EXNO : 1

Date

Install, configure and run Hadoop and HDFS

STEPS :

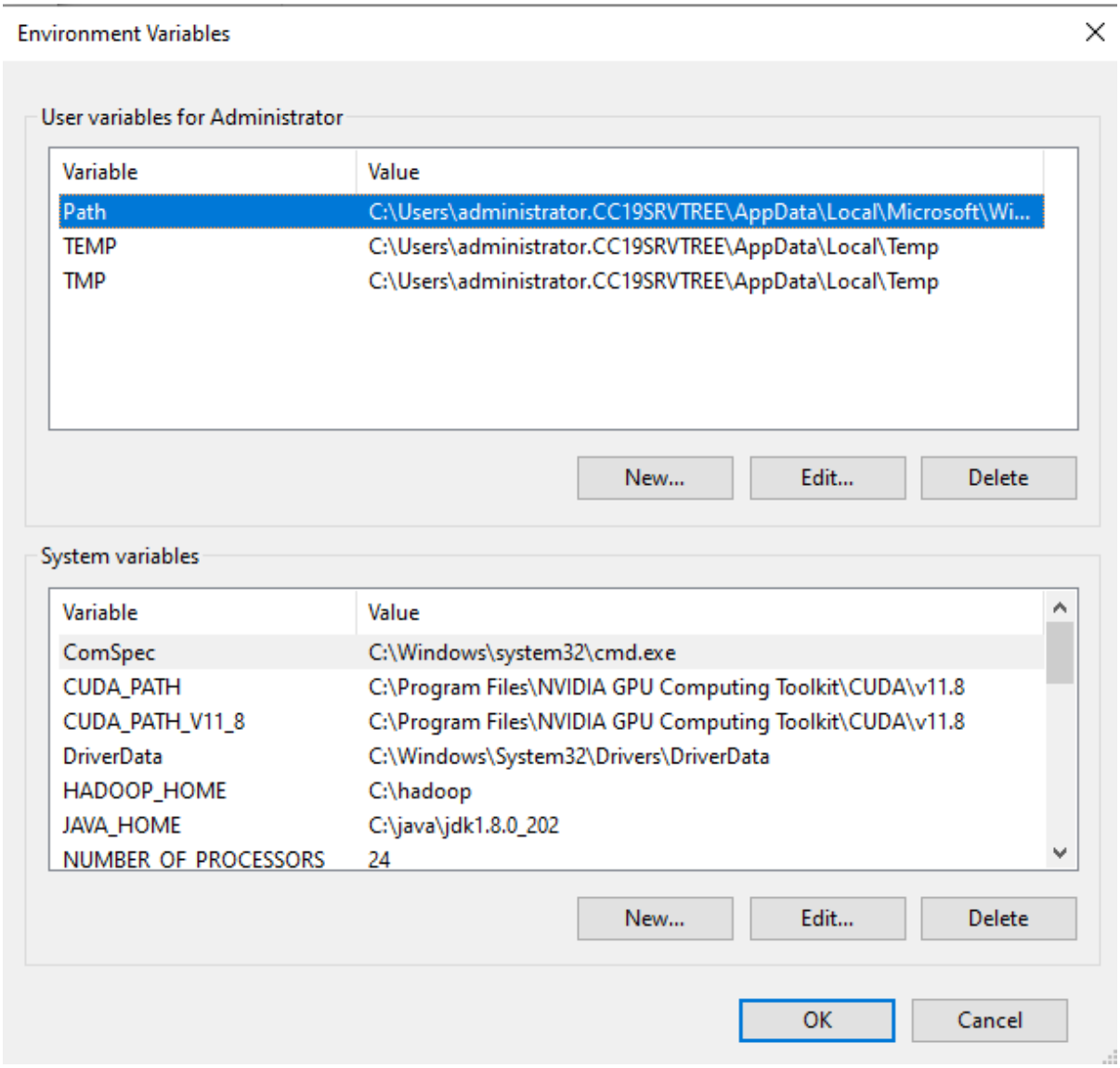
Step 1: Download Hadoop 3.2.4 get from

https://hadoop.apache.org/release/3.2.4.html

Step 2: Set the environment variable path for Hadoop

Go to Environment Variables and select it

User variable for administrator



Select new -> Create variable name is HADOOP_HOME = C:\hadoop\sbin

SYSTEMS VARIABLES

Select path and click edit -> new -> C:\hadoop\bin-> C:\hadoop\sbin

Step 3: Install java jdk 1.8.0 from

https://www.oracle.com/in/java/technologies/javase/javase8-archive-downloads.html

Set environment variable path for java like done for the Hadoop environmental variable setup

Go to Environment Variables and select it

User variable for administrator

Select new -> Create variable name is HOME_JAVA = C:\java\jdk1.8.0_202\bin

SYSTEMS VARIABLES

Select path and click edit -> new -> C:\java\jdk1.8.0_202\bin

Create a new folder named Java, and copy the JDK 1.8 folder from C:\Program Files\Java\jdk1.8.0_202 into the new Java folder

Step 4: Hadoop configuration

Set java path

Go to C:\hadoop\etc\hadoop -> hadoop-env -> @rem The java implementation to use. Required.

set JAVA_HOME=C:\java\jdk1.8.0_202

 # For core-site.xml

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

 # For hdfs-site.xml or https-site.xml

```
<property> C:\hadoop\sbin
  <name>dfs.replication</name>
  <value>1</value>
</property><property>
  <name>dfs.namenode.name.dir</name>
  <value>C:\hadoop\data\namenode</value>
</property><property>
  <name>dfs.datanode.data.dir</name>
  <value>C:\hadoop\data\datanode</value>
</property>
```

 # For mapred-site.xml

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

 # For yarn-site.xml

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce_shuffle</value>

</property><property>

<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>

Step:5 Creation of namenode and datanode

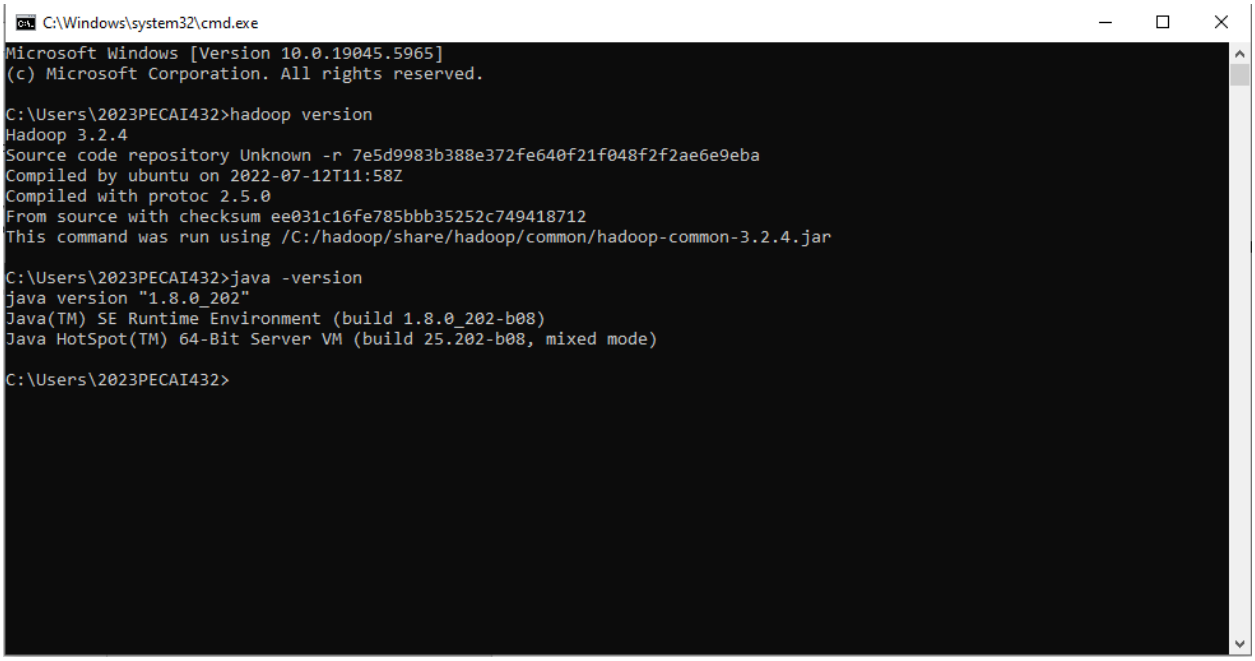
Create a new folder named data inside the Hadoop folder, and then create two subfolders inside the data folder named datanode and namenode

Step:6 Check whether Hadoop and Java are installed

Open Command Prompt and type it to check whether Hadoop and Java are installed

>>hadoop version

>>java -version



Step:7 Formatting Namenode and change directory from administrator to Hadoop sbin

Open Command Prompt and type

C:\Users\Administrator>hdfs namenode –format

Open Command Prompt as Administrator and navigate to the Hadoop sbin directory.

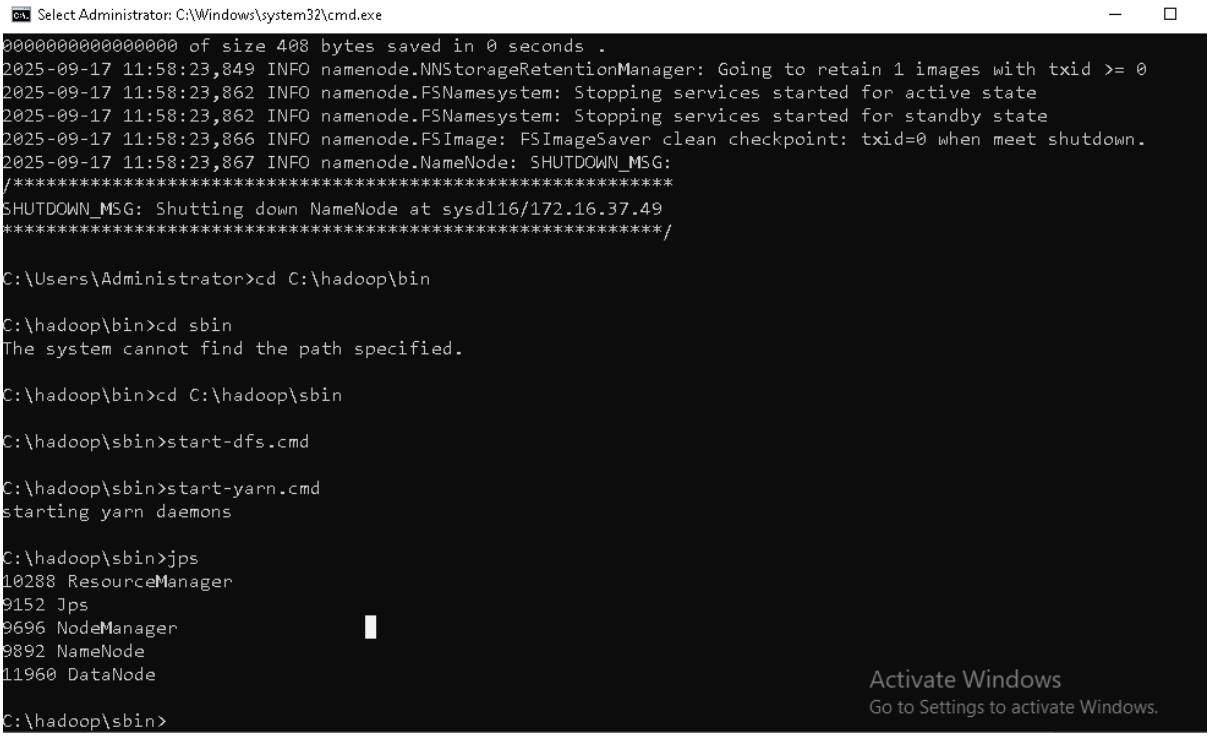
C:\Users\Administrator>cd C:\hadoop\bsin

