# Project Report: Word Count on AWS with PySpark, Node.js Deployment

# by-Vivek Sai Chinna Burada

## Objective

This project demonstrates two main tasks:

1. **Word Count on AWS EC2/LightSail using PySpark**: Setting up an EC2/LightSail instance to count words in a text file stored in an S3 bucket using PySpark.
2. **Node.js Deployment with Docker**: Deploying a simple Node.js web server in a Docker container and hosting it on an EC2 instance.

---

## Project 1: Word Count on AWS EC2/LightSail using PySpark

### 1. Prerequisites

Before starting, ensure that you have the following:

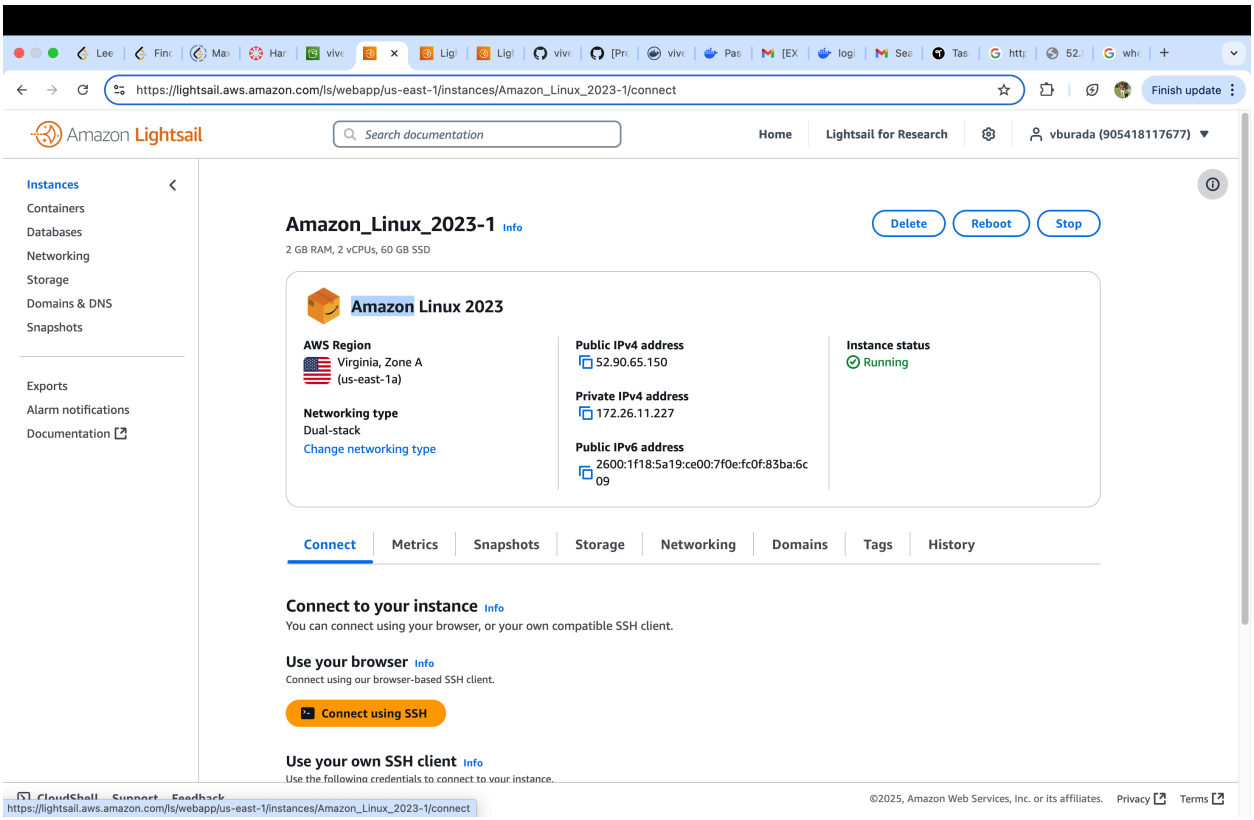
- An **AWS account** with access to **EC2**/LightSail.
- An **S3 bucket** containing the input text file.
- **SSH access** to your EC2/LightSail instance.

