**A REPORT SUBMISSION ON**

**16 JUNE 2024**

**DEVOPS PROJECT**

**HOSTING APPLICATION**

**Submitted to 3RI Technologies**

**Submitted by**

**Ashutosh Gore**

**Under the Guidance of**

**Mr. Manish Badgujar**

**(Trainer)**

**DevOps CI/CD Project**

**Introduction**

In this project, I integrated Terraform, AWS, GitHub, Jenkins, and Docker to create a robust CI/CD pipeline for hosting an application. The following documentation outlines the steps taken to achieve this.

**Step 1: Creating AWS Infrastructure with Terraform**

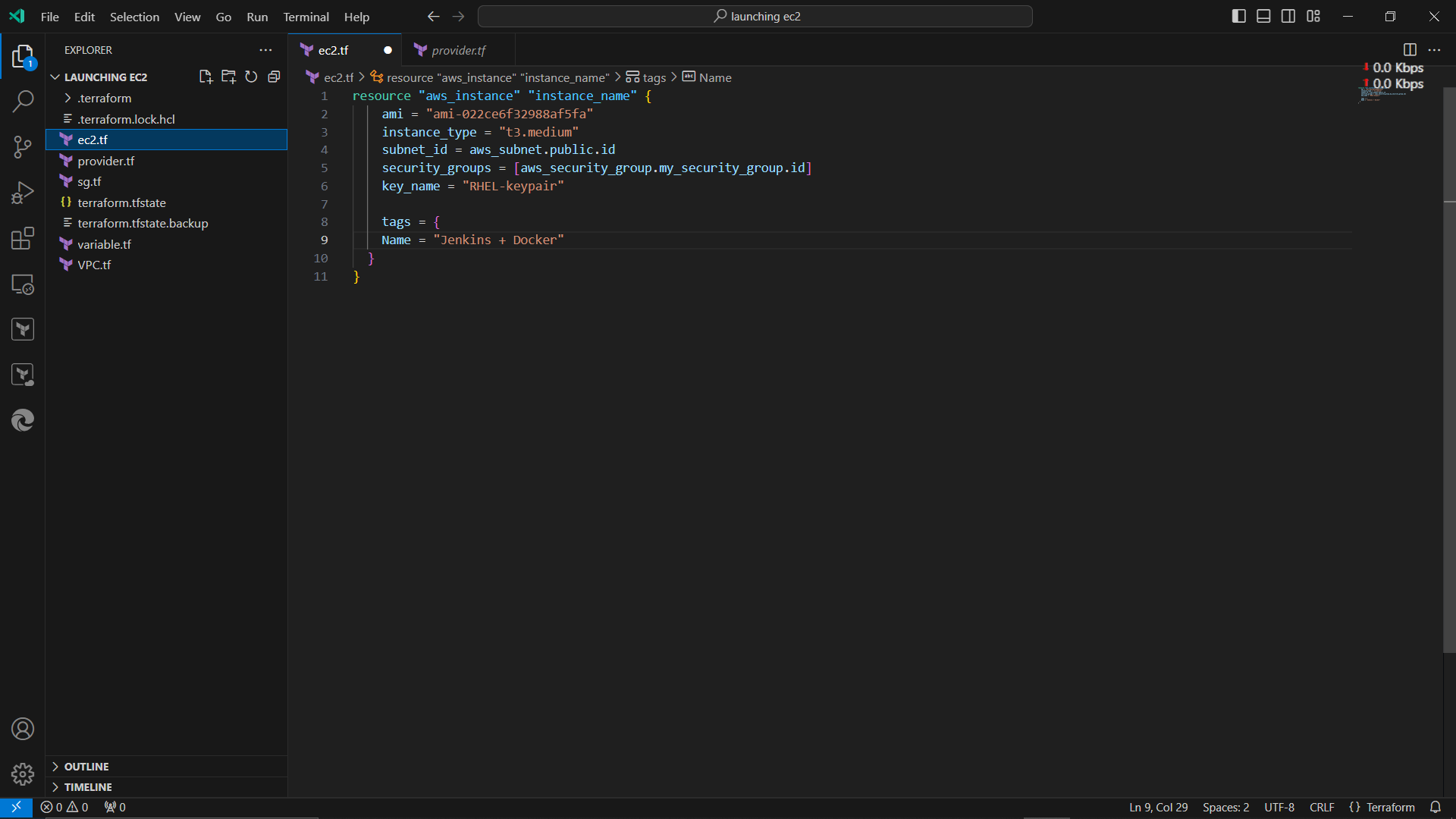
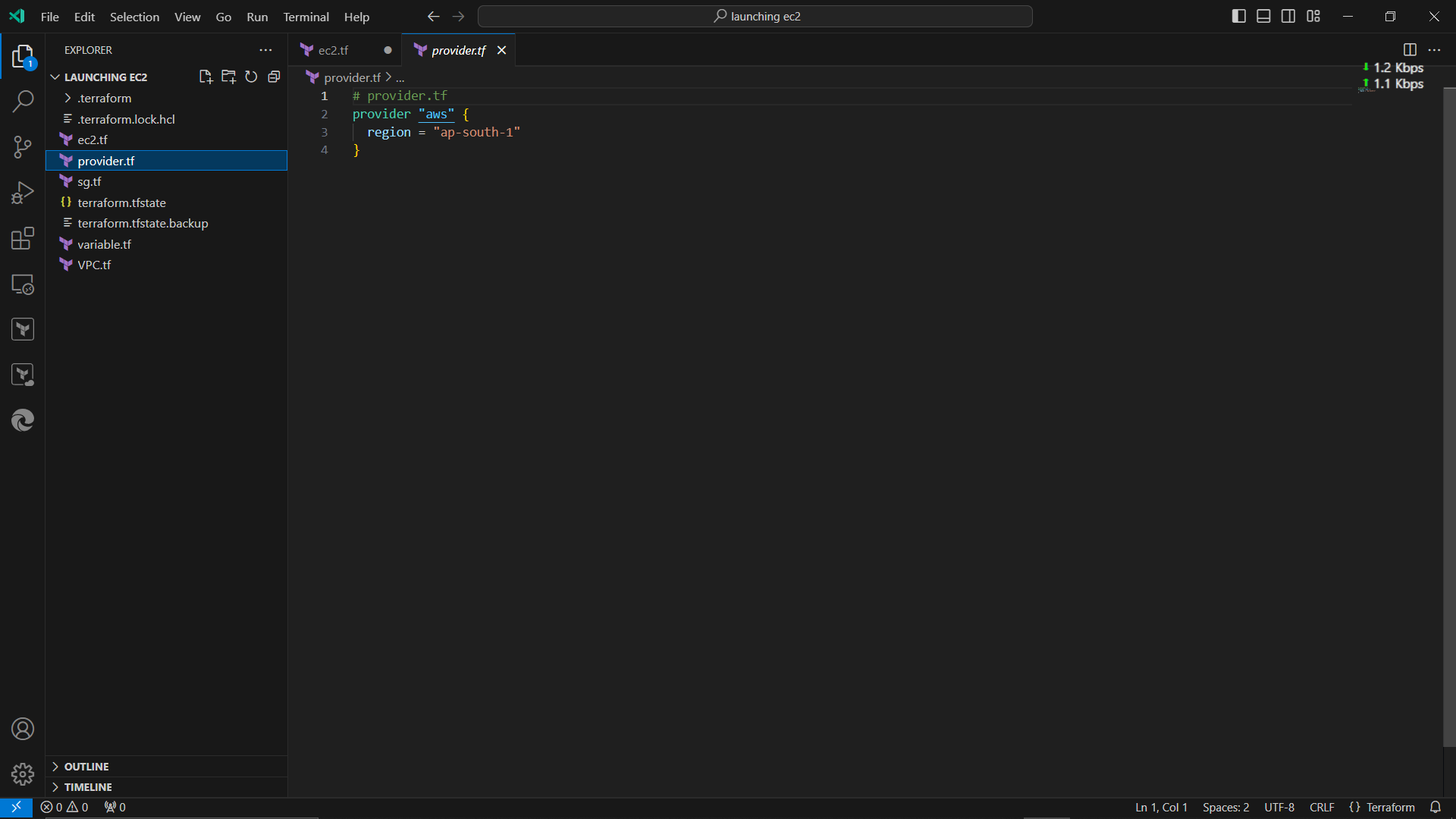
**Tools Used**

