**Experiment No. 7**

**Title:** Demonstrate the Use of Map and Reduce Tasks

**Objective:**

To showcase the functionality and implementation of map and reduce tasks in a distributed computing environment using Hadoop MapReduce framework.

**Tools used:**

- Hadoop MapReduce framework

- Terminal or Command Prompt

**Prerequisite:**

- Installed Hadoop framework

- Basic understanding of Java programming

- Sample dataset for processing

**Theory:**

MapReduce is a programming model used to process and generate large datasets in a distributed computing environment. It divides tasks into two phases: the map phase, which processes input data and produces intermediate key-value pairs, and the reduce phase, which aggregates and processes intermediate results to generate the final output.

**Steps to Demonstrate Map and Reduce Tasks:**

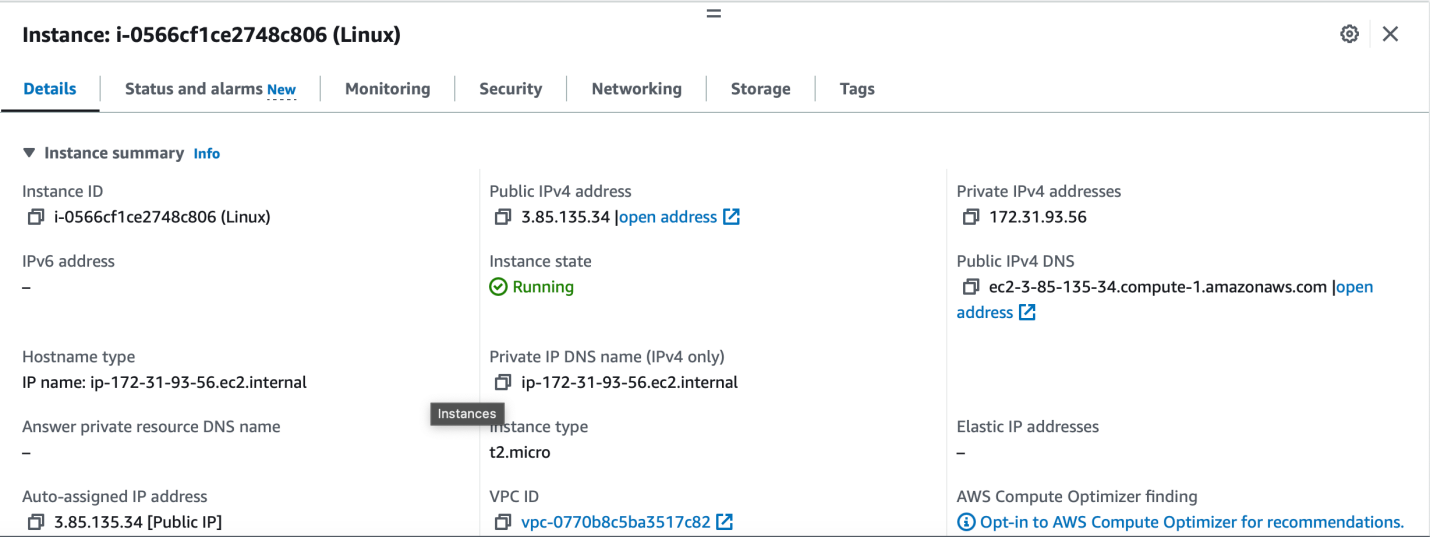
Absolutely! To demonstrate the MapReduce process using Java with Hadoop, here's an example for word count:

Sure, to set up an Amazon AWS cloud instance for running Hadoop and executing MapReduce jobs, you'll need to follow these steps:

**Step 1:**  Launch an EC2 Instance:

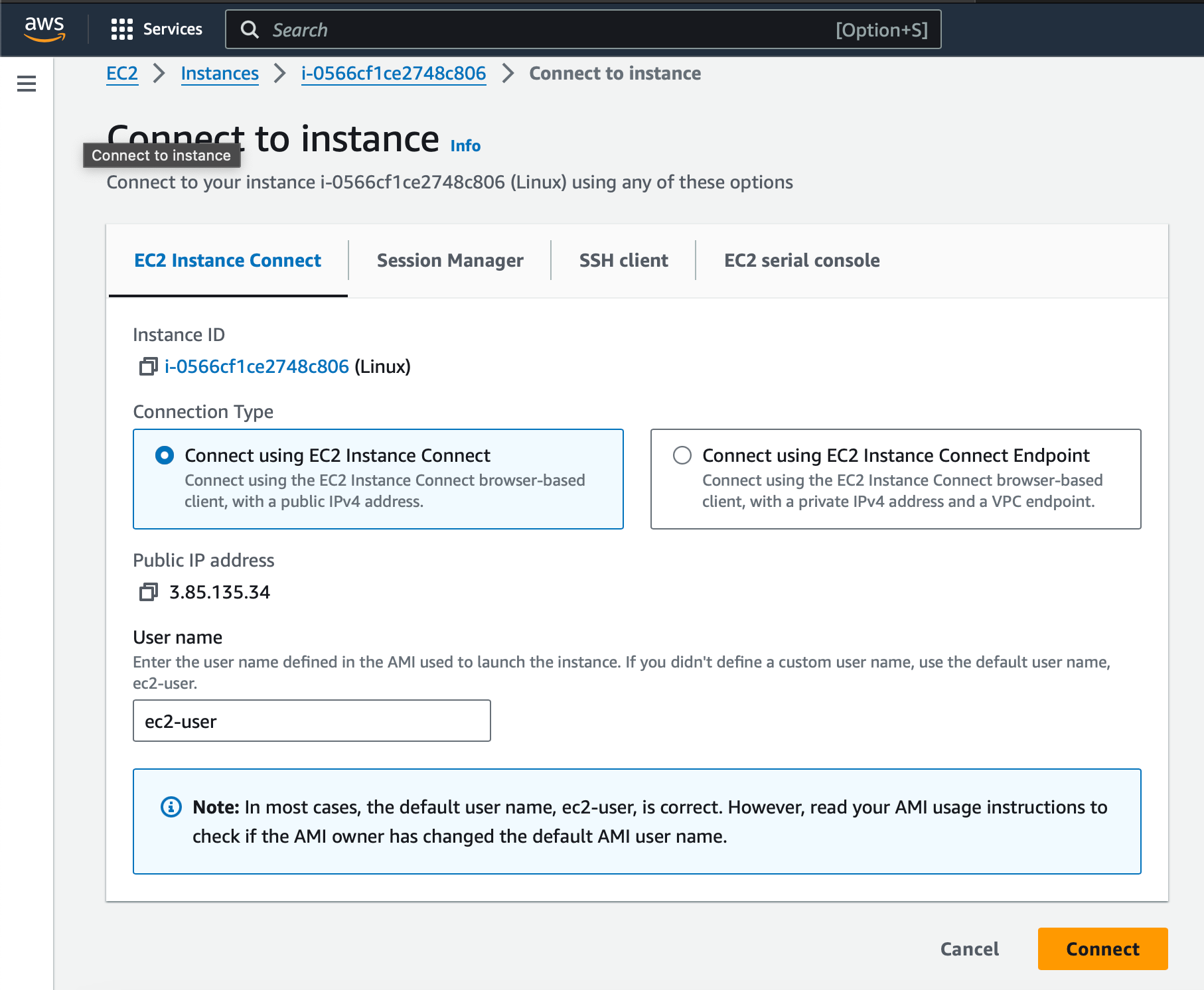
- Log in to your AWS Management Console and navigate to EC2.