### 2. Set Up EC2/LightSail Instance

Follow these steps to set up the EC2/LightSail instance:

1. **Launch an Amazon Linux 2 Instance**:
   Launch the instance from the AWS Management Console.
2. **Connect to the Instance via SSH**:
   Use the following command to connect to the EC2 instance:

   ```
   bash
   CopyEdit
   ssh -i "your-key.pem" ec2-user@your-instance-ip
   ```




3. **Install Java 11**:
   Java is required for running Spark. Install Java 11:

```
bash
CopyEdit
sudo yum install java-11 -y
export JAVA_HOME=$(dirname $(dirname $(readlink -f $(which java))))
java -version
```

4. **Increase `/tmp` Size**:
   Increase the `/tmp` directory size to prevent Spark errors:

```
bash
CopyEdit
sudo mount -o remount,size=2G /tmp
```

5. **Install Python and PySpark**:
   Install Python and PySpark:

```
bash
CopyEdit
sudo yum install python3-pip -y
pip install pyspark
spark-submit --version
```

## 3. Word Count Script

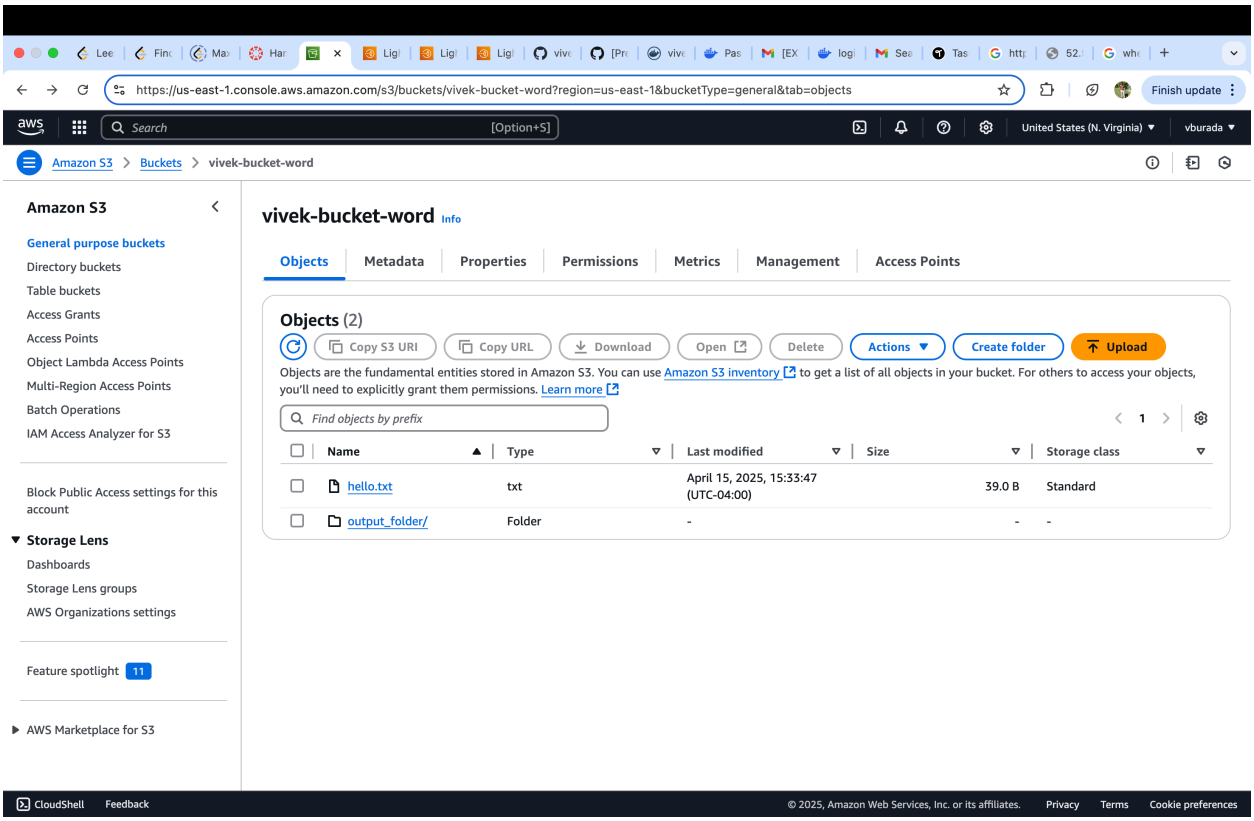
Create a Python script `word_count.py` to perform the word count operation.