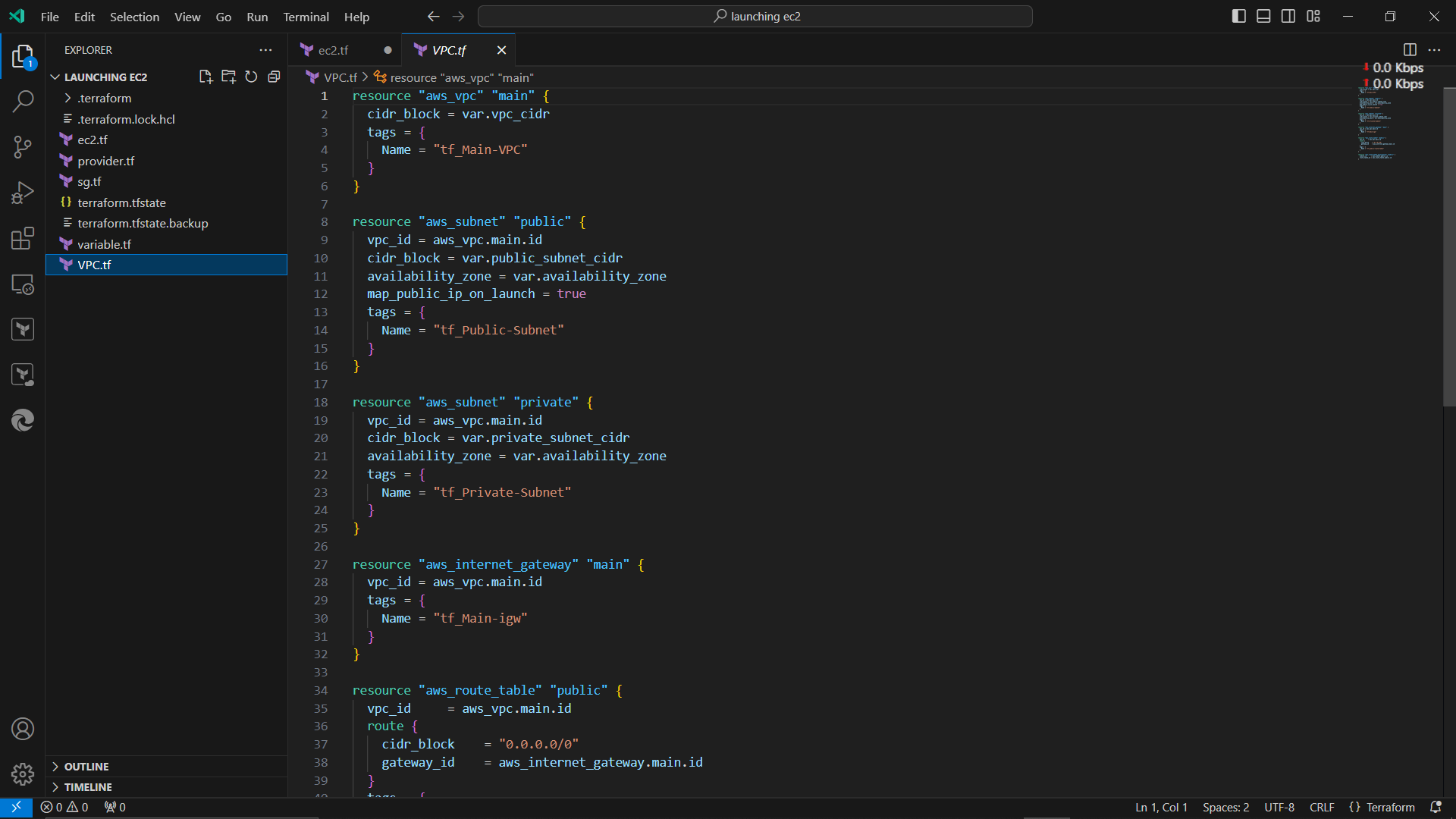
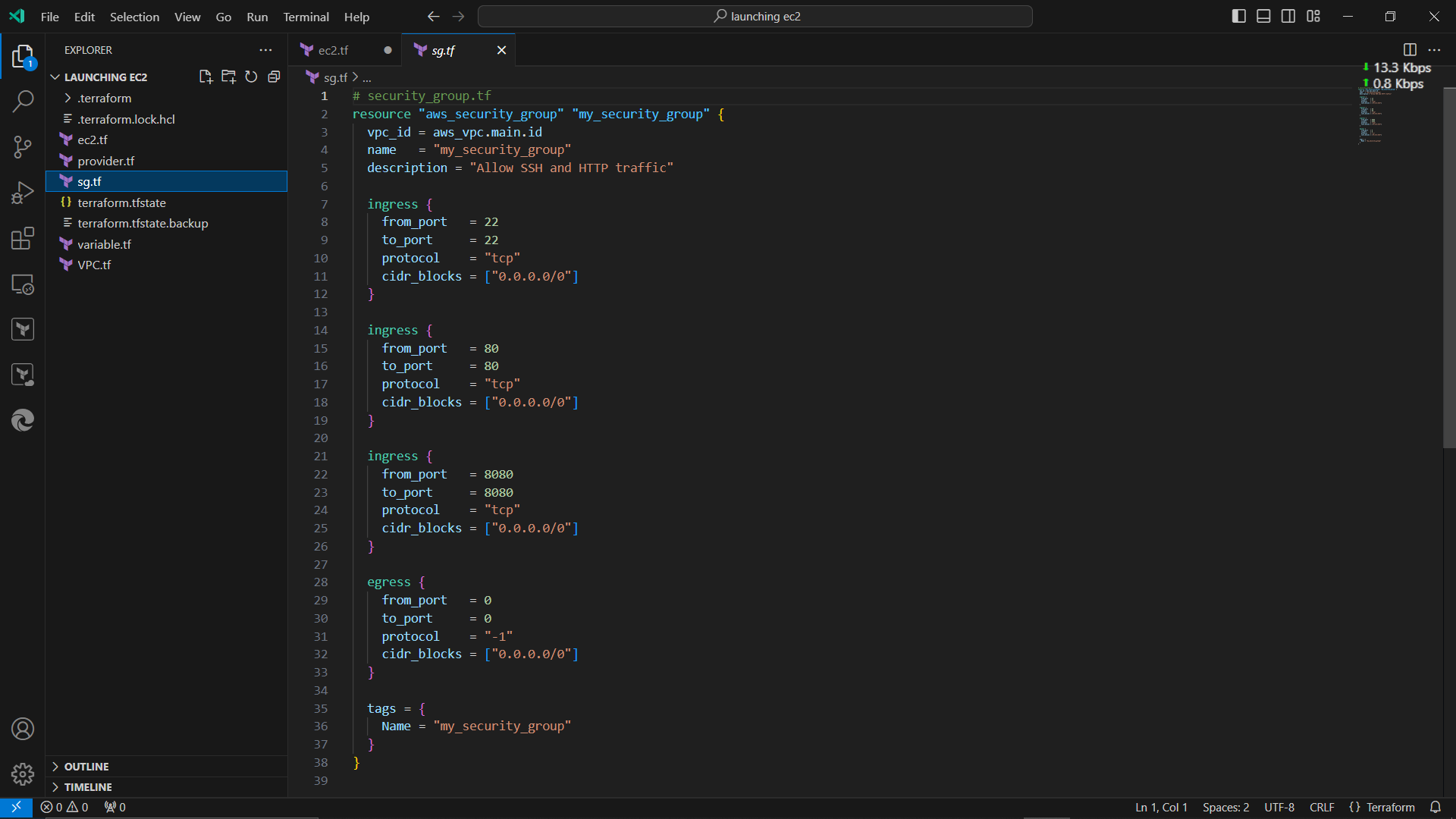
* **Terraform**: An open-source infrastructure as a code software tool.
* **Visual Studio Code**: A source-code editor used for writing and managing Terraform scripts.