Step:8 Now start all the Hadoop features

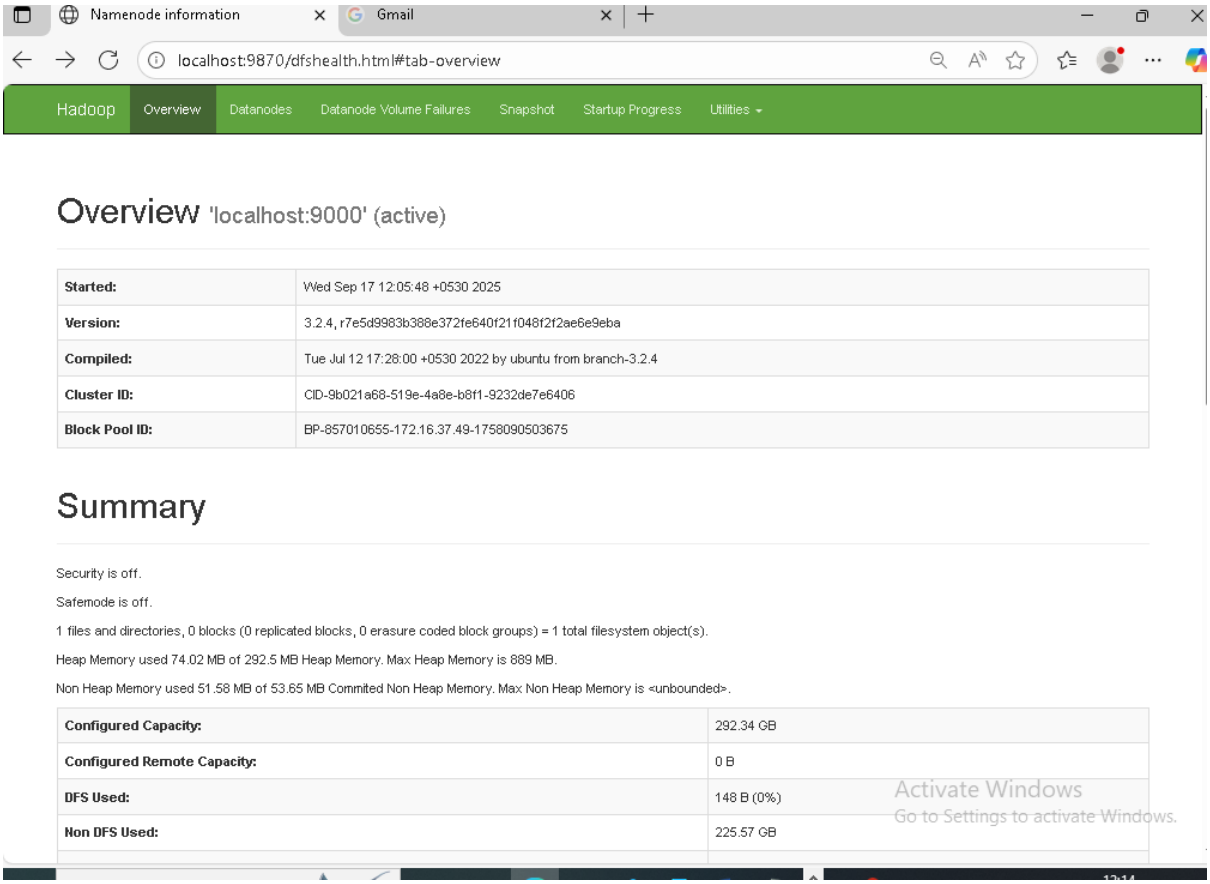
C:\hadoop\sbin>start-dfs.cmd

C:\hadoop\sbin>start-yarn.cmd

starting yarn daemons



Step 9: Now open Microsoft Edge and go to localhost http://localhost:9870 to check the Namenode interface in the Hadoop application



EXNO : 2

Date

Implement word count / frequency programs using MapReduce

Step 1: Formating the Namenode

Open Command Prompt as Administrator

C:\Users\Administrator>hdfs namenode –format

Step 2: Changing the directory from administrator to Hadoop sbin

Open Command Prompt as Administrator and navigate to the Hadoop sbin directory.

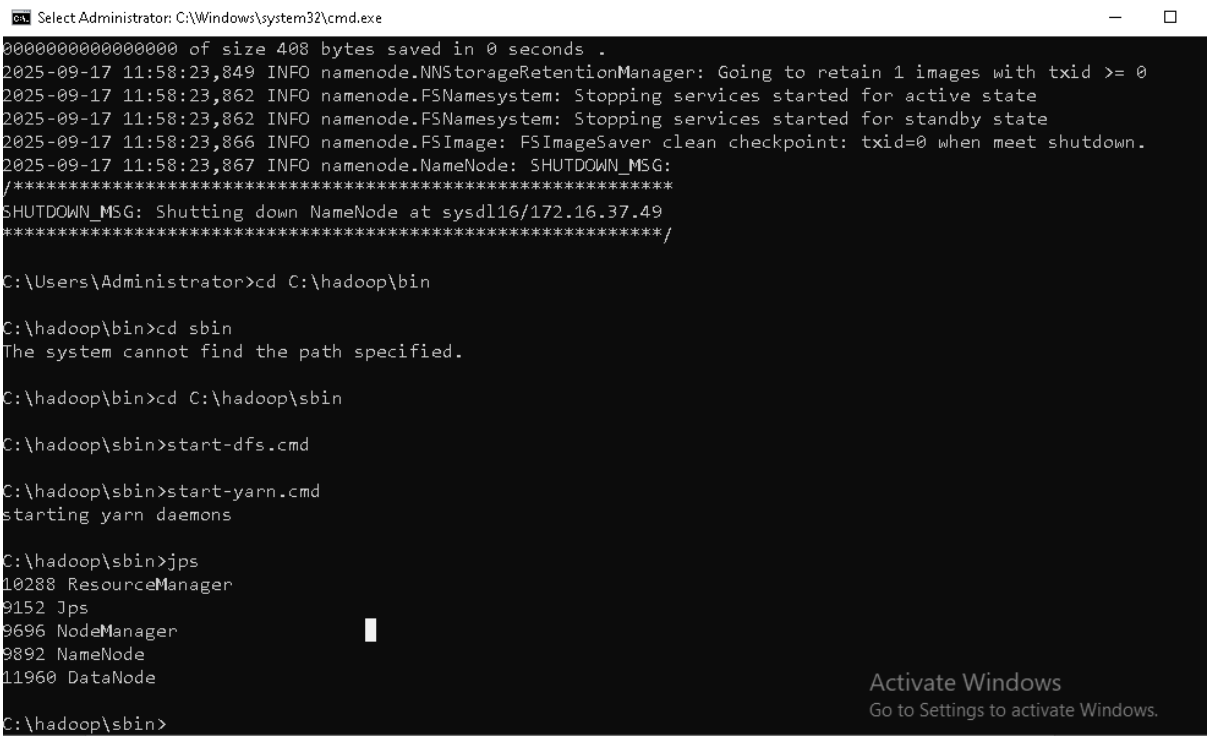
C:\Users\Administrator>cd C:\hadoop\bsin

Step 3: Start all Hadoop Features

C:\hadoop\sbin>start-dfs.cmd

C:\hadoop\sbin>start-yarn.cmd

starting yarn daemons



Step 4: Create a indata.txt in the input directory

Open command prompt and type the following commands to create the indata.txt file in the directory named input

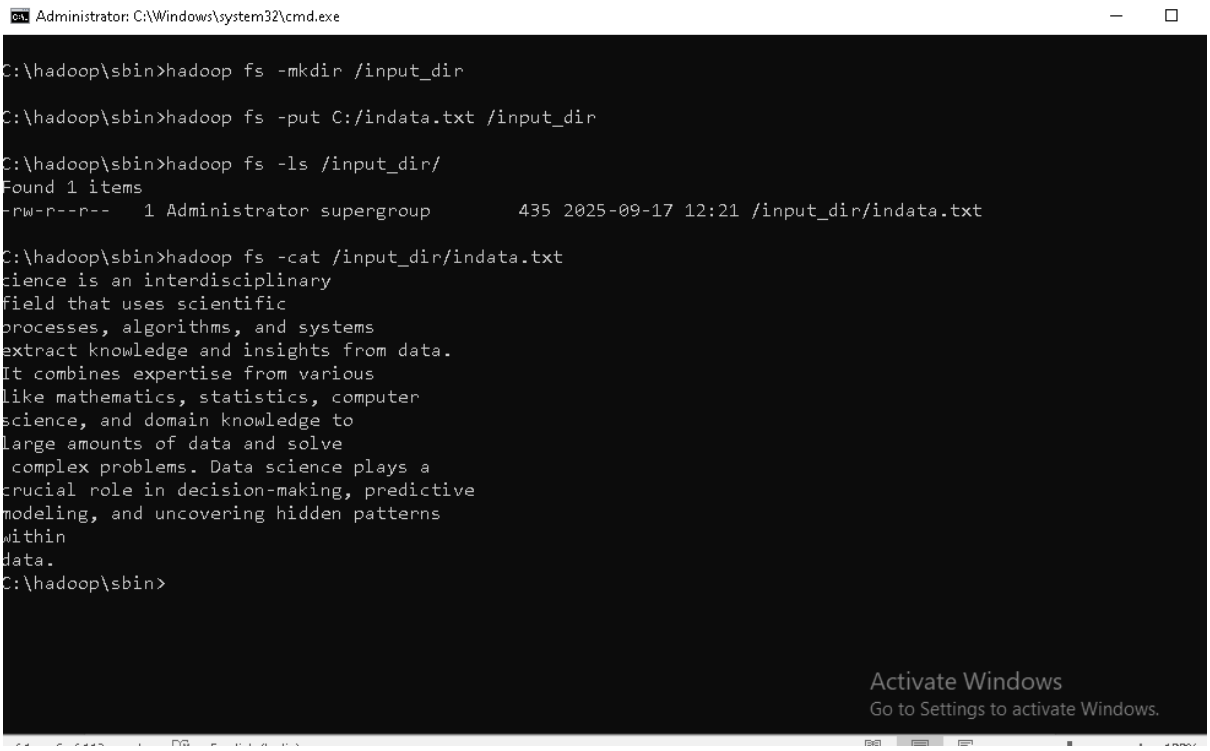
Use this directory for input file C:\hadoop\input_dir\indata.txt