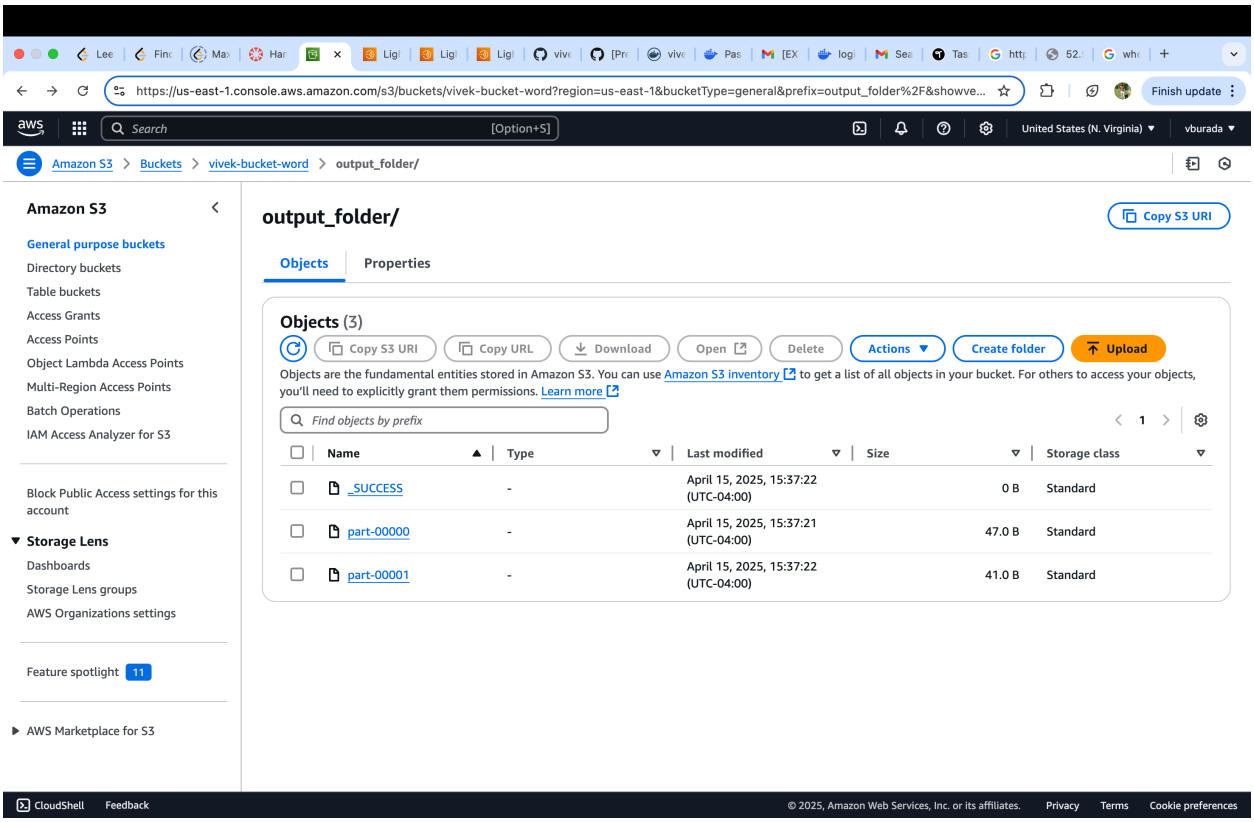
```
python
CopyEdit
from pyspark.sql import SparkSession

# AWS Credentials
AWS_ACCESS_KEY_ID = 'YOUR_ACCESS_KEY'
AWS_SECRET_ACCESS_KEY = 'YOUR_SECRET_KEY'

# S3 paths
S3_INPUT = 's3a://your-bucket-name/input_file.txt'
S3_OUTPUT = 's3a://your-bucket-name/output_folder/'

# Spark Session
spark = SparkSession.builder \
    .appName("WordCount") \
    .config("spark.jars.packages", "org.apache.hadoop:hadoop-
aws:3.3.1,com.amazonaws:aws-java-sdk-bundle:1.11.901") \
    .getOrCreate()

# Hadoop S3 Configuration
hadoop_conf = spark.sparkContext._jsc.hadoopConfiguration()
hadoop_conf.set("fs.s3a.access.key", AWS_ACCESS_KEY_ID)
hadoop_conf.set("fs.s3a.secret.key", AWS_SECRET_ACCESS_KEY)

# Word Count Logic
text_file = spark.sparkContext.textFile(S3_INPUT)
counts = text_file.flatMap(lambda line: line.split()) \
                  .map(lambda word: (word, 1)) \
                  .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile(S3_OUTPUT)
spark.stop()
```