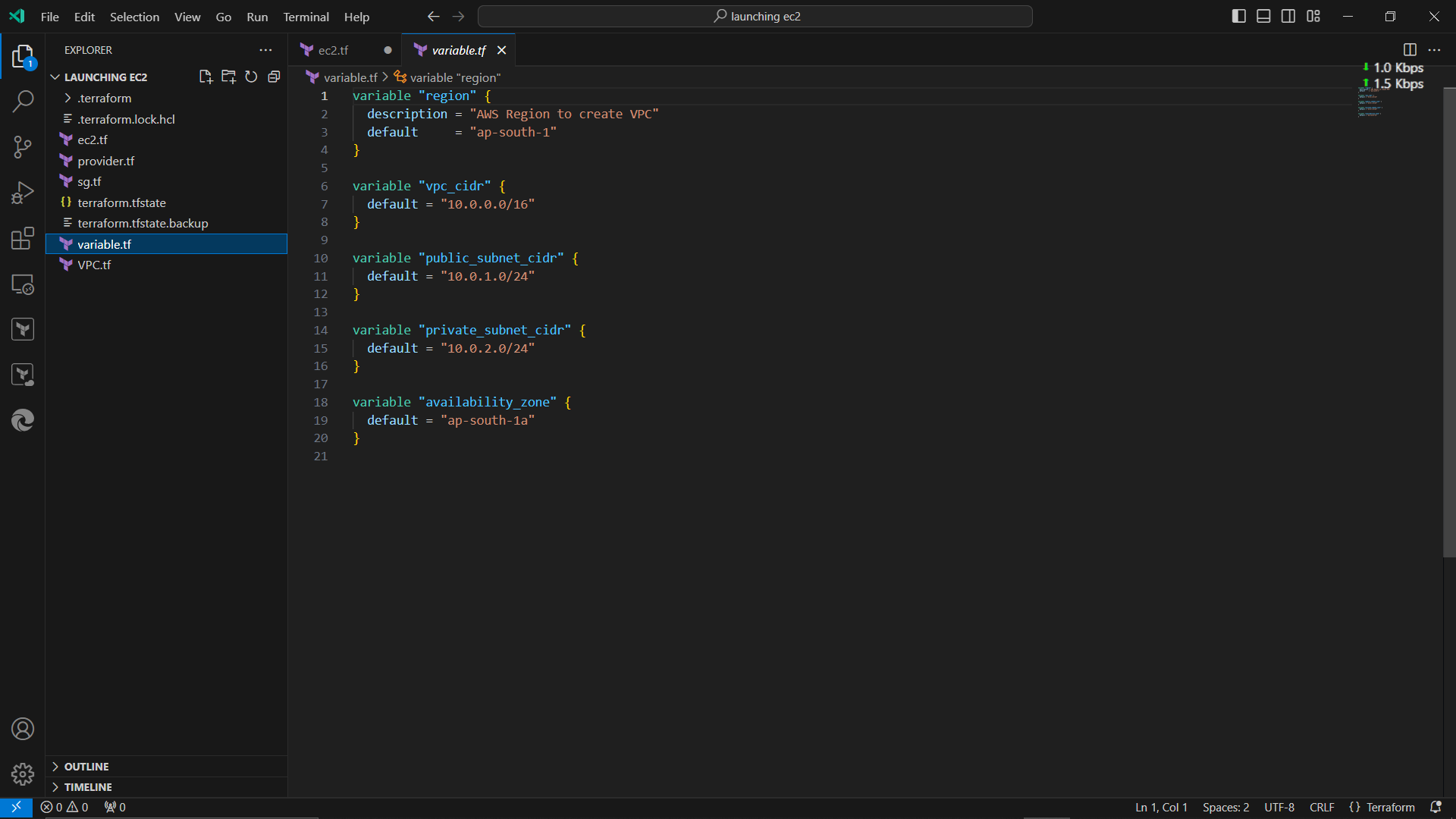
**Objective**

To provision and manage the necessary AWS infrastructure including EC2 instances, VPCs, and security groups using Terraform.

1. **Set Up Your Terraform Environment**
   * Ensure Terraform is installed on your machine.
   * Install Visual Studio Code.
2. **Write the Terraform Code**
   * Open Visual Studio Code and create a new directory for your Terraform scripts (‘LAUNCHING EC2’ directory name).