>> hadoop fs -mkdir /input_dir

>> hadoop fs -put C:/indata.txt /input_dir

>> hadoop fs -ls /input_dir/

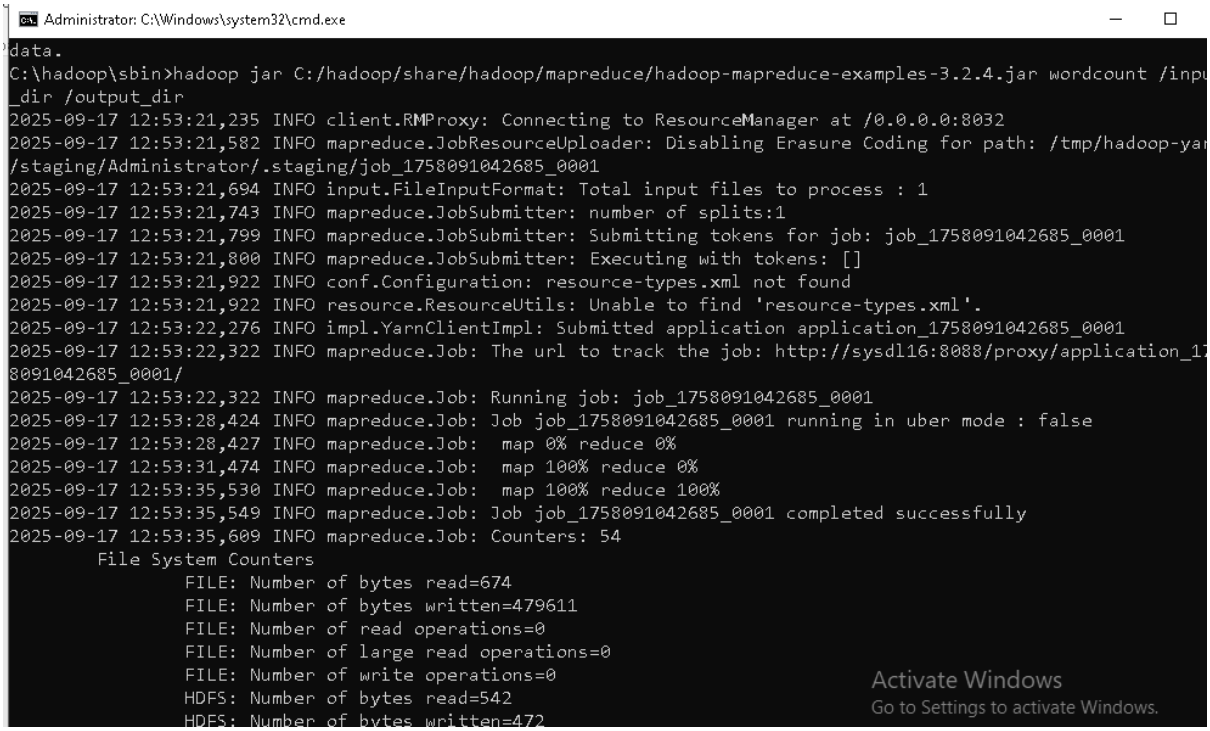
>> hadoop fs -cat /input_dir/indata.txt

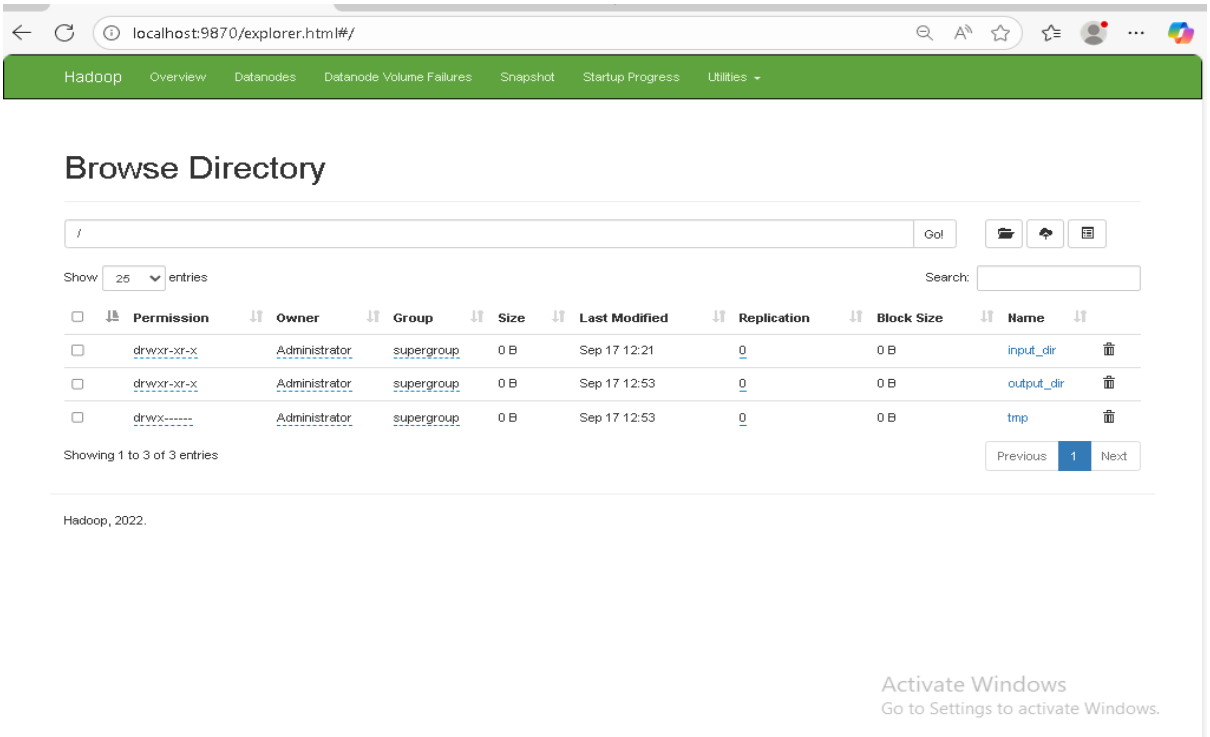


Step 5: Save the output in the output directory

Hadoop jar C:/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar wordcount /input_dir /output_dir

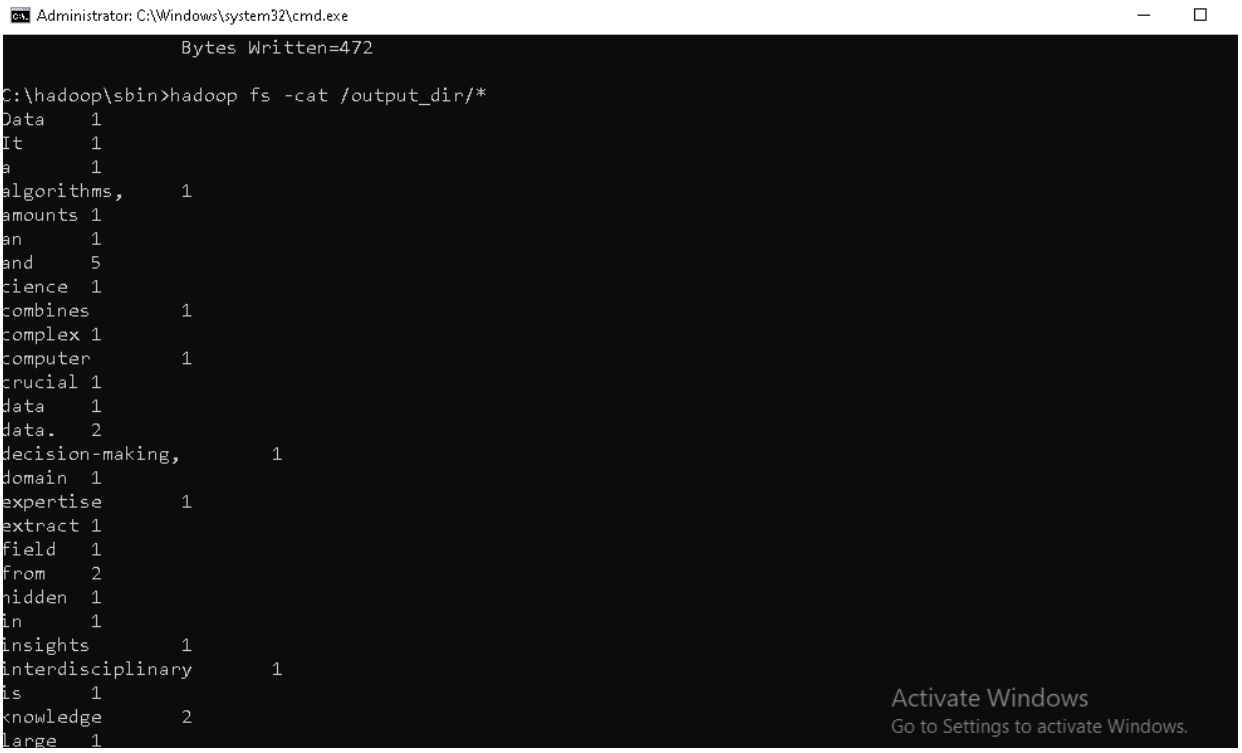
Use this directory to get the wordcount from the input file in the output directory


Administrator: C:\Windows\system32\cmd.exe
data.
C:\hadoop\sbin>hadoop jar C:/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.4.jar wordcount /inpu
_dir /output_dir
2025-09-17 12:53:21,235 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
2025-09-17 12:53:21,582 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yar
/staging/Administrator/.staging/job_1758091042685_0001
2025-09-17 12:53:21,694 INFO input.FileInputFormat: Total input files to process : 1
2025-09-17 12:53:21,743 INFO mapreduce.JobSubmitter: number of splits:1
2025-09-17 12:53:21,799 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1758091042685_0001
2025-09-17 12:53:21,800 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-09-17 12:53:21,922 INFO conf.Configuration: resource-types.xml not found
2025-09-17 12:53:21,922 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-09-17 12:53:22,276 INFO impl.YarnClientImpl: Submitted application application_1758091042685_0001
2025-09-17 12:53:22,322 INFO mapreduce.Job: The url to track the job: http://sysdl16:8088/proxy/application_17
8091042685_0001/
2025-09-17 12:53:22,322 INFO mapreduce.Job: Running job: job_1758091042685_0001
2025-09-17 12:53:28,424 INFO mapreduce.Job: Job job_1758091042685_0001 running in uber mode : false
2025-09-17 12:53:28,427 INFO mapreduce.Job: map 0% reduce 0%
2025-09-17 12:53:31,474 INFO mapreduce.Job: map 100% reduce 0%
2025-09-17 12:53:35,530 INFO mapreduce.Job: map 100% reduce 100%
2025-09-17 12:53:35,549 INFO mapreduce.Job: Job job_1758091042685_0001 completed successfully
2025-09-17 12:53:35,609 INFO mapreduce.Job: Counters: 54
File System Counters
FILE: Number of bytes read=674
FILE: Number of bytes written=479611
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=542
HDFS: Number of bytes written=472
Activate Windows
Go to Settings to activate Windows.


localhost:9870/explorer.html#/
Hadoop
Overview
Datanodes
Datanode Volume Failures
Snapshot
Startup Progress
Utilities
Browse Directory
/
Go!
Show 25 entries
Search:
Permission
Owner
Group
Size
Last Modified
Replication
Block Size
Name
drwxr-xr-x Administrator supergroup 0 B Sep 17 12:21 0 0 B input_dir
drwxr-xr-x Administrator supergroup 0 B Sep 17 12:53 0 0 B output_dir
drwx------ Administrator supergroup 0 B Sep 17 12:53 0 0 B tmp
Showing 1 to 3 of 3 entries
Previous
1
Next
Hadoop, 2022.
Activate Windows
Go to Settings to activate Windows.

OUTPUT:

hadoop fs -cat /output_dir/*



EXNO : 3

Date

Implement an MR program that processes a weather dataset

Step 1:Download netbeans 8.2 from

https://netbeans-ide.informer.com/download/

Step 2: Install the netbeans 8.2

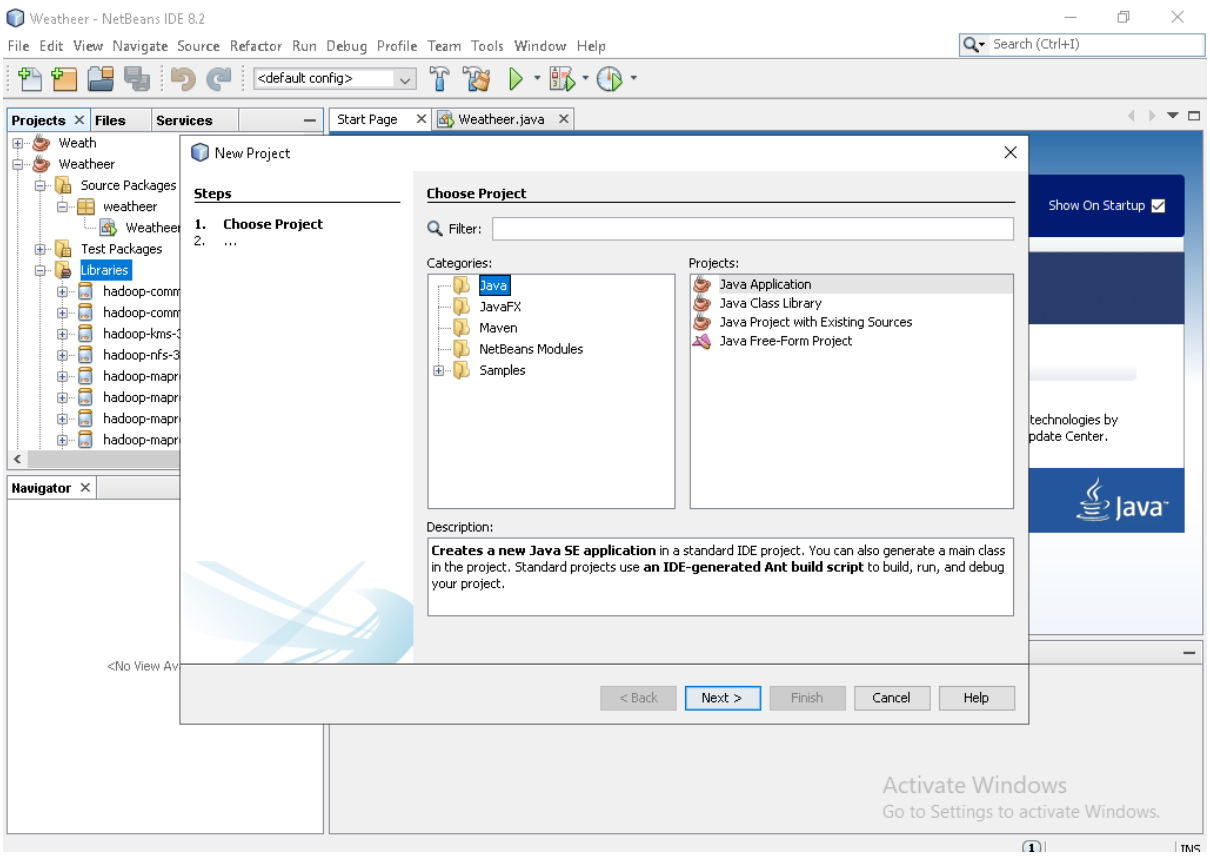
Install the downloaded netbeans 8.2 application in the system.