## 4. Running the Script

Run the word count script using `spark-submit`:

```
bash
CopyEdit
spark-submit word_count.py
```


https://us-east-1.console.aws.amazon.com/s3/buckets/vivek-bucket-word?region=us-east-1&bucketType=general&tab=objects
Finish update
Search
[Option+S]
United States (N. Virginia)
vburada
Amazon S3 > Buckets > vivek-bucket-word
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 11
AWS Marketplace for S3
vivek-bucket-word Info
Objects
Metadata
Properties
Permissions
Metrics
Management
Access Points
Objects (2)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
hello.txt
txt
April 15, 2025, 15:33:47 (UTC-04:00)
39.0 B
Standard
output_folder/
Folder
-
-
-
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/s3/buckets/vivek-bucket-word?region=us-east-1&bucketType=general&prefix=output_folder%2F&showve...
Finish update
Search
[Option+S]
United States (N. Virginia)
vburada
Amazon S3 > Buckets > vivek-bucket-word > output_folder/
Amazon S3
General purpose buckets
Directory buckets
Table buckets
Access Grants
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
IAM Access Analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
Storage Lens groups
AWS Organizations settings
Feature spotlight 11
AWS Marketplace for S3
output_folder/
Copy S3 URI
Objects
Properties
Objects (3)
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Find objects by prefix
Name
Type
Last modified
Size
Storage class
_SUCCESS
-
April 15, 2025, 15:37:22 (UTC-04:00)
0 B
Standard
part-00000
-
April 15, 2025, 15:37:21 (UTC-04:00)
47.0 B
Standard
part-00001
-
April 15, 2025, 15:37:22 (UTC-04:00)
41.0 B
Standard
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