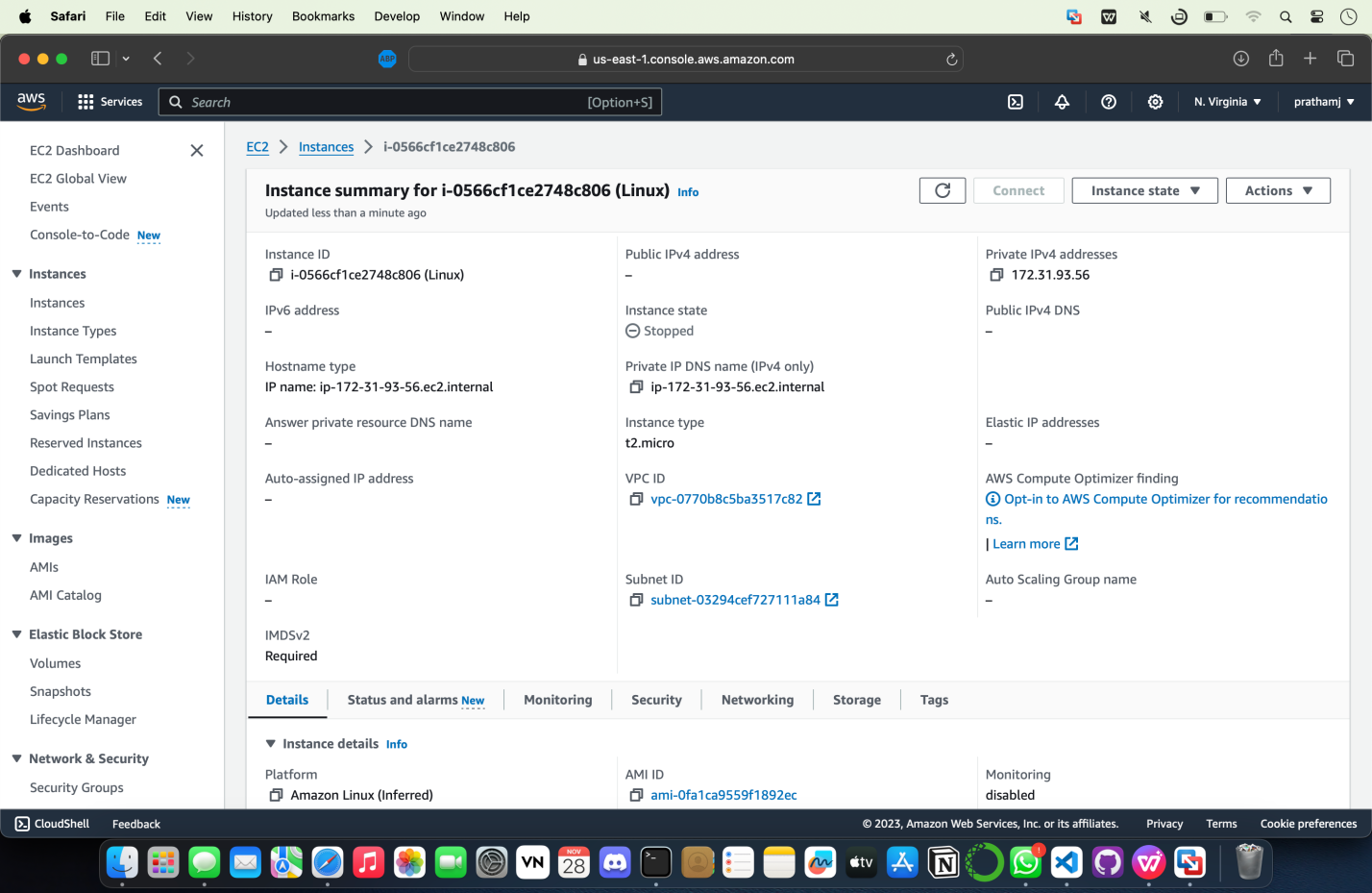
- Launch a new instance, choosing an Amazon Machine Image (AMI) based on Linux (Amazon Linux or Ubuntu) that fits your requirements. Ensure it has enough resources (RAM, CPU, storage) for your Hadoop setup.