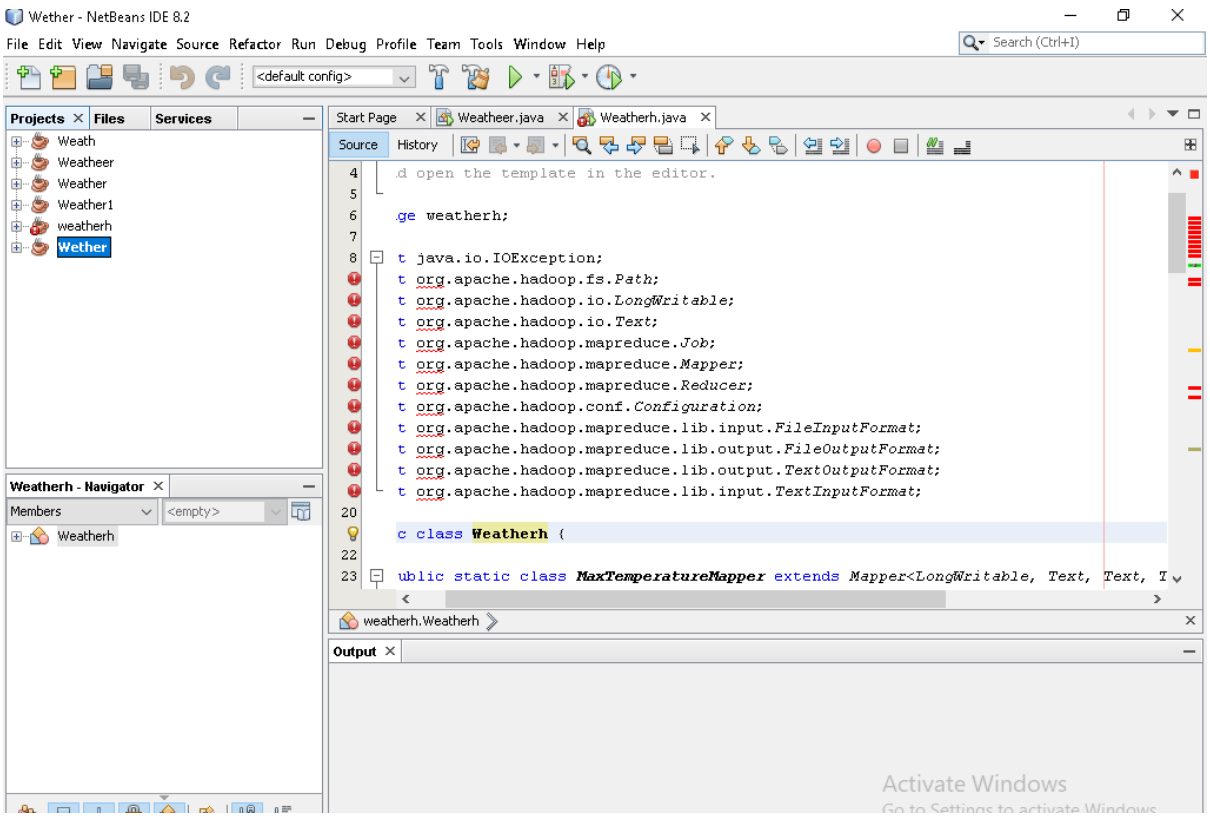
Step 3: Creating s new Project

Follow the steps below to create a new project in netbeans

File -> select new project -> categories -> java and project -> java application click on next

Enter your project name





#PROGRAM:

```
package weatheer;

import java.io.IOException;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;


public class Weatheer {
```

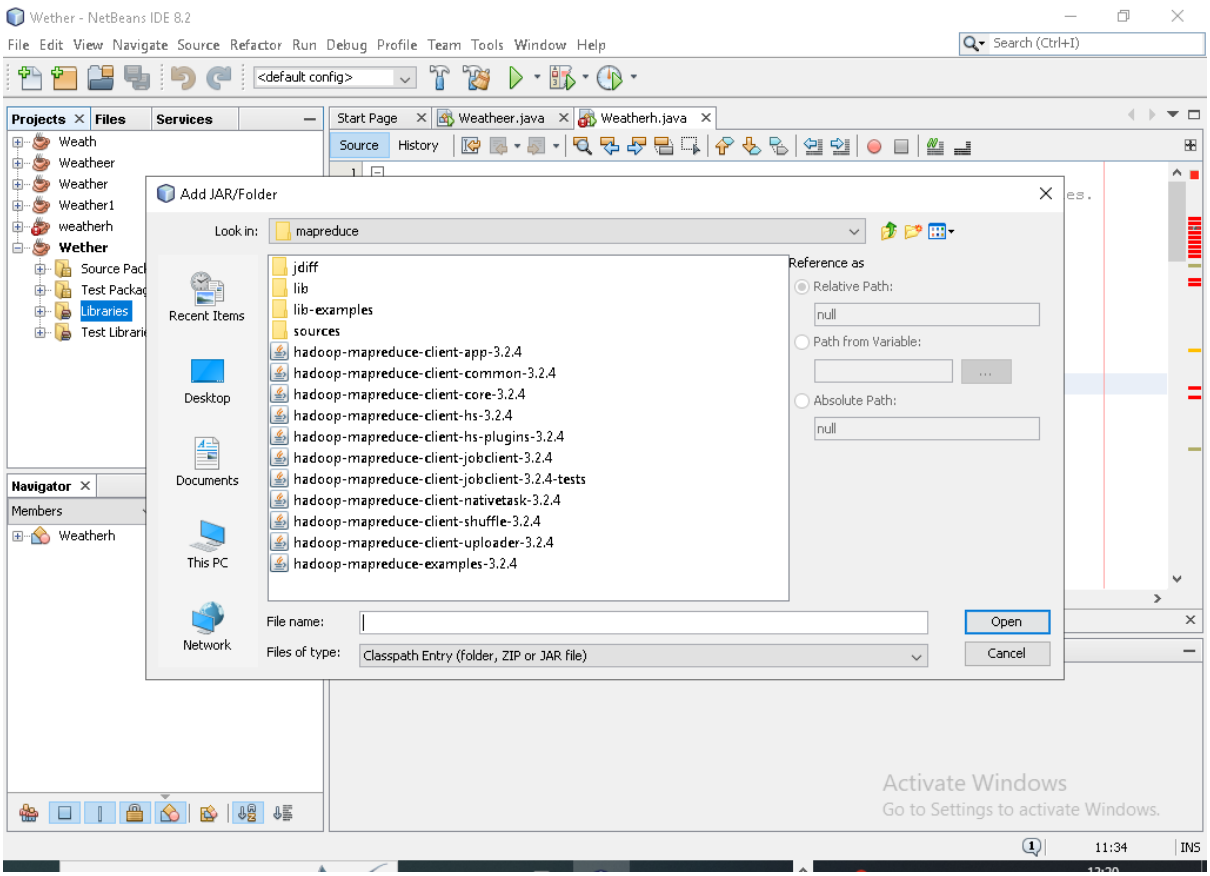
```
  public static class MaxTemperatureMapper extends Mapper<LongWritable, Text, Text, Text> {

    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {
      String line = value.toString();

      if (line.trim().isEmpty()) return;  // Skip empty lines

      try {
        String date = line.substring(6, 14);
        float tempMax = Float.parseFloat(line.substring(39, 45).trim());
        float tempMin = Float.parseFloat(line.substring(47, 53).trim());

        if (tempMax > 35.0) {
          context.write(new Text("Hot Day " + date), new Text(String.valueOf(tempMax)));
        }

        if (tempMin < 10.0) {
          context.write(new Text("Cold Day " + date), new Text(String.valueOf(tempMin)));
        }
      } catch (Exception e) {
        // Handle lines with invalid format or parsing errors
        // You can log this if needed
      }
    }
  }

  public static class MaxTemperatureReducer extends Reducer<Text, Text, Text, Text> {
```

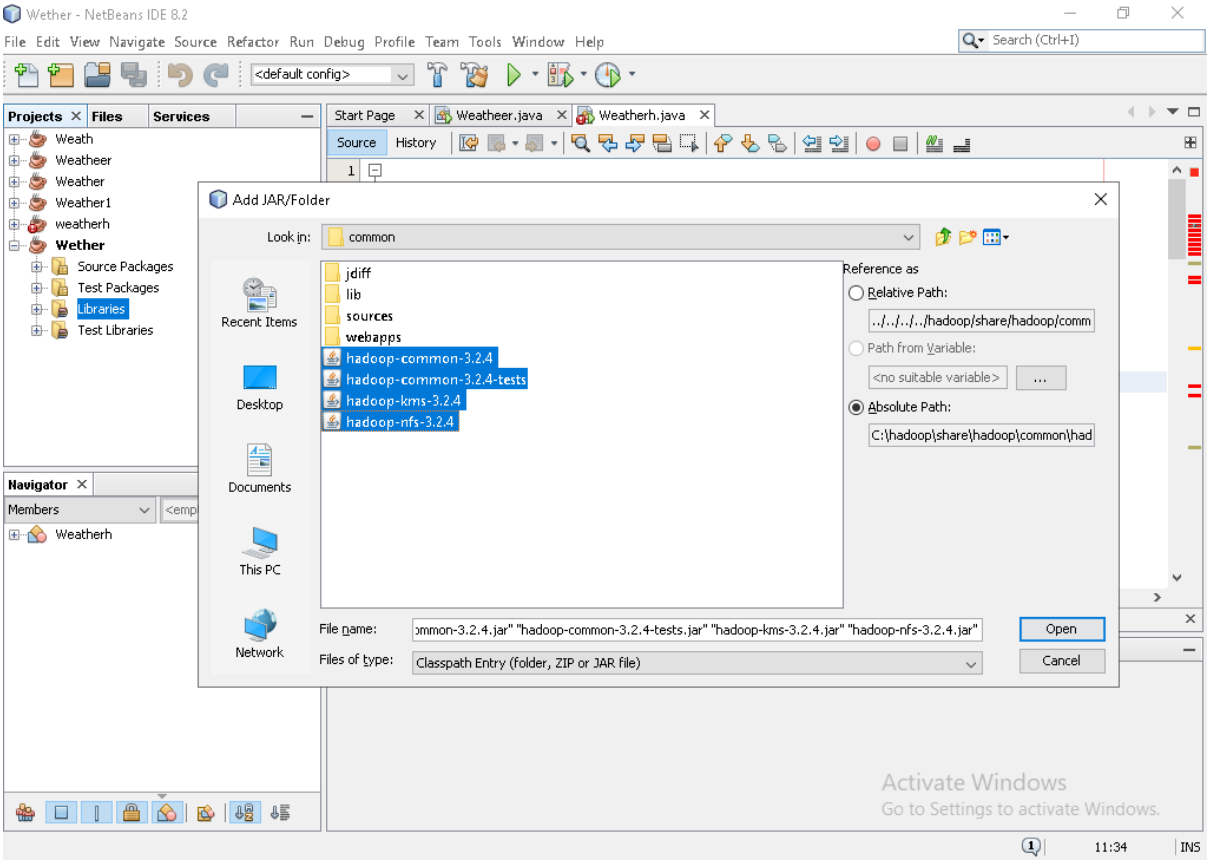
```
    @Override
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
      // Simply write the first temperature found
      context.write(key, values.iterator().next());
    }
  }

  public static void main(String[] args) throws Exception {
    if (args.length < 2) {
      System.err.println("Usage: Weatheer <input path> <output path>");
      System.exit(-1);
    }

    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "Weather Example");

    job.setJarByClass(Weatheer.class);
    job.setMapperClass(MaxTemperatureMapper.class);
    job.setReducerClass(MaxTemperatureReducer.class);

    job.setMapOutputKeyClass(Text.class);
    job.setMapOutputValueClass(Text.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);

    job.setInputFormatClass(TextInputFormat.class);
    job.setOutputFormatClass(TextOutputFormat.class);

    FileInputFormat.addInputPath(job, new Path(args[0]));
    Path outputPath = new Path(args[1]);
```