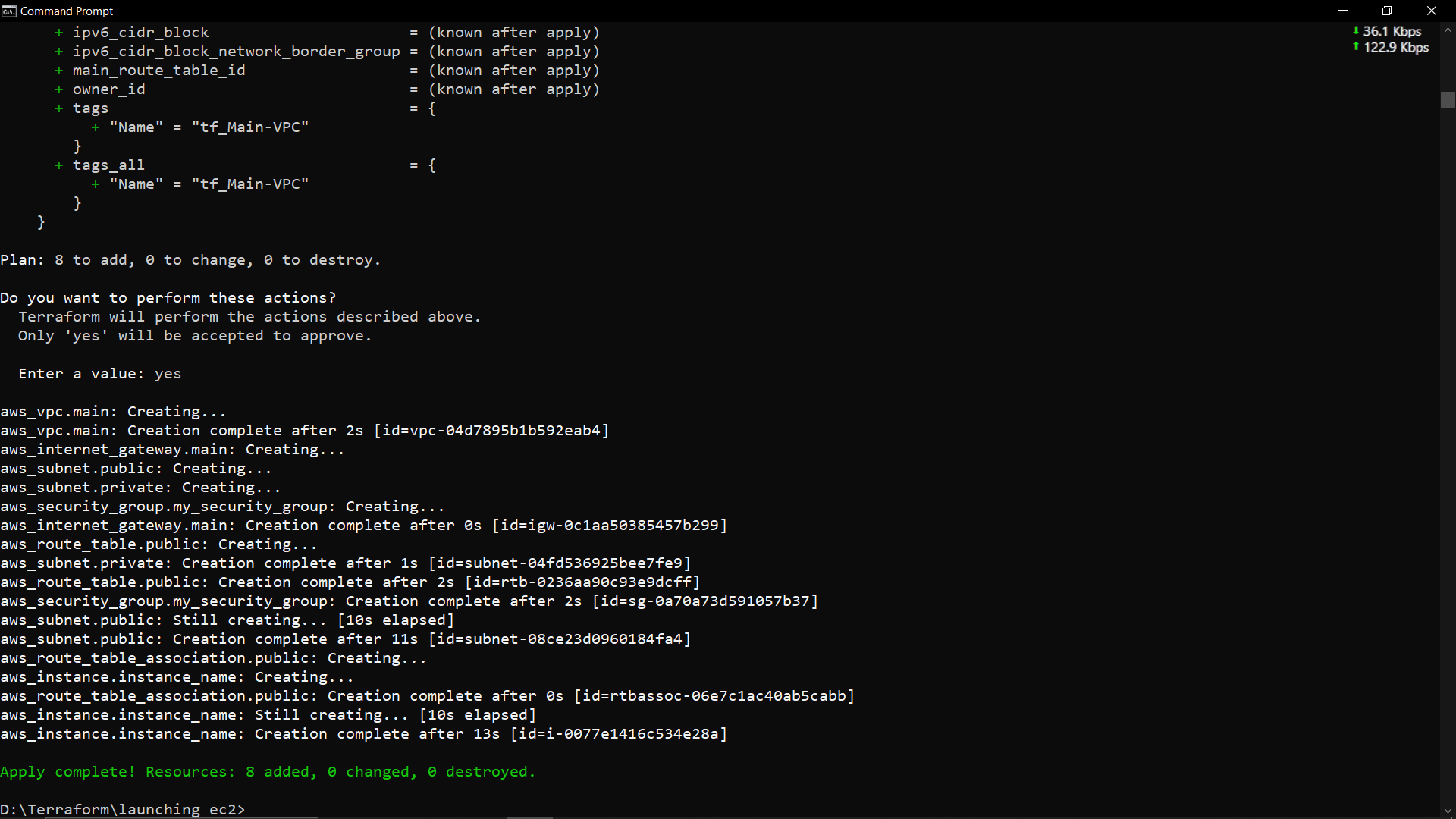


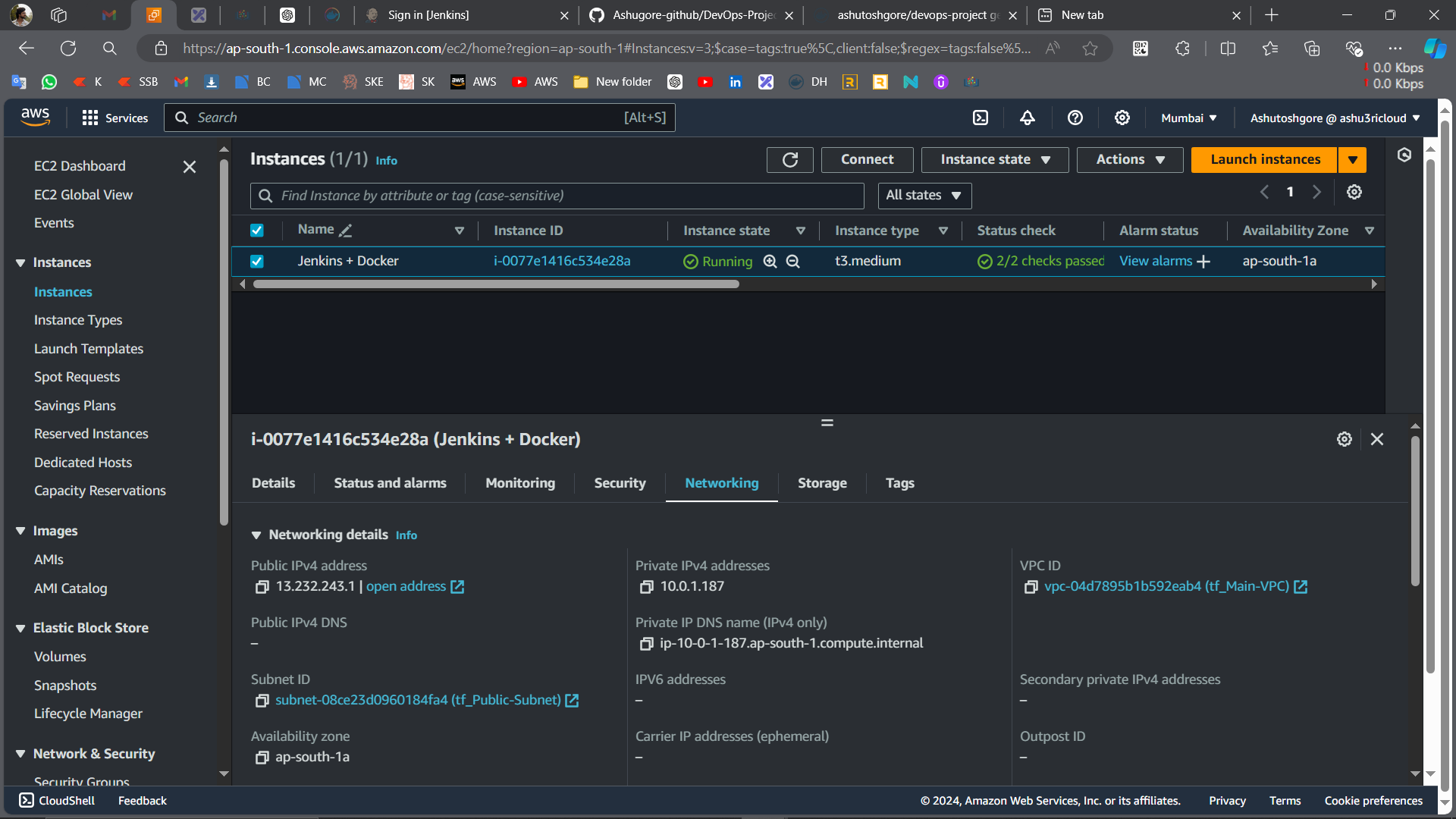




1. **Initialize and Apply Terraform**