**Step** 2. Connect to Your EC2 Instance:

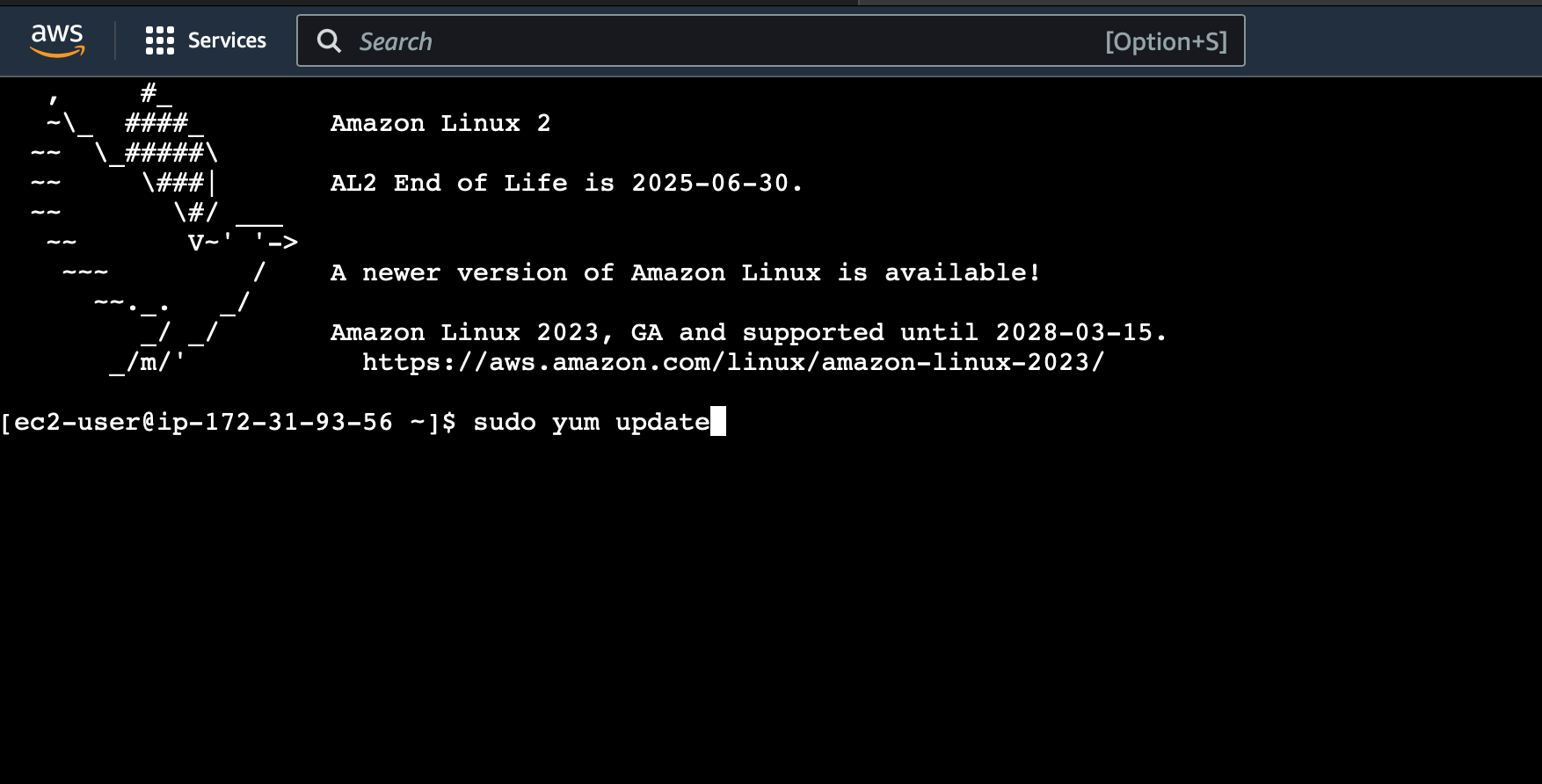
- Once the instance is running, connect to it via SSH using a terminal or an SSH client like PuTTY (for Windows). Use the key pair you generated during instance creation for authentication.