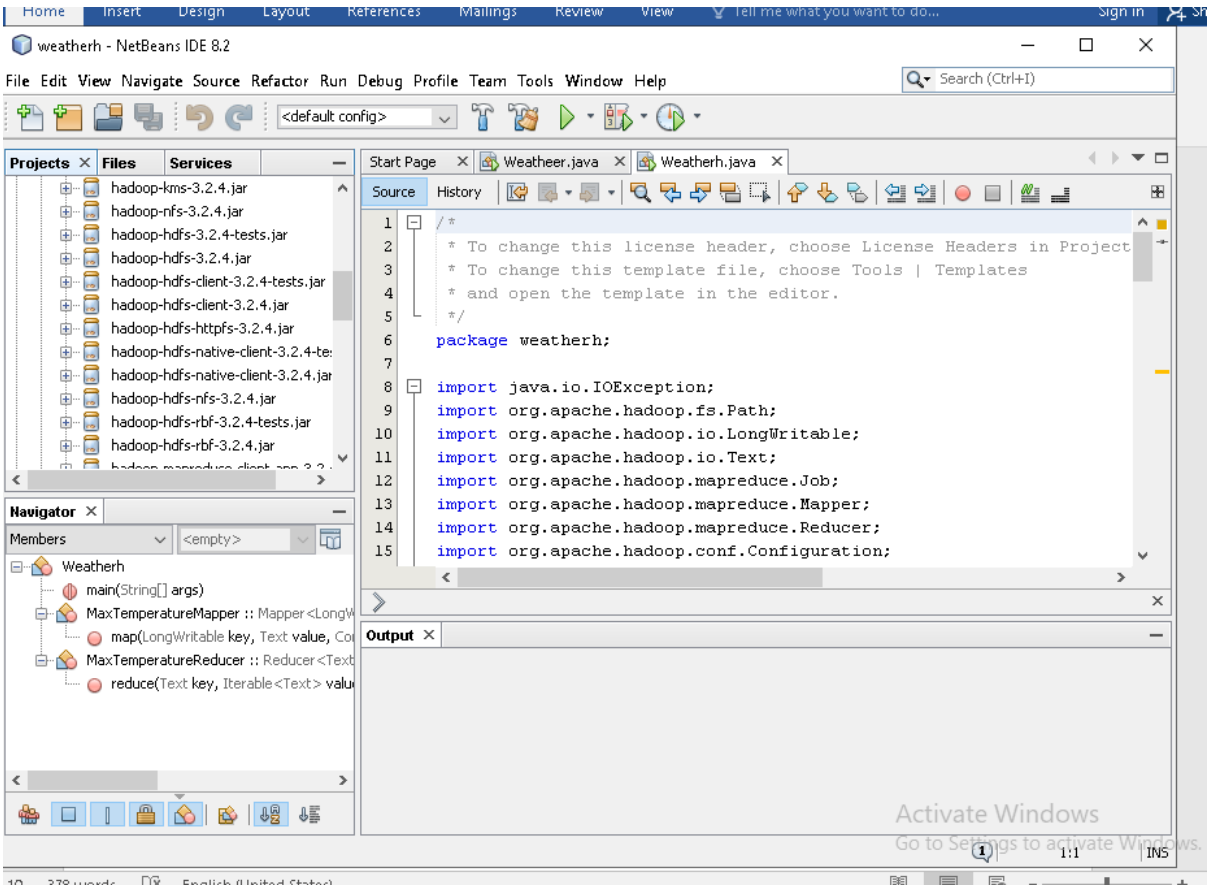
```
        // Delete output path if exists
        outputPath.getFileSystem(conf).delete(outputPath, true);
        FileOutputFormat.setOutputPath(job, outputPath);

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Step 4: Adding the JAR files from Hadoop directory

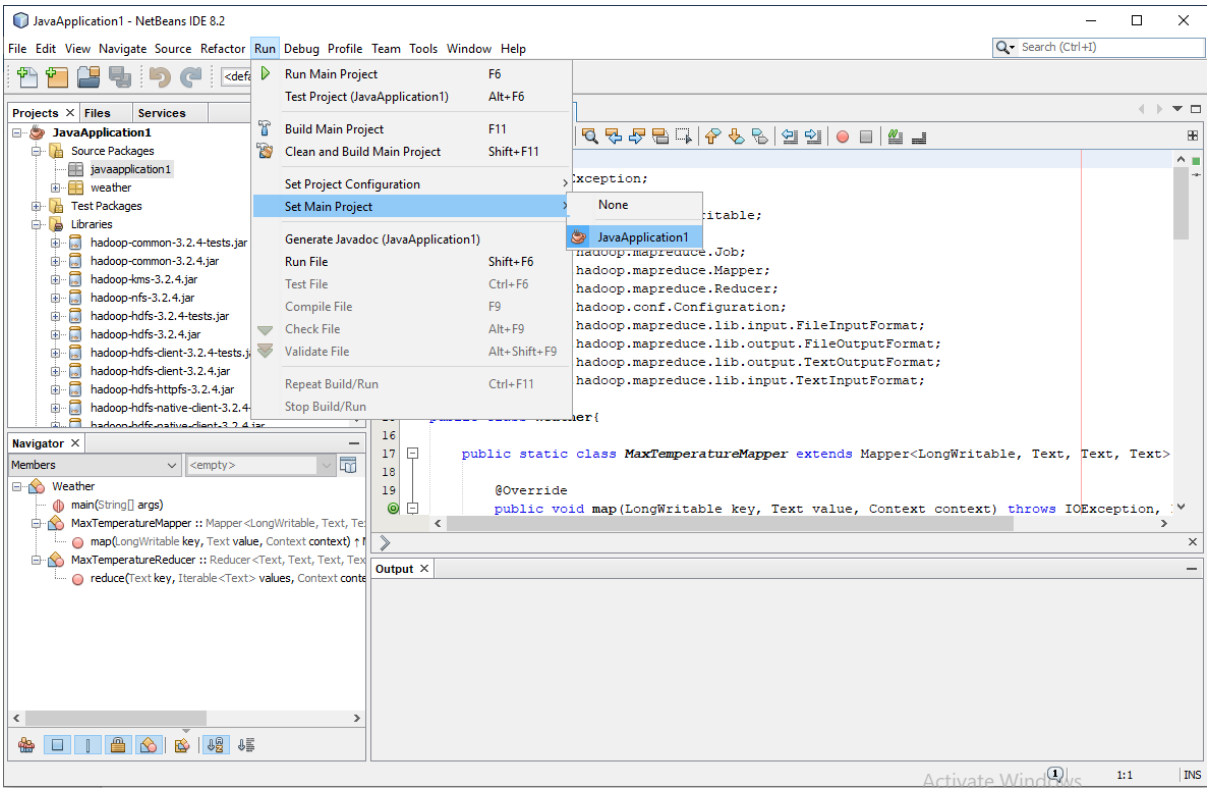
Go to project panel and select your project name and right click on select ADD JAR/ FOLDER and go Hadoop folder in the directory C:\hadoop\share\hadoop and the import all the JAR files from the folders named common, hdfs, mapreduce and yarn to run the project




Wether - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects Files Services
Weath
Weatheer
Weather
Weather1
weatherh
Wether
Source Packages
Test Packages
Libraries
Test Libraries
Start Page
Weatheer.java
Weatherh.java
Source History
Add JAR/Folder
Look in: common
Recent Items
Desktop
Documents
This PC
Network
jdiff
lib
sources
webapps
hadoop-common-3.2.4
hadoop-common-3.2.4-tests
hadoop-kms-3.2.4
hadoop-nfs-3.2.4
Reference as
Relative Path:
../../../../hadoop/share/hadoop/comm
Path from Variable:
<no suitable variable>
Absolute Path:
C:\hadoop\share\hadoop\common\had
File name: ommon-3.2.4.jar" "hadoop-common-3.2.4-tests.jar" "hadoop-kms-3.2.4.jar" "hadoop-nfs-3.2.4.jar"
Files of type: Classpath Entry (folder, ZIP or JAR file)
Open
Cancel
Navigator
Members
Weatherh
Activate Windows
Go to Settings to activate Windows.
11:34
INS


weatherh - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects Files Services
hadoop-kms-3.2.4.jar
hadoop-nfs-3.2.4.jar
hadoop-hdfs-3.2.4-tests.jar
hadoop-hdfs-3.2.4.jar
hadoop-hdfs-client-3.2.4-tests.jar
hadoop-hdfs-client-3.2.4.jar
hadoop-hdfs-httpfs-3.2.4.jar
hadoop-hdfs-native-client-3.2.4-te
hadoop-hdfs-native-client-3.2.4.jar
hadoop-hdfs-nfs-3.2.4.jar
hadoop-hdfs-rbf-3.2.4-tests.jar
hadoop-hdfs-rbf-3.2.4.jar
Start Page
Weatheer.java
Weatherh.java
Source History
1 /*
2 * To change this license header, choose License Headers in Project
3 * To change this template file, choose Tools | Templates
4 * and open the template in the editor.
5 */
6 package weatherh;
7
8 import java.io.IOException;
9 import org.apache.hadoop.fs.Path;
10 import org.apache.hadoop.io.LongWritable;
11 import org.apache.hadoop.io.Text;
12 import org.apache.hadoop.mapreduce.Job;
13 import org.apache.hadoop.mapreduce.Mapper;
14 import org.apache.hadoop.mapreduce.Reducer;
15 import org.apache.hadoop.conf.Configuration;
Navigator
Members
<empty>
Weatherh
main(String[] args)
MaxTemperatureMapper :: Mapper<LongW
map(LongWritable key, Text value, Co
MaxTemperatureReducer :: Reducer<Text
reduce(Text key, Iterable<Text> valu
Output
Activate Windows
Go to Settings to activate Windows.
1:1
INS

Step 5:Building the main project



Now set our project as the Main Project in the netbeans and then Clean and built the main project now a JAR file of the main project will be created in the directory

C:\Users\2023PECAI432\Documents\NetBeansProjects\JavaApplication1\dist

Copy the file to C directory which is the file to run in cmd to get the output.

Step 6: Formating the Namenode

Open Command Prompt as Administrator

C:\Users\Administrator>hdfs namenode –format

Step 7: Start all Hadoop Features

start-all.cmd

start-dfs.cmd

start-yarn.cmd

Step 8: Input and output file

Download weather dataset from any source

Open command prompt and type the following commands to open weather.txt file in the directory named input_dir

Use this directory for input file C:\hadoop\input_dir\weather.txt

hadoop fs -mkdir /input_dir

hadoop fs -put C:/weatherr.txt /input_dir

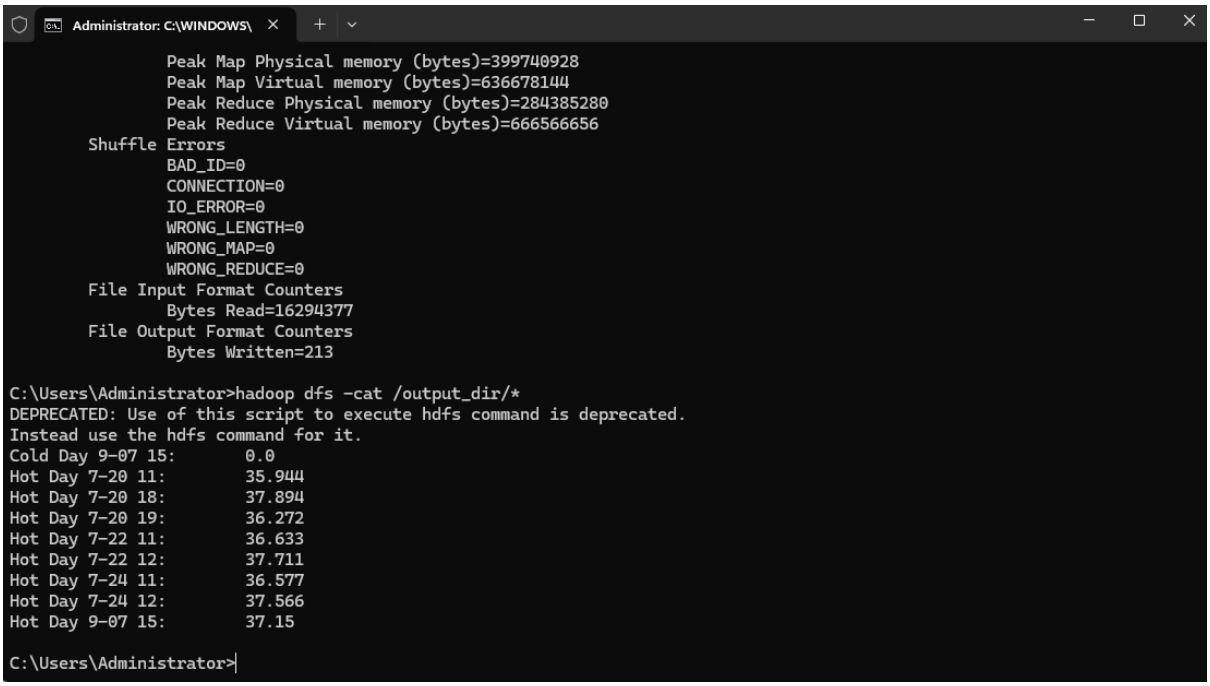
Hadoop fs –cat /input_dir/weatherr.txt

Use the following commands to show the output in the output_dir directory

hadoop jar c:/Weath.jar /input_dir /output_dir

hadoop jar C:/Weatheer.jar /input_dir /output_dir

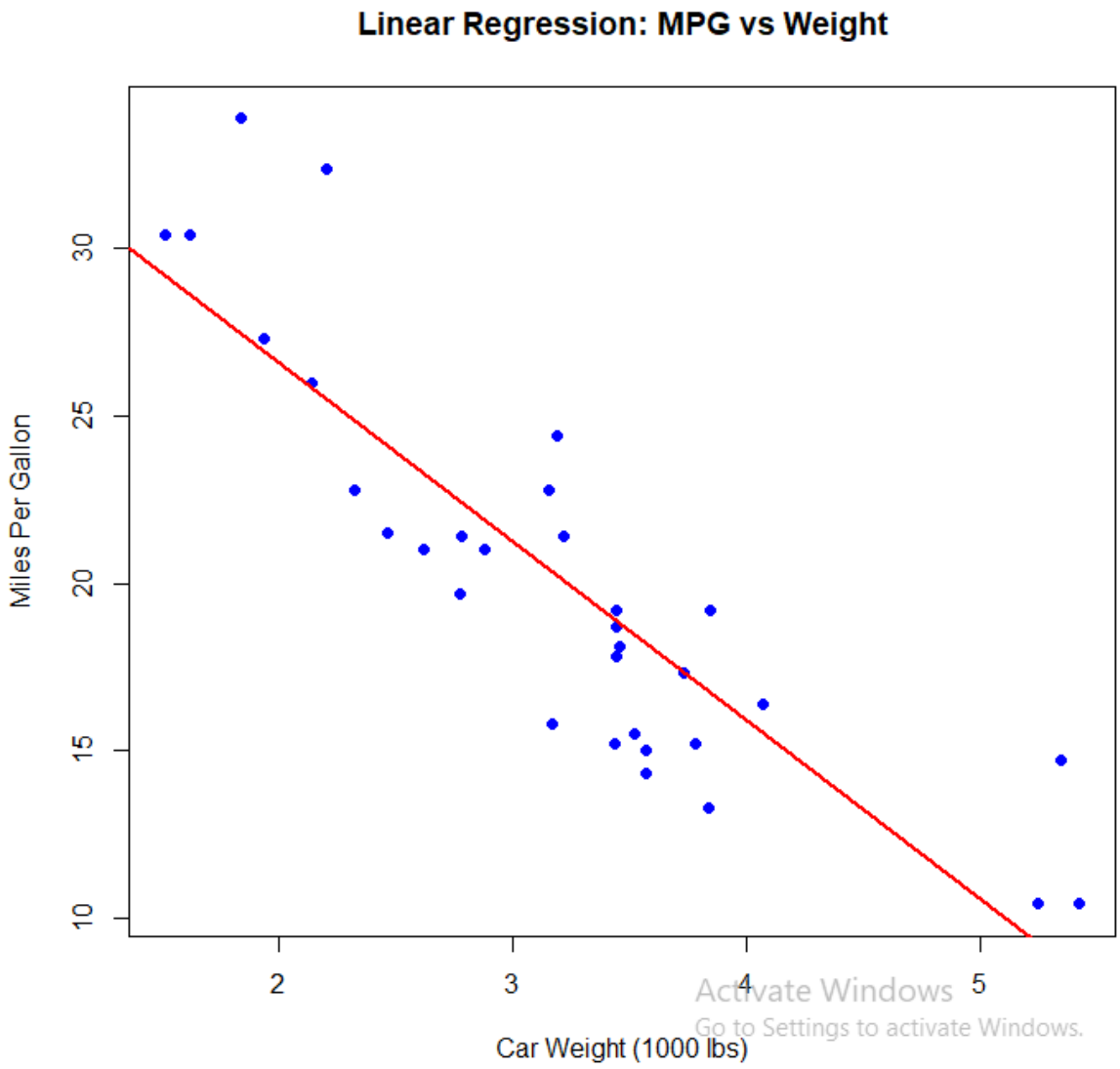
hadoop dfs -cat /output_dir/*



Program :