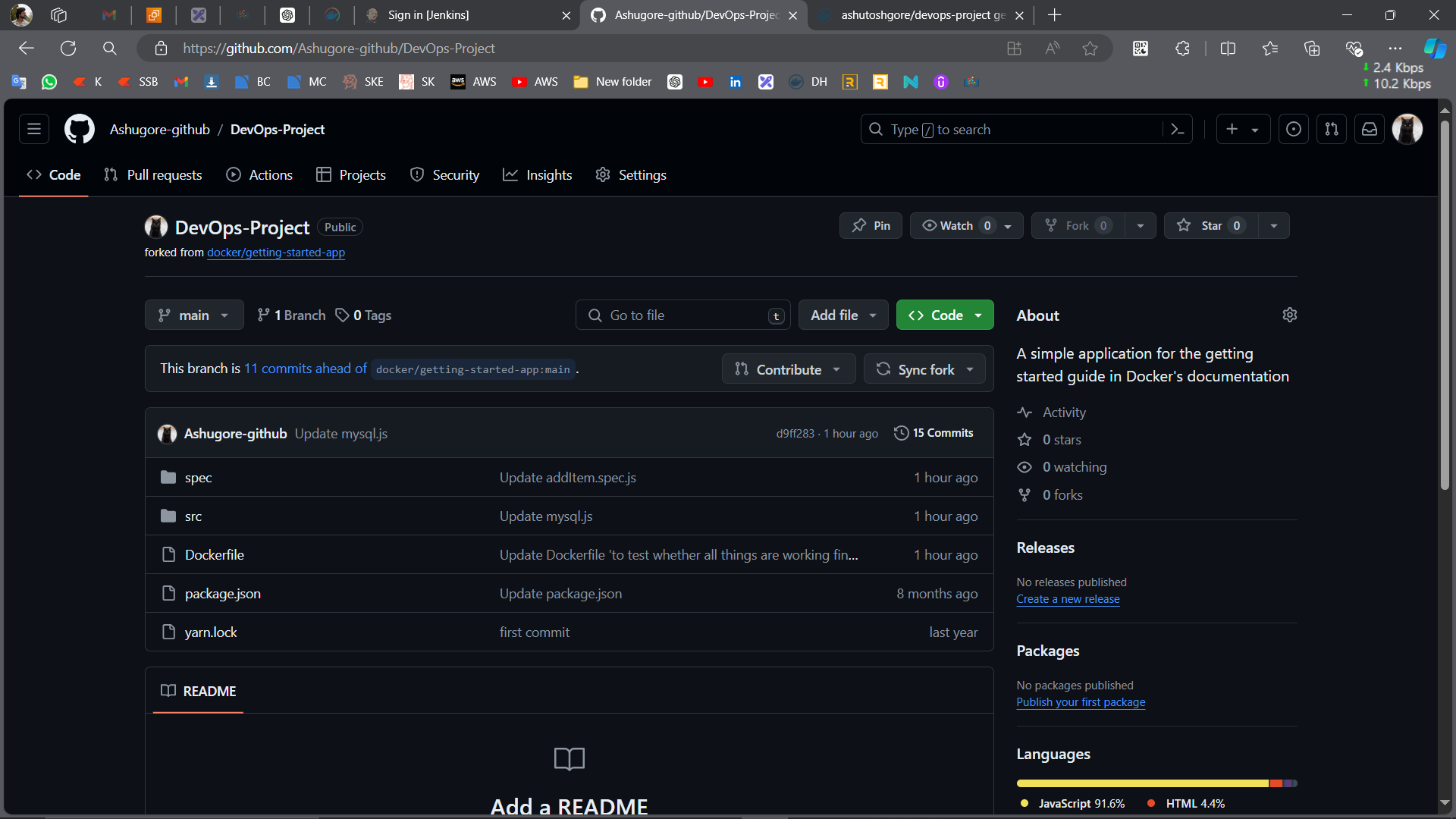
* Open a terminal in CMD
* Run ‘terraform plan’
* Run ‘terraform apply’