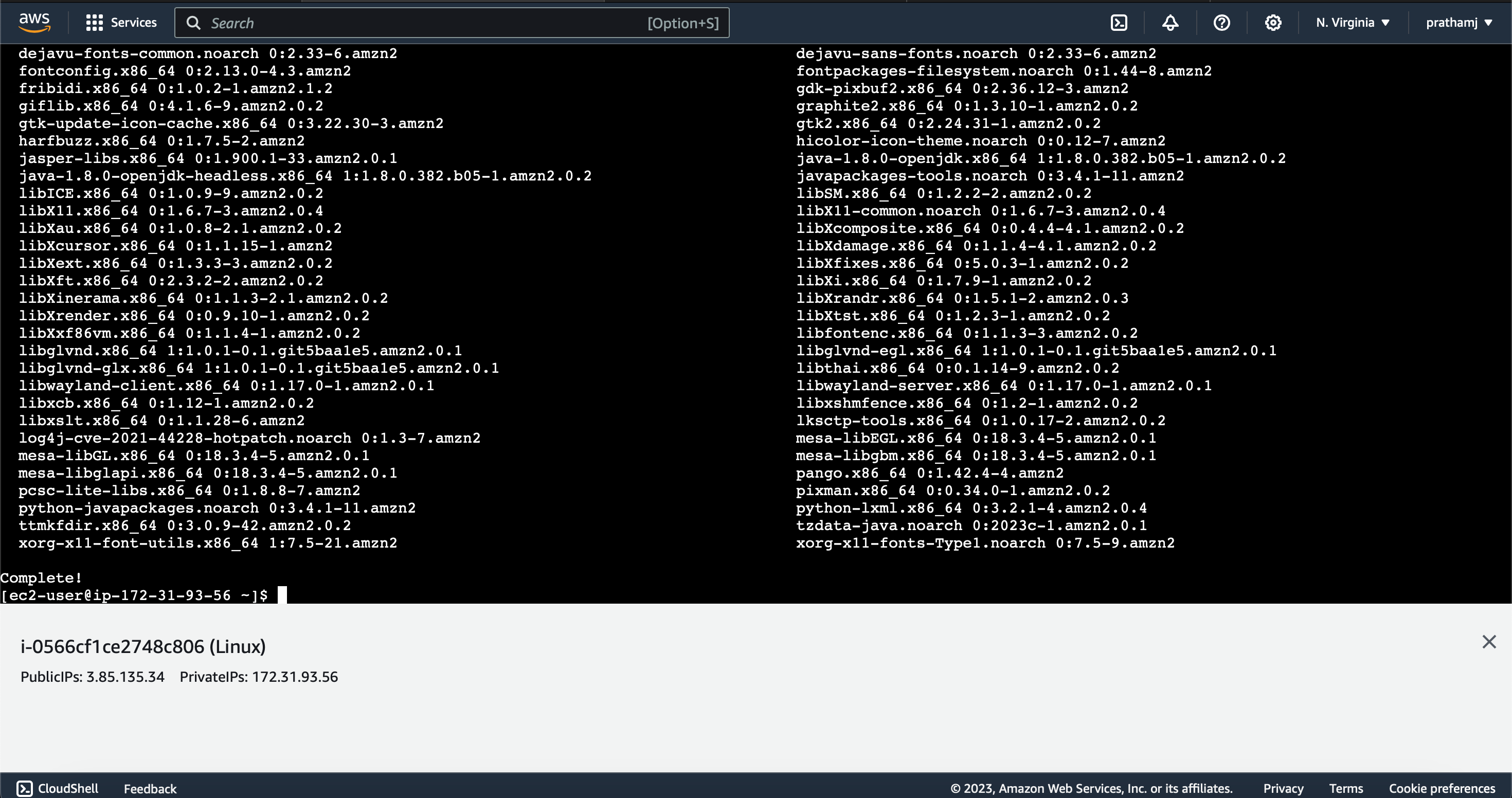


**Step** 3. Update and Install Necessary Packages:

- Update the system: `sudo yum update` (for Amazon Linux) or `sudo apt update && sudo apt upgrade` (for Ubuntu).

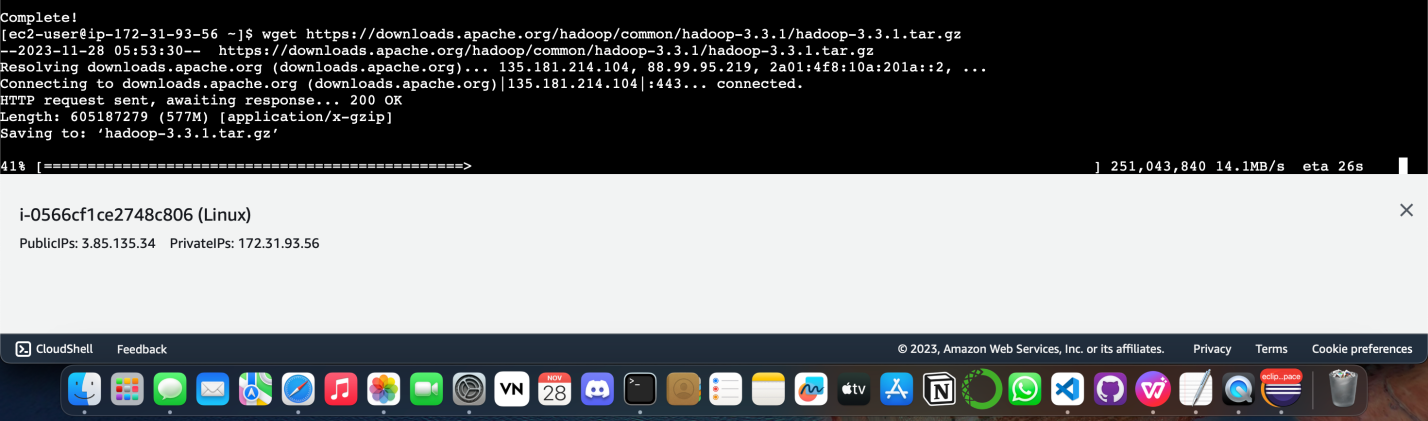


- Install Java Development Kit (JDK) if not already installed: `sudo yum install java-1.8.0-openjdk-devel` or `sudo apt install default-jdk`.