```
data(mtcars)

# Fit linear regression model
linear_model <- lm(mpg ~ wt, data = mtcars)

# Summary of the model
summary(linear_model)

# Predict using model
predicted_mpg <- predict(linear_model)

# Plot
plot(mtcars$wt, mtcars$mpg,
    main = "Linear Regression: MPG vs Weight",
    xlab = "Car Weight (1000 lbs)",
    ylab = "Miles Per Gallon",
    pch = 19, col = "blue")
abline(linear_model, col = "red", lwd = 2)

# Calculate R-squared
r_squared <- summary(linear_model)$r.squared
cat("R-squared:", round(r_squared, 3), "\n")
```

Output :



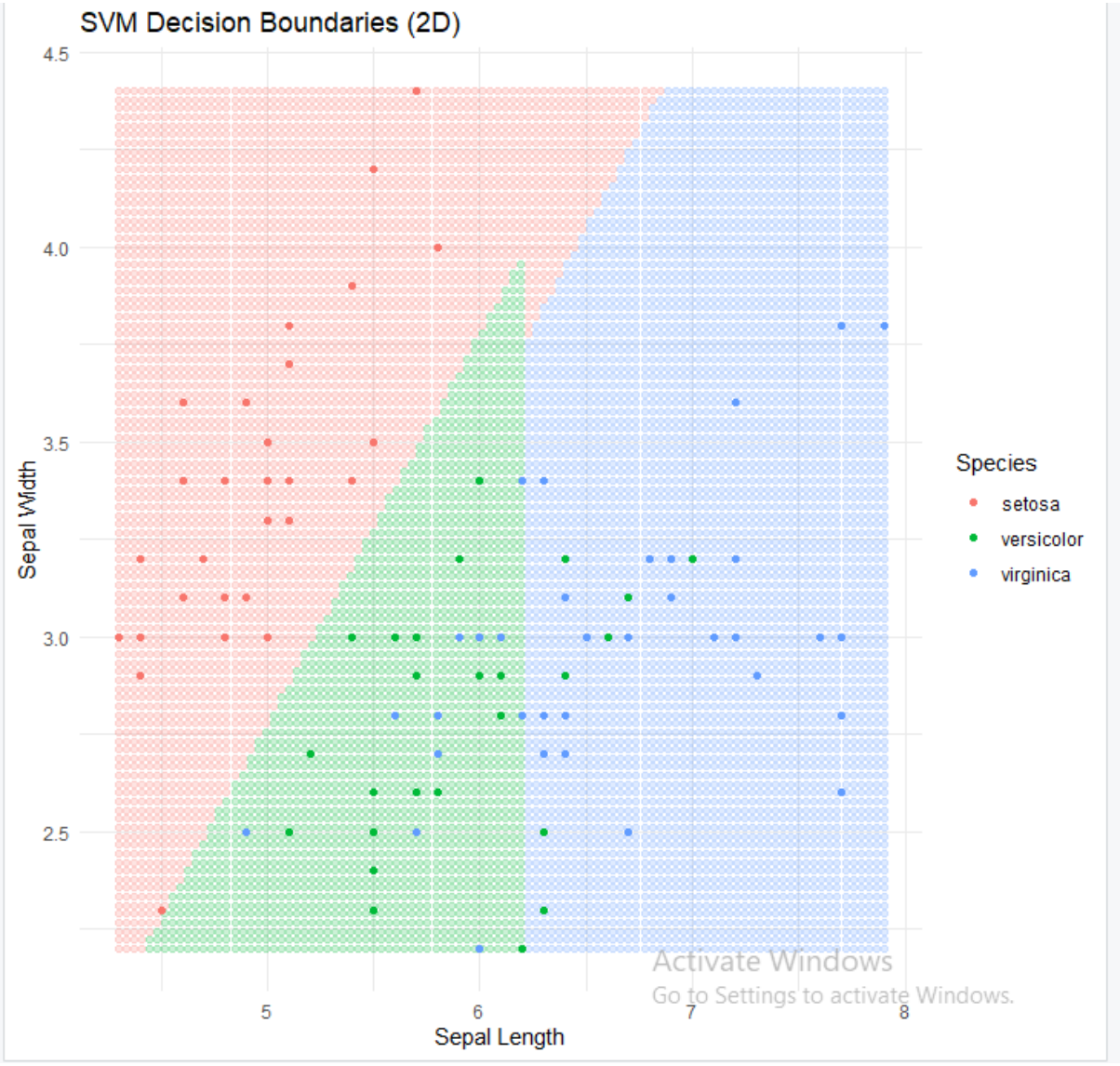
Result :

Program :

```
# Install & load required packages
if (!require("e1071")) install.packages("e1071")
if (!require("ggplot2")) install.packages("ggplot2")

library(e1071)
library(ggplot2)

# Load dataset
data(iris)

# Split data into training (70%) and testing (30%)
set.seed(123)
sample_index <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_index, ]
test_data <- iris[-sample_index, ]

# Train SVM model (linear kernel)
svm_model <- svm(Species ~ ., data = train_data, kernel = "linear")

# Predict on test data
svm_pred <- predict(svm_model, test_data)

# Confusion Matrix
cat("SVM Confusion Matrix:\n")
print(table(Predicted = svm_pred, Actual = test_data$Species))

# 2D Decision Boundary using Sepal.Length & Sepal.Width
svm_model_2d <- svm(Species ~ Sepal.Length + Sepal.Width, data = train_data, kernel = "linear")

xrange <- seq(min(train_data$Sepal.Length), max(train_data$Sepal.Length), length.out = 100)
```

```
yrange <- seq(min(train_data$Sepal.Width), max(train_data$Sepal.Width), length.out = 100)
grid <- expand.grid(Sepal.Length = xrange, Sepal.Width = yrange)
grid$Species <- predict(svm_model_2d, newdata = grid)

# Plot SVM decision boundaries
ggplot() +
  geom_point(data = grid, aes(Sepal.Length, Sepal.Width, color = Species), alpha = 0.2) +
  geom_point(data = train_data, aes(Sepal.Length, Sepal.Width, color = Species), shape = 19) +
  labs(title = "SVM Decision Boundaries (2D)",
       x = "Sepal Length", y = "Sepal Width") +
  theme_minimal()
```

Output :



Result :

Program:

```
# Install & load required packages
if (!require("ggplot2")) install.packages("ggplot2")
if (!require("factoextra")) install.packages("factoextra")
if (!require("cluster")) install.packages("cluster")

library(ggplot2)
library(factoextra)
library(cluster)

# Load dataset
data(iris)

# Use only numeric columns (remove Species)
iris_data <- iris[, -5]

# Run K-Means with 3 clusters
set.seed(123)
kmeans_result <- kmeans(iris_data, centers = 3, nstart = 25)

# Display cluster centers
cat("K-Means Cluster Centers:\n")
print(kmeans_result$centers)

# Compare clusters with actual species
cat("\nK-Means Cluster Assignments:\n")
print(table(Actual = iris$Species, Cluster = kmeans_result$cluster))

# Visualize K-Means clustering
fviz_cluster(kmeans_result, data = iris_data,
        palette = c("#2E9FDF", "#00AFBB", "#E7B800"),
```

```
geom = "point", ellipse.type = "convex",
ggtheme = theme_minimal(),
main = "K-Means Clustering - Iris Dataset")
```

Output :

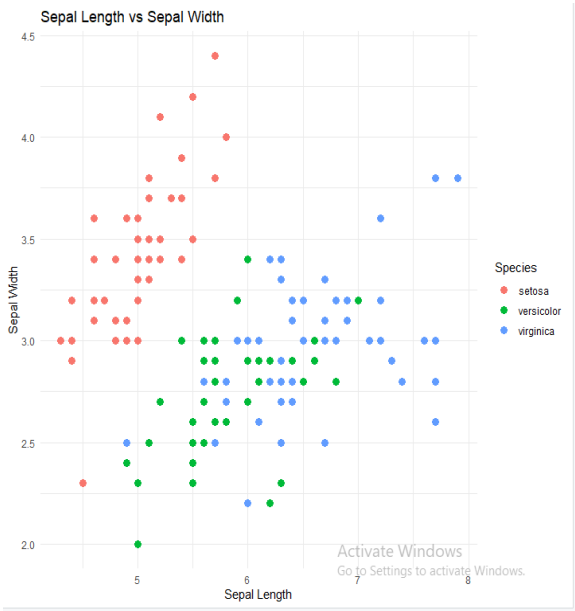
Result :

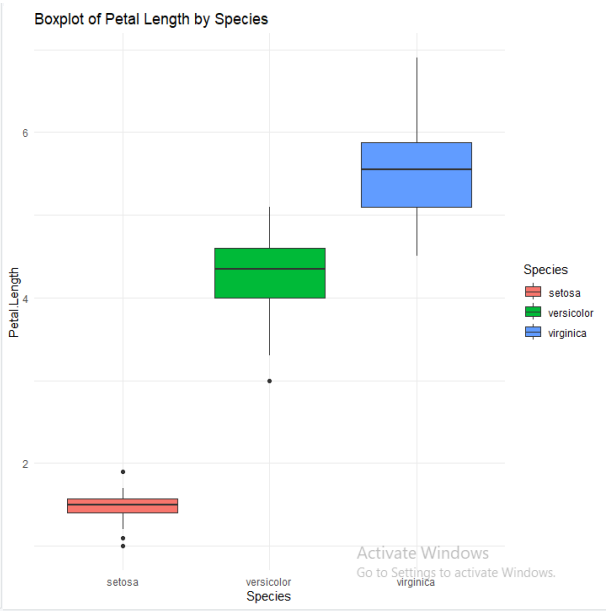
Program :

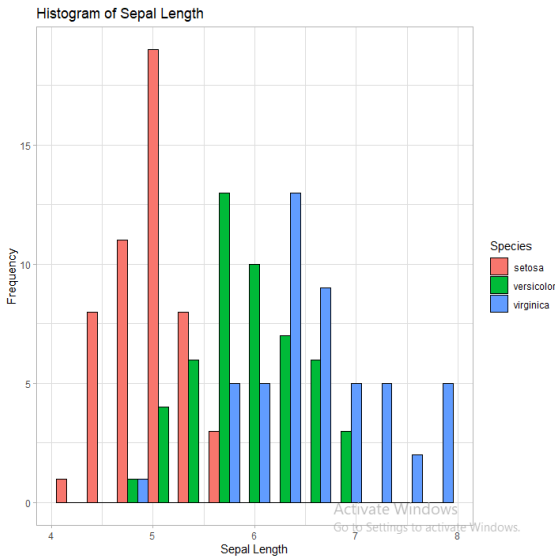
```
if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)
data(iris)
ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +
  geom_point(size = 3) +
  labs(title = "Sepal Length vs Sepal Width",
       x = "Sepal Length",
       y = "Sepal Width") +
  theme_minimal()