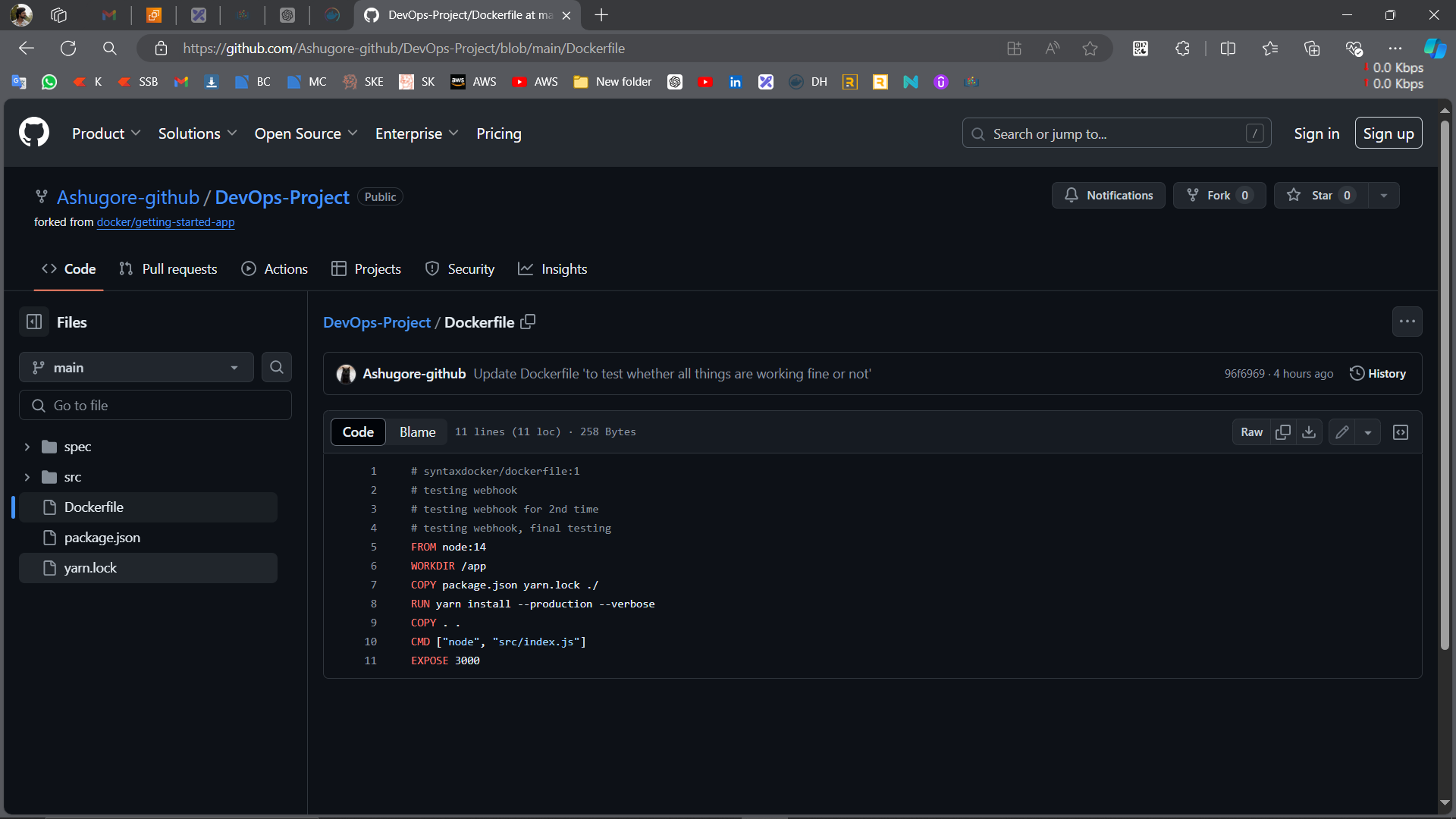




**Step 2: Forking the Application Code from GitHub**

My Github