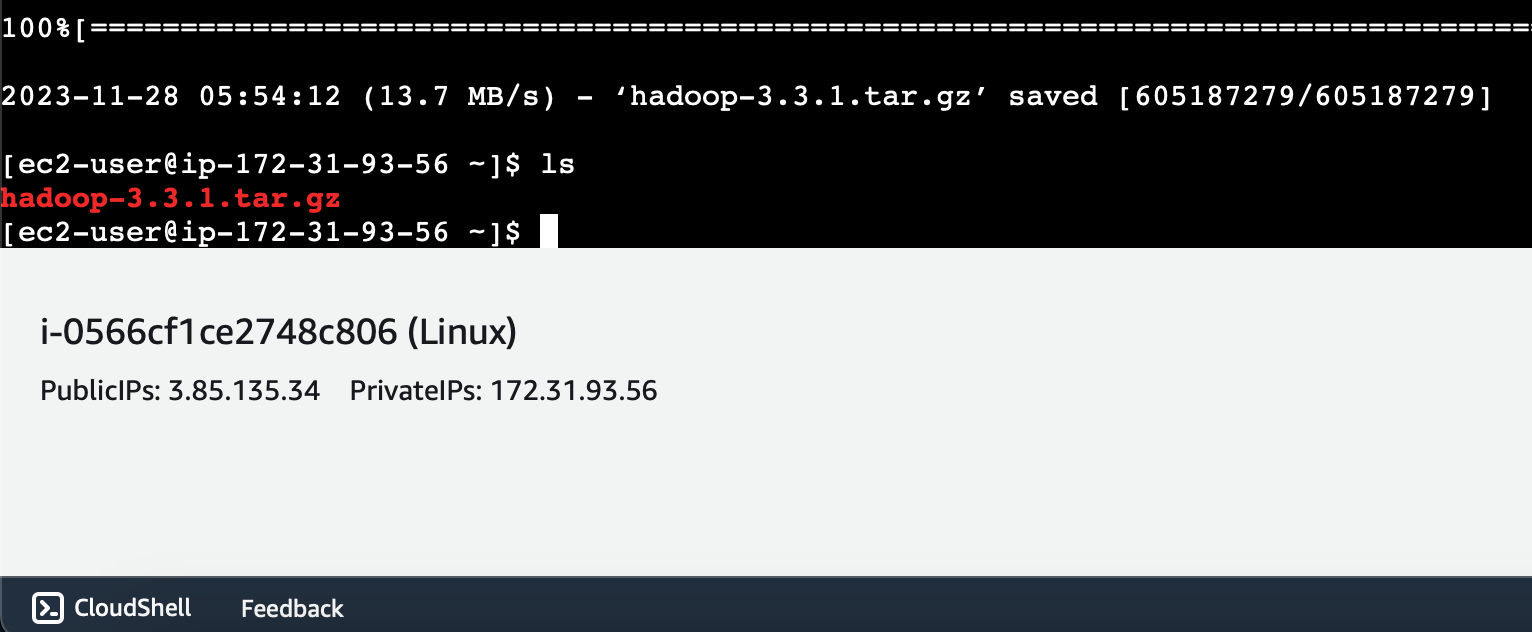


- Set up other required packages like `ssh`, `wget`, etc., depending on your specific needs.

**Step** 4. Download and Install Hadoop:

- Download Hadoop from the Apache Hadoop website or use `wget` directly on the EC2 instance.

- Extract the downloaded Hadoop tarball: `tar -xzf hadoop-3.x.x.tar.gz` (replace `3.x.x` with your downloaded version).



- Set up Hadoop environment variables like `HADOOP\_HOME`, `JAVA\_HOME`, and add them to the `.bashrc` or `.bash\_profile` file.

Setting up environment variables like `HADOOP\_HOME` and `JAVA\_HOME` involves editing the `.bashrc` or `.bash\_profile` file in your user's home directory on the EC2 instance. Here's a step-by-step guide to add these variables:

**1. Find the Paths for Hadoop and Java:**

You'll need the paths where Hadoop and Java are installed. If you've downloaded Hadoop as discussed earlier, your Hadoop path will be the directory where you extracted Hadoop. To find the Java path, you can use the `which java` command:

```bash

which java

```

**2. Edit `.bashrc` or `.bash\_profile`:**

Use a text editor like `nano` or `vi` to open the `.bashrc` or `.bash\_profile` file:

```bash

nano ~/.bashrc # or nano ~/.bash\_profile

```

**3. Add Environment Variables:**

Scroll to the bottom of the file and add the following lines:

```bash

export JAVA\_HOME=/path/to/your/java # Replace '/path/to/your/java' with the Java path you obtained

export HADOOP\_HOME=/path/to/your/hadoop # Replace '/path/to/your/hadoop' with the Hadoop path

export PATH=$PATH:$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin

```

Ensure you replace `/path/to/your/java` and `/path/to/your/hadoop` with the actual paths on your EC2 instance.

**4. Save and Apply Changes:**

Press `Ctrl + X`, then `Y` to save the changes in nano editor.

**5. Load the Updated Configuration:**

To apply the changes without logging out, use the following command:

```bash

source ~/.bashrc # or source ~/.bash\_profile