ggplot(iris, aes(x = Species, y = Petal.Length, fill = Species)) +
  geom_boxplot() +
  labs(title = "Boxplot of Petal Length by Species") +
  theme_minimal()

ggplot(iris, aes(x = Sepal.Length, fill = Species)) +
  geom_histogram(binwidth = 0.3, position = "dodge", color = "black") +
  labs(title = "Histogram of Sepal Length",
       x = "Sepal Length",
       y = "Frequency") +
  theme_light()

if (!require("GGally")) install.packages("GGally")
library(GGally)

ggpairs(iris, aes(color = Species),
        title = "Scatterplot Matrix of Iris Features")
ggplot(iris, aes(x = Species, y = Petal.Width, fill = Species)) +
```

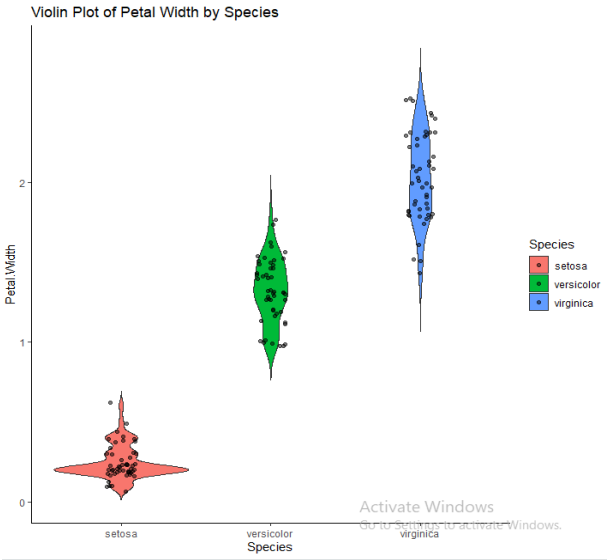
```
geom_violin(trim = FALSE) +
geom_jitter(width = 0.1, alpha = 0.5) +
labs(title = "Violin Plot of Petal Width by Species") +
theme_classic()
```

Output :







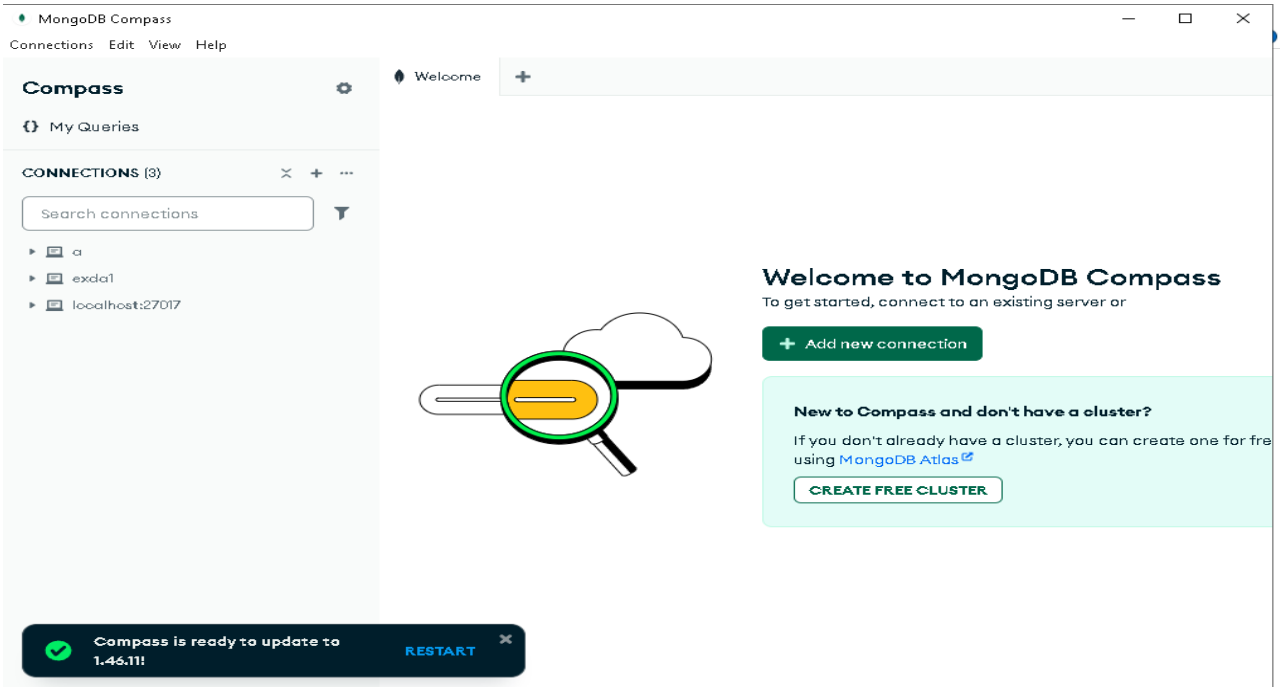


Result :

Program :

Step: 1https://www.mongodb.com/try/download/community

Step2: create mogodb connection , click on Add New connection and sav &e connection



Step 1: Install and load `mongolite` package

install.packages("mongolite") # Run only once

library(mongolite)

Step 2: Connect to MongoDB

Replace with your MongoDB URI and database/collection names

mongo_conn <- mongo(collection = "students", db = "school", url = "mongodb://localhost")

 # Insert a single document

mongo_conn$insert('{"name": "John", "age": 21, "grade": "A"}')

 # Insert multiple documents

data <- data.frame(

name = c("Alice", "Bob"),

```
  age = c(22, 23),
  grade = c("B", "A")
)
mongo_conn$insert(data)
```

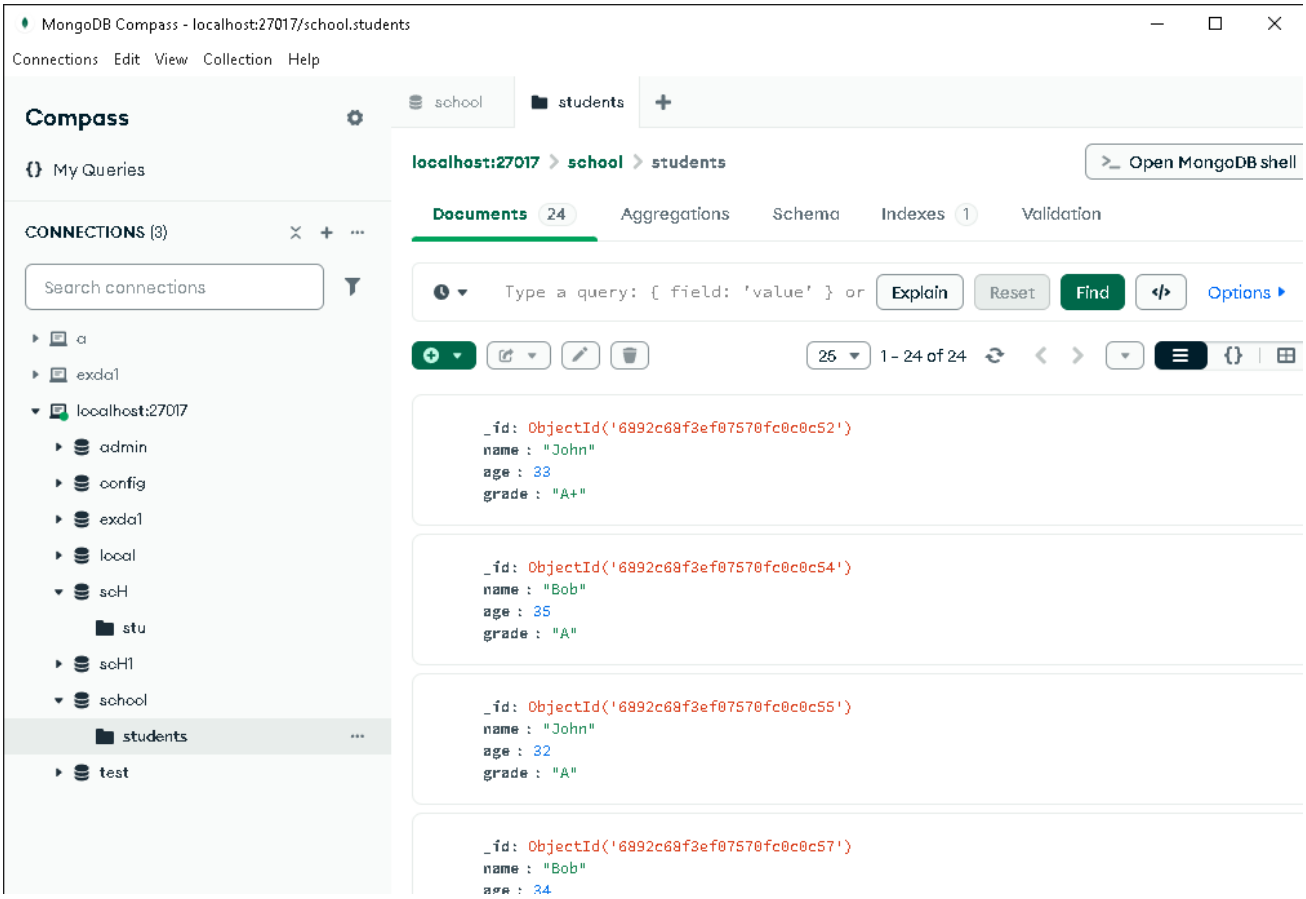
Find documents

```
# Find all documents . Retrieve Data
all_docs <- mongo_conn$find()
print(all_docs)

# Find with filter
result <- mongo_conn$find('{"grade": "A"}')
print(result)
```

Update documents

```
# Update one document: change grade to "A+" where name is "John"
mongo_conn$update('{"name": "John"}', '{"$set": {"grade": "A+"}}')

# Update multiple documents: increase age by 1 for all
mongo_conn$update('{"age": {"$exists": true}}', '{"$inc": {"age": 1}}', multiple = TRUE)
# Delete one document where name is "Alice"
mongo_conn$remove('{"name": "Alice"}')

