears: 1
asprimary: 1
scanners: 1
datelinebodyreal: 1
isoglucose: 1
rolein: 1
visitgeorgetown: 1

cit: 1
offs: 1
bcrsicorpltrsicndq: 1
datetopicsdcpidtopicsplacesdluxembourgdplacespeoplepeopleorgsdecdorgsexchangesexchan
gescompaniescompaniesunknown: 1
silvercopper: 1
bctwasoldfourm: 1
olson: 1
itsintroduction: 1
productsltd: 1
telephonebased: 1
congressmen: 1
wascommitted: 1
incanada: 1
unknowntexttitlecray: 1
corpccc: 1
allegations: 1
cancer: 1
isdetermined: 1
telephoniques: 1
bycommercial: 1
bcusshoeinc: 1
denshindenwa: 1
asturias: 1
pill: 1
investigators: 1
oregonutah: 1
whenit: 1
reformer: 1
protecting: 1

Full code can be found in the given repository: https://git.cs.dal.ca/boza/csci-5408/-/tree/main/A3.

References:

[1] "Download intellij idea – the leading Java and Kotlin IDE," *JetBrains* [Online]. Available: https://www.jetbrains.com/idea/download/?section=mac [Accessed: 01 July 2023].

[2] Google, "Google Cloud Platform," Google, [Online]. Available: https://console.cloud.google.com/ [Accessed: 25 July 2023].

[3] "The most popular database for modern apps", MongoDB, 2020. [Online]. Available: https://www.mongodb.com/ [Accessed: 20 July 2023].

[4] D. Tucakov, "How to Install Spark on Ubuntu," phoenixNAP, [Online]. Available: https://phoenixnap.com/kb/install-spark-on-ubuntu. [Accessed: 23 July 2023].

[5] "Build, test, and debug regex," *regex101* [Online]. Available: https://regex101.com/ [Accessed: 20 July 2023].

[6] "Lab-5," *Brightspace Dalhousie University* [Online]. Available: https://dal.brightspace.com/d2l/le/content/271677/viewContent/3661458/View [Accessed: July 28, 2023].

[7] "Lab-7," *Brightspace Dalhousie University* [Online]. Available: https://dal.brightspace.com/d2l/le/content/271677/viewContent/3681262/View [Accessed: July 28, 2023].

[8] "Flowchart Maker & Online Diagram Software," *Draw.io* [Online]. Available: https://app.diagrams.net [Accessed: 29 July 2023].