# Project 2: Deploy a Node.js Web Server with Docker

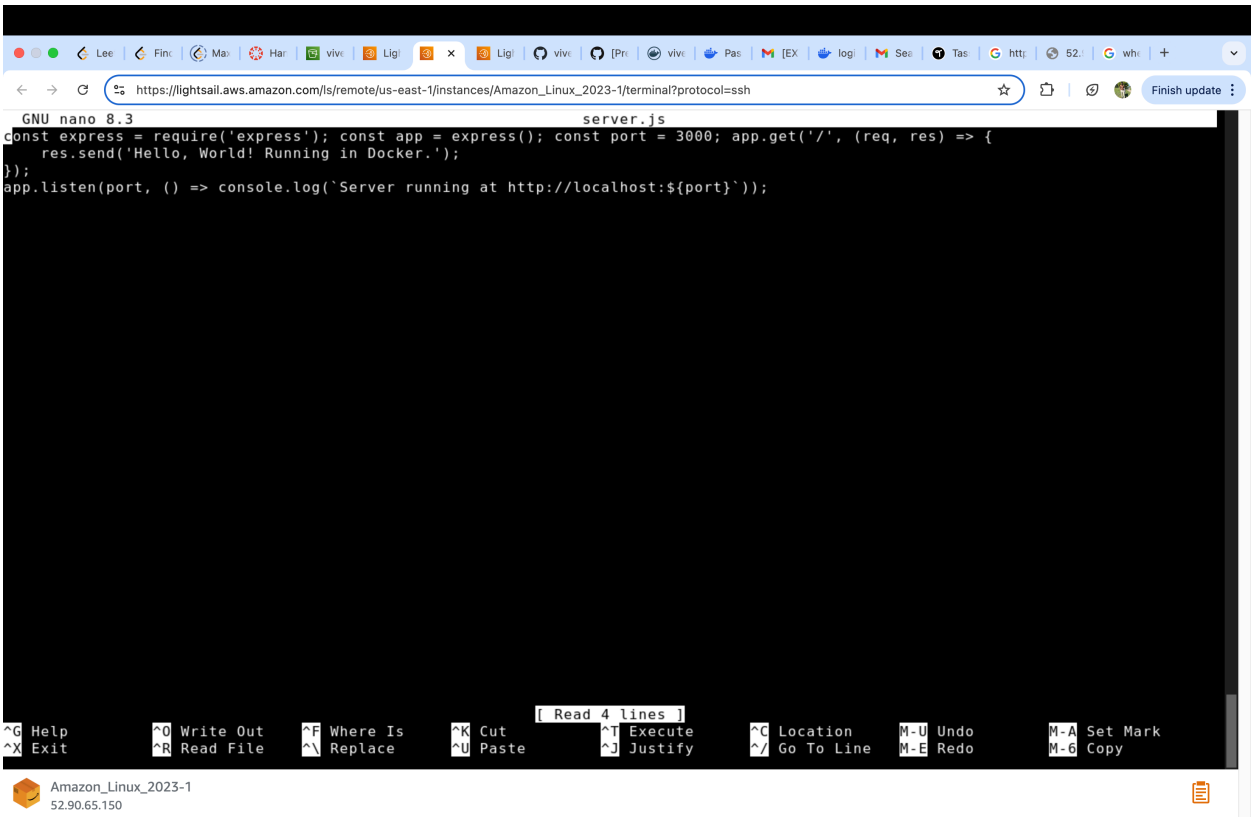
## 1. Node.js Server

1. **Create a Directory and Initialize the Project**:
   Create a directory and initialize a Node.js project:

   ```
   bash
   CopyEdit
   mkdir node-webserver && cd node-webserver
   npm init -y
   npm install express
   ```

2. **Create the `server.js` File**:
   The `server.js` file contains a simple Express server:

   ```
   javascript
   CopyEdit
   const express = require('express');
   const app = express();
   const port = 3000;

   app.get('/', (req, res) => {
       res.send('Hello, World! Running in Docker.');
   });

   app.listen(port, () => console.log(`Server running at
   http://localhost:${port}`));
   ```




## 2. Dockerize the Application

1. **Create the `Dockerfile`:**
   Create a `Dockerfile` to define how the application will be containerized:

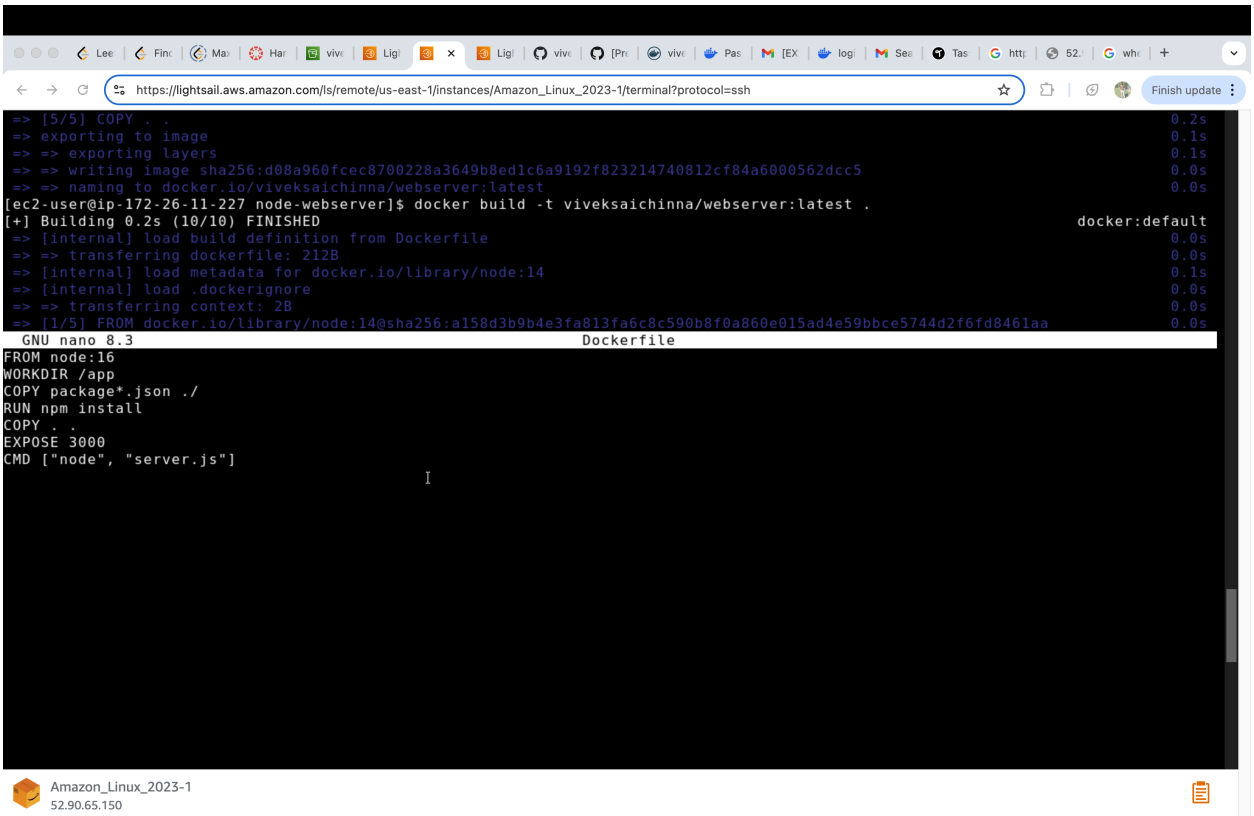
   ```
   dockerfile
   CopyEdit
   FROM node:14
   WORKDIR /app
   COPY package*.json ./
   RUN npm install
   COPY . .
   EXPOSE 3000
   CMD ["node", "server.js"]
   ```

2. **Build and Push the Docker Image to Docker Hub**: Build the Docker image:

   ```
   bash
   CopyEdit
   docker build -t your-dockerhub-username/webserver:latest .
   ```

   Push the image to Docker Hub:

   ```
   bash
   CopyEdit
   docker push your-dockerhub-username/webserver:latest
   ```