```

**6. Verify the Environment Variables:**

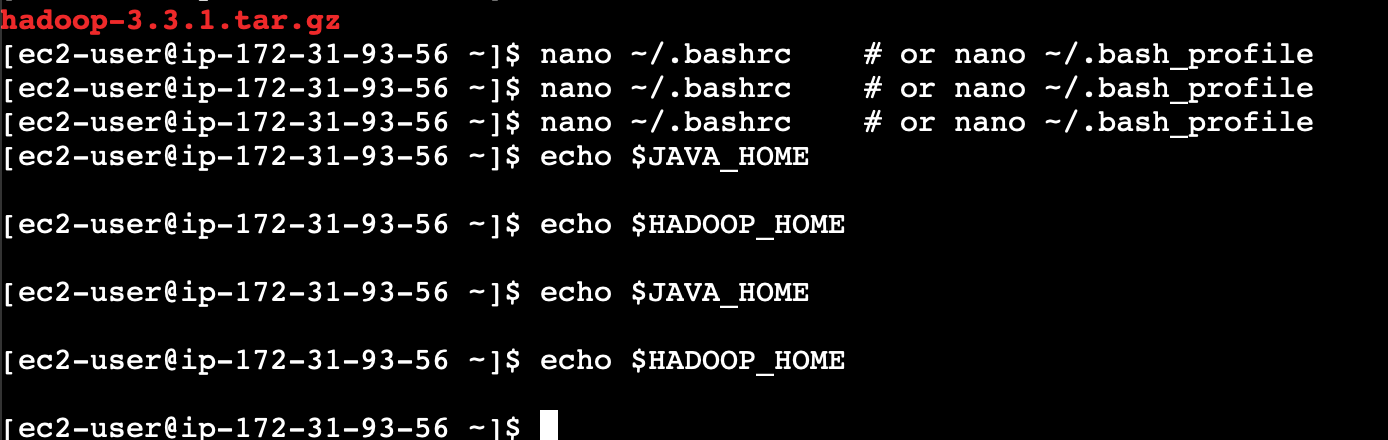
To confirm that the environment variables are set correctly, you can use `echo`:

```bash

echo $JAVA\_HOME

echo $HADOOP\_HOME

```

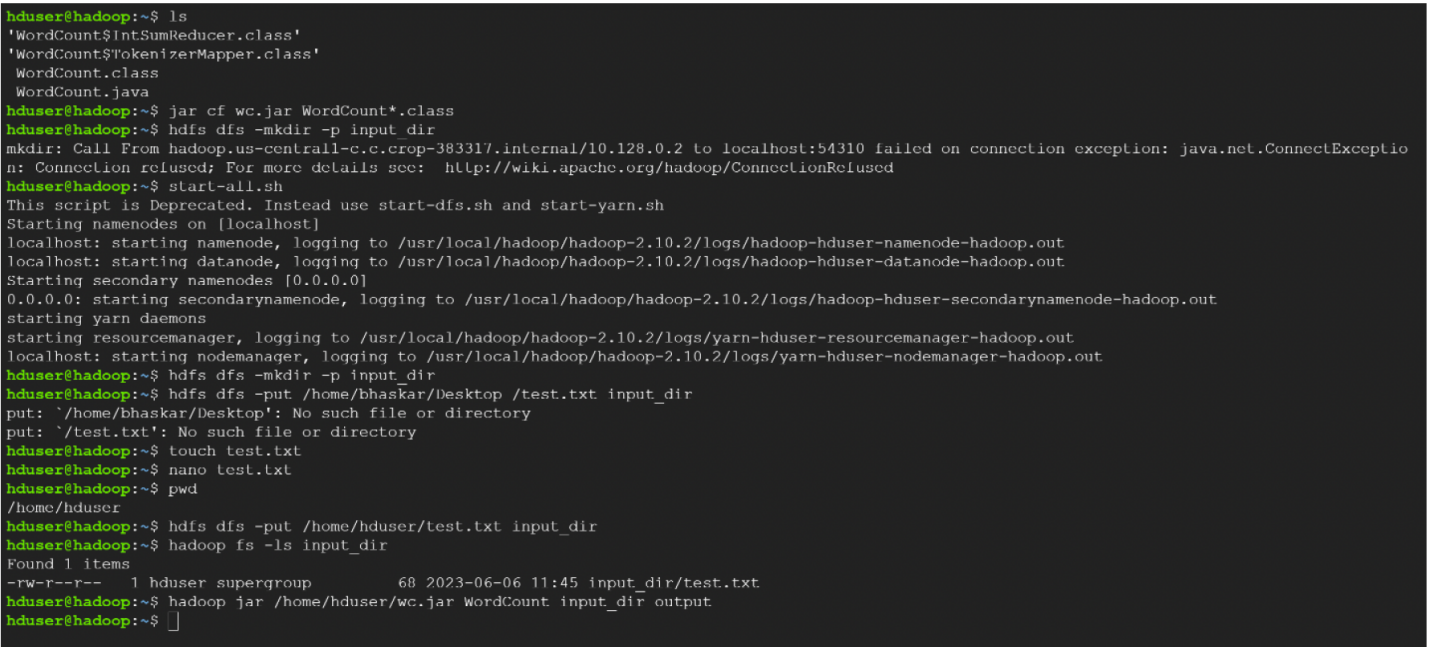


Setting these environment variables ensures that Hadoop and Java commands can be executed from any directory in the terminal without needing to provide their full paths.

After setting these variables, you should be able to run Hadoop commands like `hadoop`, `hdfs`, etc., from any directory within your EC2 instance's terminal session.