# Delete all documents with age > 23
mongo_conn$remove('{"age": {"$gt": 23}}')
mongo_conn$drop()
mongodb://localhost:27017
```

Output :

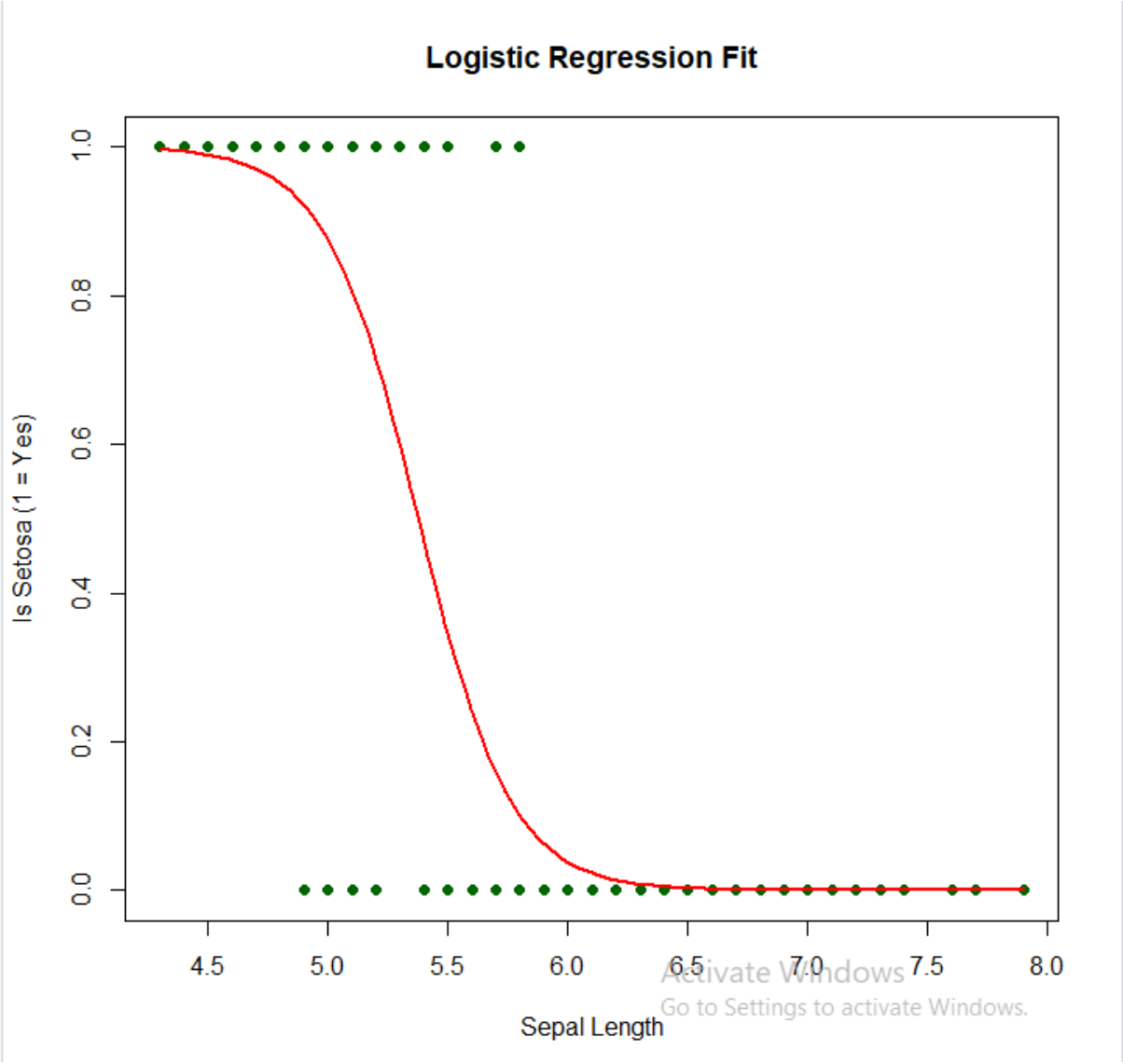


Result :

Program:

```
data(iris)

# Create binary target: 1 if Setosa, 0 otherwise
iris$IsSetosa <- ifelse(iris$Species == "setosa", 1, 0)

# Fit logistic regression model
logistic_model <- glm(IsSetosa ~ Sepal.Length, data = iris, family = "binomial")

# Summary
summary(logistic_model)

# Predict probabilities
predicted_probs <- predict(logistic_model, type = "response")

# Convert to class labels (0 or 1) with 0.5 threshold
predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)

# Accuracy
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
cat("Logistic Regression Accuracy:", round(accuracy * 100, 2), "%\n")

# Plot
plot(iris$Sepal.Length, iris$IsSetosa,
    pch = 19, col = "darkgreen",
    xlab = "Sepal Length", ylab = "Is Setosa (1 = Yes)",
    main = "Logistic Regression Fit")
curve(predict(logistic_model, data.frame(Sepal.Length = x), type = "response"),
     add = TRUE, col = "red", lwd = 2)
```

Output :

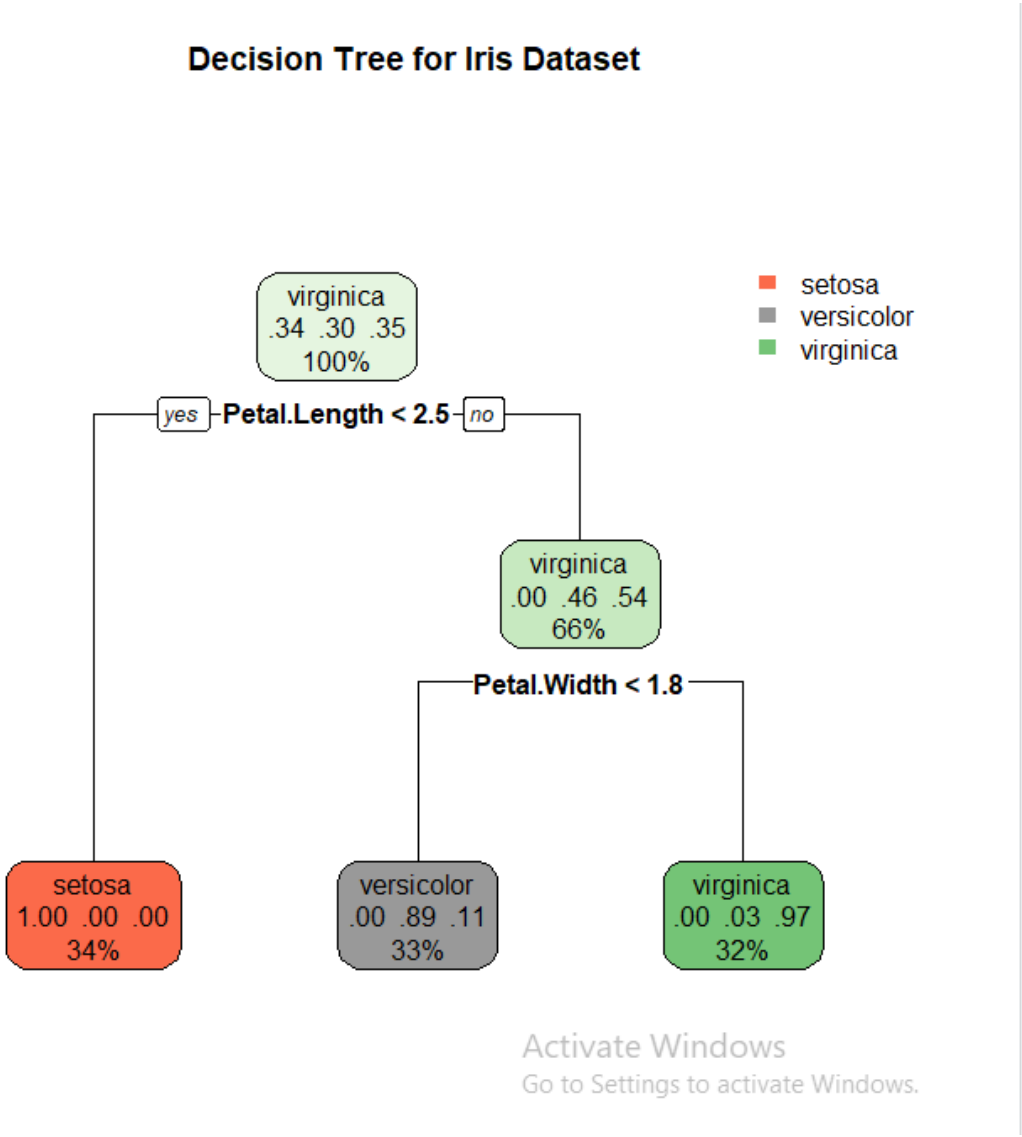


Result :

Program :

```
# Install & load required packages
if (!require("rpart")) install.packages("rpart")
if (!require("rpart.plot")) install.packages("rpart.plot")

library(rpart)
library(rpart.plot)

# Load dataset
data(iris)

# Split data into training (70%) and testing (30%)
set.seed(123)
sample_index <- sample(1:nrow(iris), 0.7 * nrow(iris))
train_data <- iris[sample_index, ]
test_data <- iris[-sample_index, ]

# Train Decision Tree model
tree_model <- rpart(Species ~ ., data = train_data, method = "class")

# Predict on test data
tree_pred <- predict(tree_model, test_data, type = "class")

# Confusion Matrix
cat("Decision Tree Confusion Matrix:\n")
print(table(Predicted = tree_pred, Actual = test_data$Species))

# Plot Decision Tree
rpart.plot(tree_model, main = "Decision Tree for Iris Dataset")
```

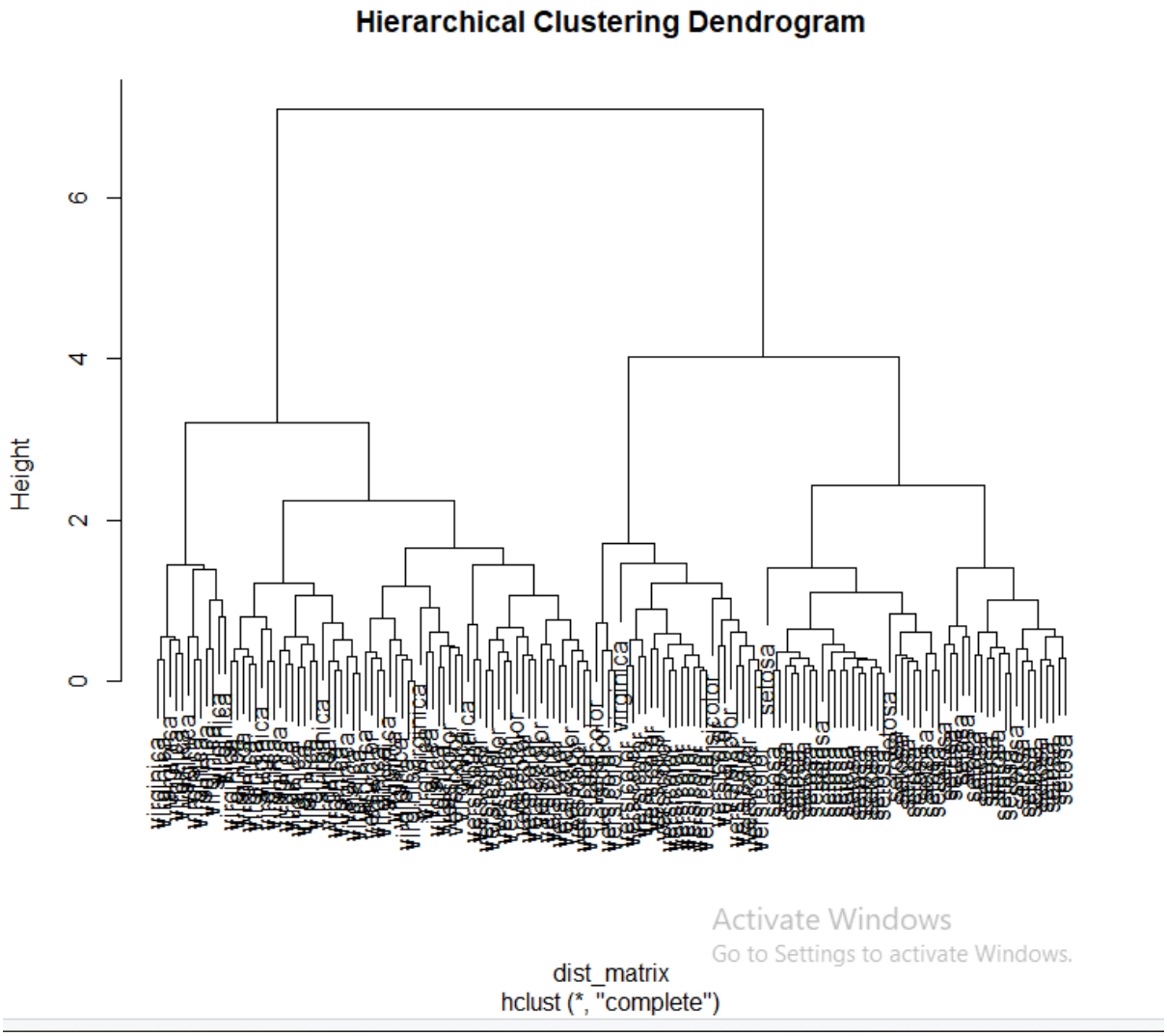
Output :


Decision Tree for Iris Dataset
setosa
versicolor
virginica
virginica
.34 .30 .35
100%
yes
Petal.Length < 2.5
no
virginica
.00 .46 .54
66%
Petal.Width < 1.8
setosa
1.00 .00 .00
34%
versicolor
.00 .89 .11
33%
virginica
.00 .03 .97
32%
Activate Windows
Go to Settings to activate Windows.

Program :

```
# Hierarchical Clustering on Iris
# Install & load required packages
if (!require("cluster")) install.packages("cluster")

library(cluster)

# Load dataset
data(iris)

# Use only numeric columns (remove Species)
iris_data <- iris[, -5]

# Compute distance matrix
dist_matrix <- dist(iris_data)

# Build hierarchical model using complete linkage
hc_model <- hclust(dist_matrix, method = "complete")

# Plot dendrogram with species labels
plot(hc_model, labels = iris$Species,
    main = "Hierarchical Clustering Dendrogram")

# Cut dendrogram into 3 clusters
hc_clusters <- cutree(hc_model, k = 3)

# Compare clusters with actual species
cat("\nHierarchical Clustering Assignments:\n")
print(table(Actual = iris$Species, Cluster = hc_clusters))
```

Output :



Result :