URL: - https://github.com/Ashugore-github/DevOps-Project

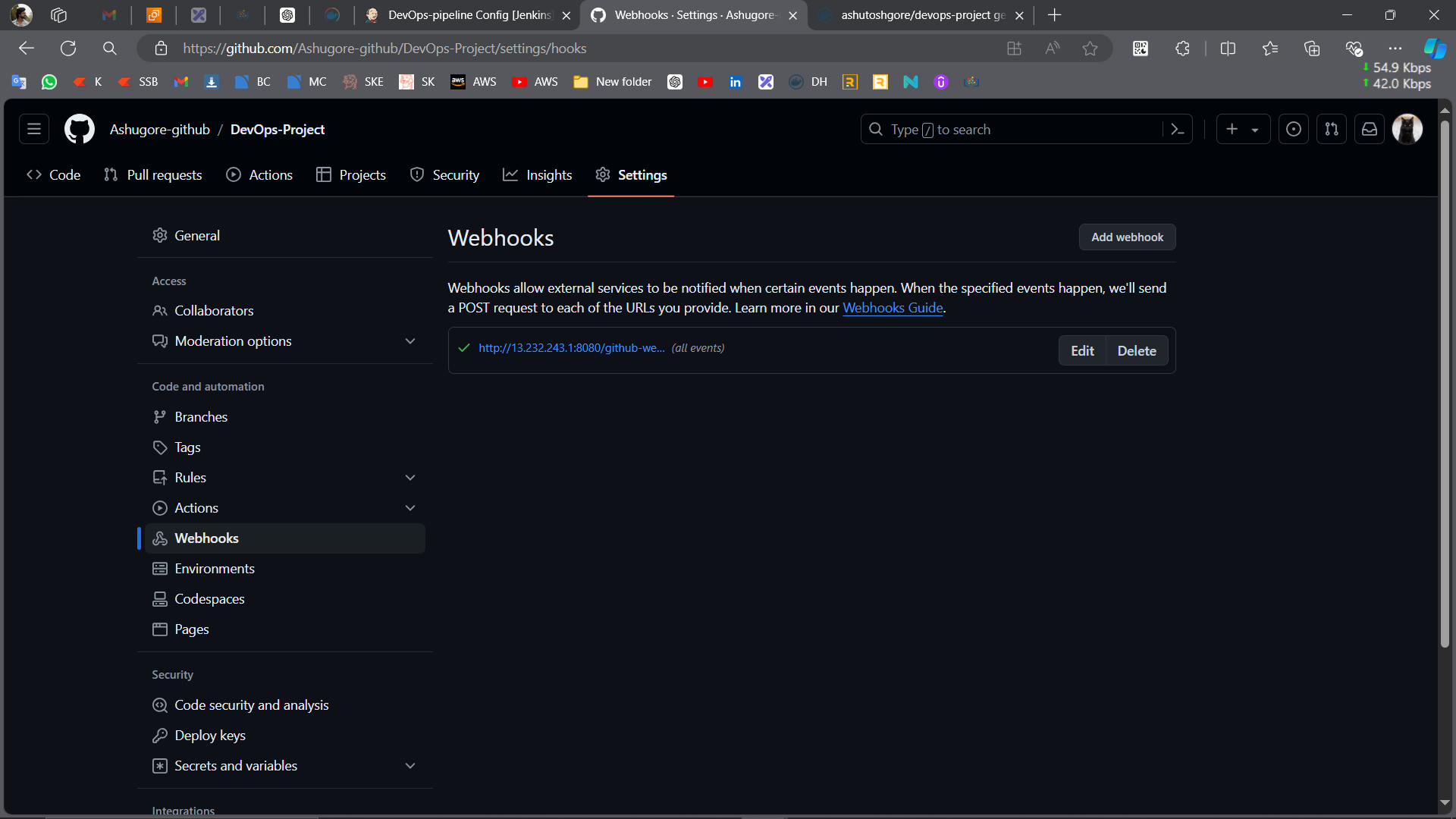