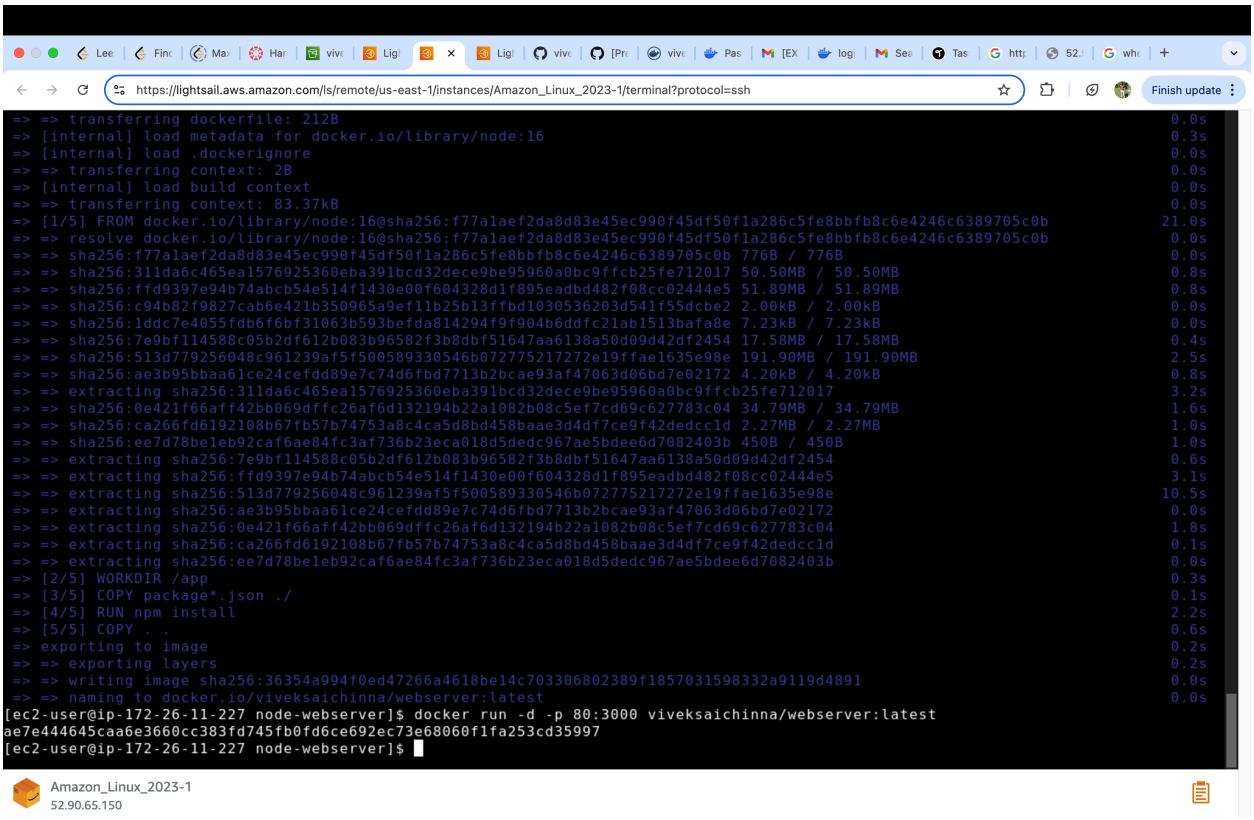

## 3. Deploy on EC2

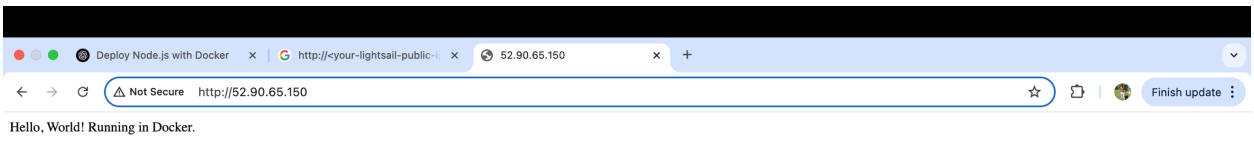
1. **Install Docker on EC2**: Update and install Docker:

   ```
   bash
   CopyEdit
   sudo yum update -y
   sudo yum install docker -y
   sudo service docker start
   sudo usermod -aG docker ec2-user
   ```

2. **Run the Container**:
   Pull the image and run the container:

   ```
   bash
   CopyEdit
   docker pull your-dockerhub-username/webserver:latest
   docker run -d -p 80:3000 your-dockerhub-username/webserver:latest
   ```




Hello, World! Running in Docker.

URL of we app hosted:

http://52.90.65.150/

---

# Conclusion

In this project, we successfully:

- Set up an EC2/LightSail instance to run PySpark for counting words in a text file stored in an S3 bucket.
- Deployed a Node.js server in a Docker container and hosted it on an EC2 instance.

---

# GitHub Repository Link

# https://github.com/viveksaichinna/word-count-spark

---

### Word Count:

The total word count for this report, excluding the code snippets and instructions, is

('name', 1)

('is', 1)

('sai', 1)

('chinna', 1)

('hello', 1)

('vivek', 1)

('burada.', 1)