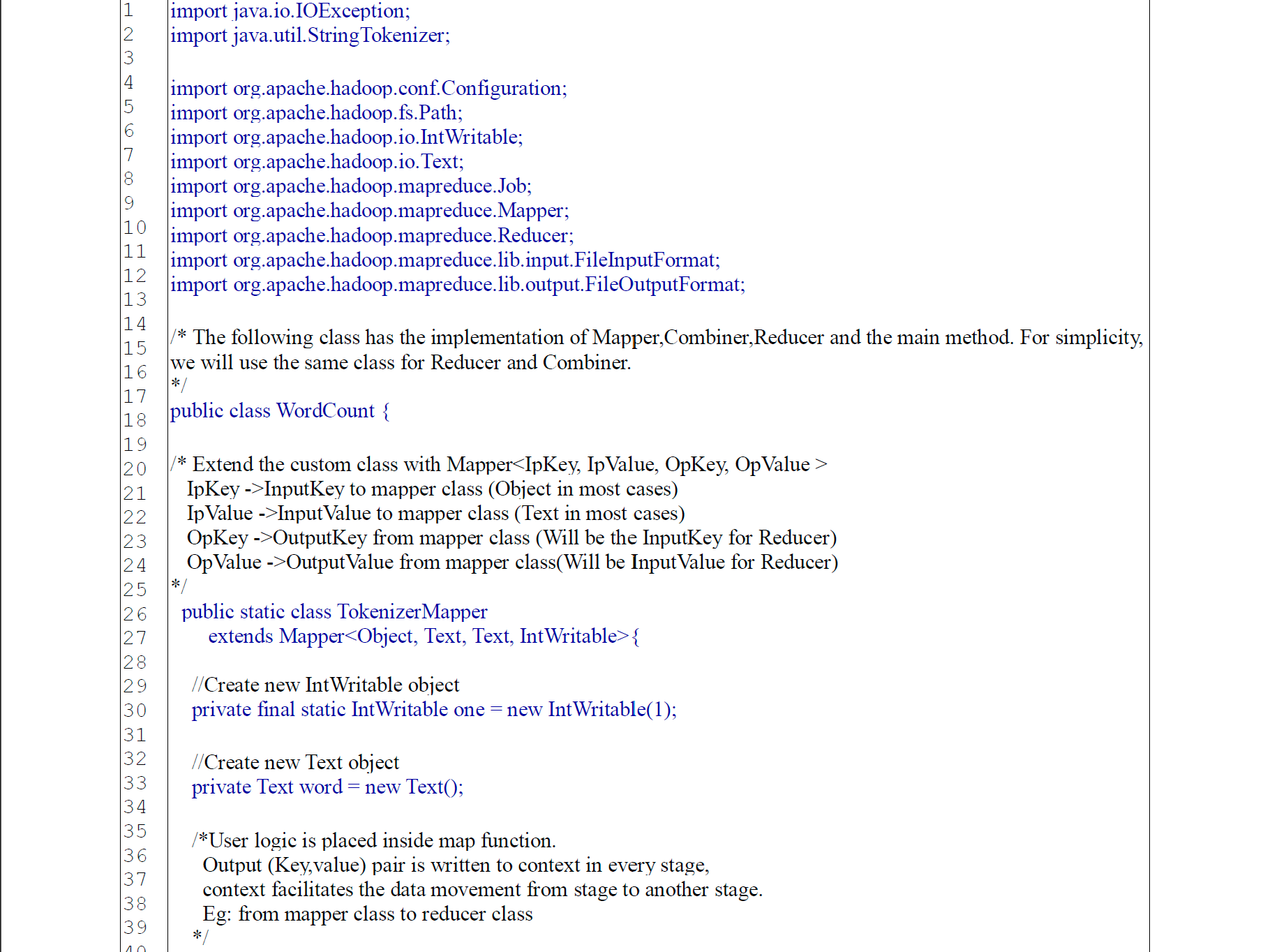
**Step** 5. Configuration:

- Configure Hadoop files such as `core-site.xml`, `hdfs-site.xml`, `mapred-site.xml`, and `yarn-site.xml` located in the Hadoop configuration directory (`$HADOOP\_HOME/etc/hadoop/`). Modify these files according to your cluster setup and specifications.



**Step** 6. Upload Your Code and Data:

- Transfer your Java MapReduce code (`WordCountMapper.java`, `WordCountReducer.java`, and the driver) and the sample input data file (`input.txt`) to your EC2 instance using `scp` or `sftp` from your local machine.



**Step** 7. Compile Your MapReduce Code:

- Compile your Java files (`WordCountMapper.java`, `WordCountReducer.java`, and the driver) using `javac`. Ensure Hadoop libraries are included in the classpath during compilation.

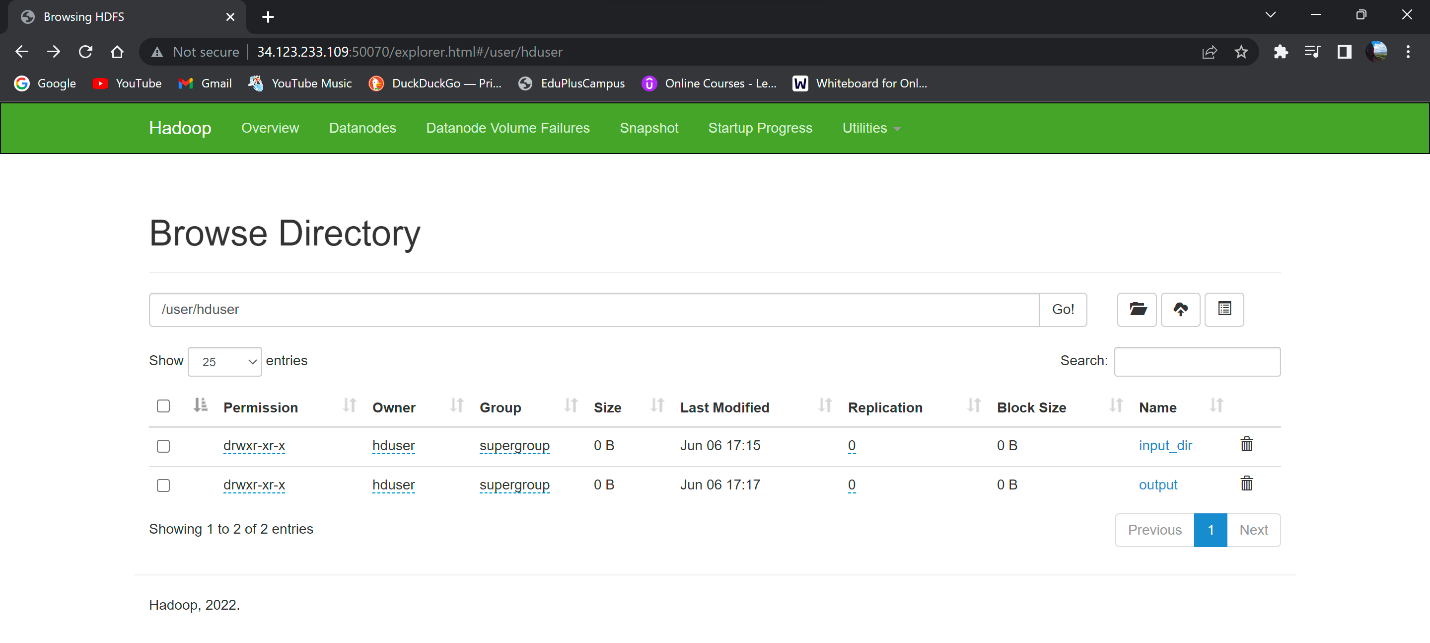
**Step** 8. Run Your MapReduce Job:

- Execute your MapReduce job using the Hadoop command-line interface (`hadoop jar` command). Ensure to provide the correct input and output paths.

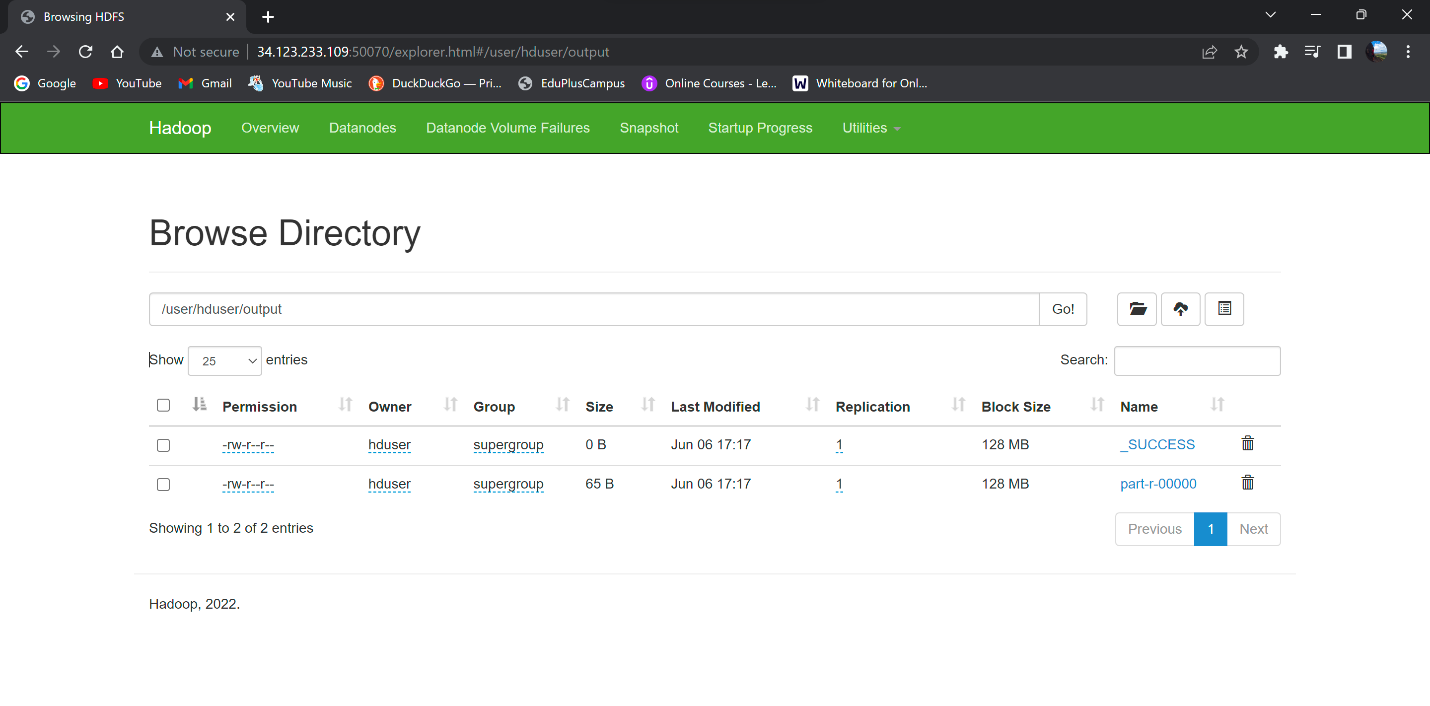
**Step** 9. Monitor Job Execution:

- Use Hadoop's command-line tools (`yarn` or `mapred` commands) or access the Hadoop web interface to monitor the job's progress and status.

**Step** 10. Review Output:

- Once the job completes, check the output directory specified in your Hadoop job submission for the results (`output\_directory`).

Sucess



Remember, configuring Hadoop on AWS EC2 involves setting up security groups, configuring ports, and potentially additional settings for AWS-specific networking and storage services. Also, manage your EC2 instance security by restricting access through security groups and using key pairs for SSH access.

Be cautious with AWS resources to avoid unnecessary charges, especially when running instances or services. Adjust the instance type and configurations based on your processing needs and budget.

**Conclusion:**

Successfully demonstrated the use of map and reduce tasks in a distributed computing environment using the Hadoop MapReduce framework. This experiment highlighted the process of dividing computational tasks into smaller map and reduce phases, showcasing how Hadoop efficiently processes large datasets by leveraging parallel processing and distributed computing techniques.