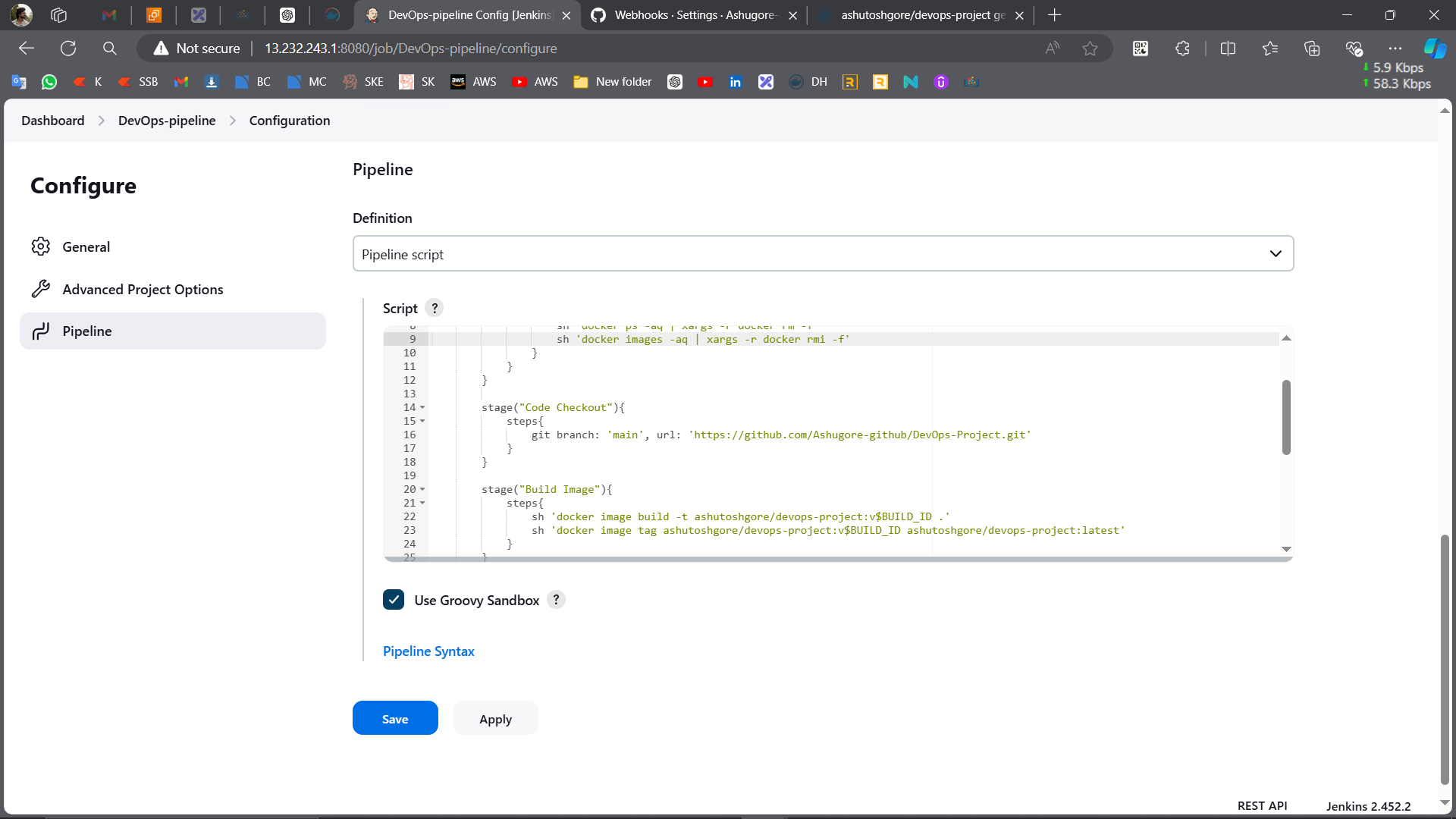
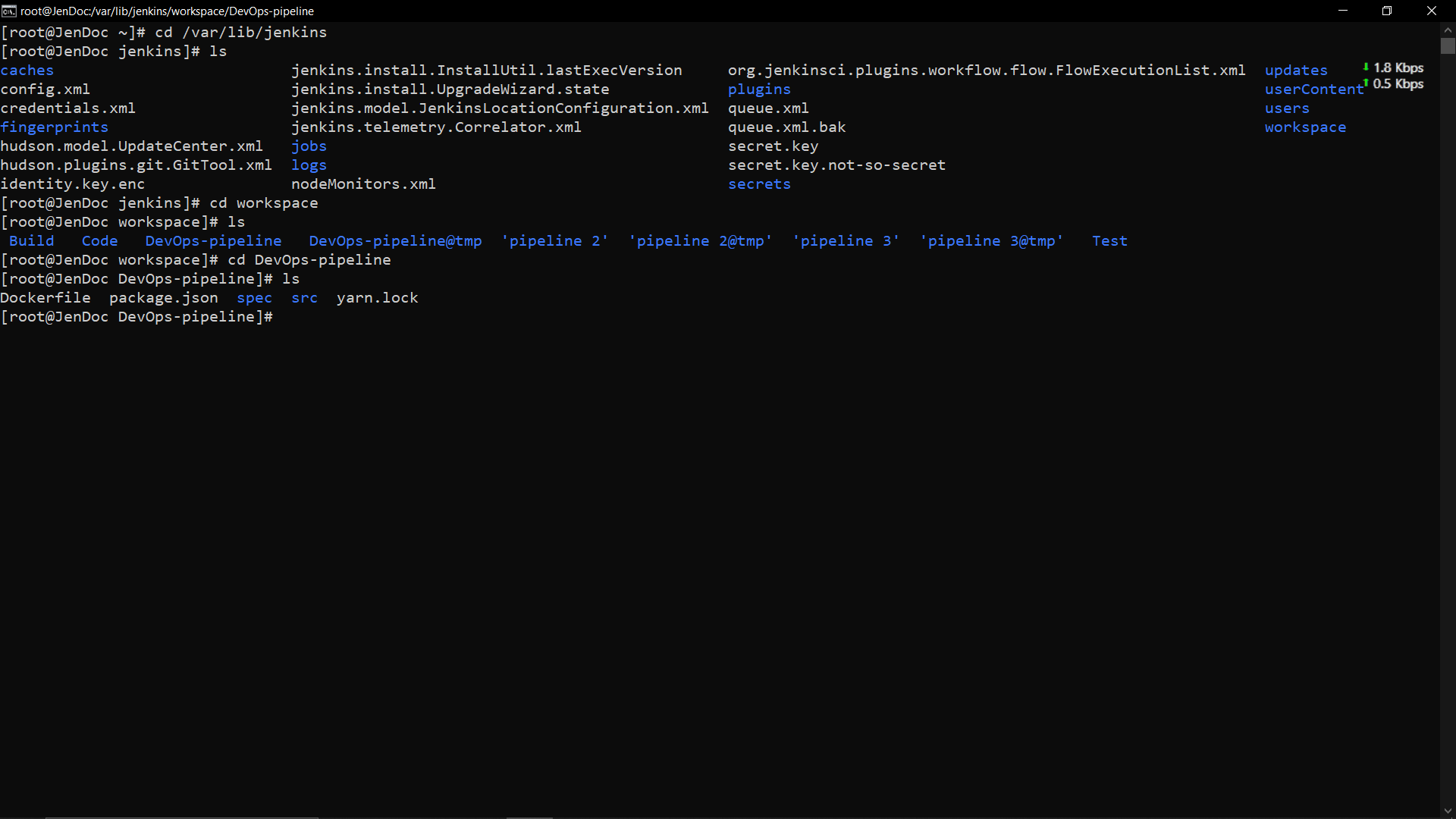


* Create Dockerfile in GitHub.
* 

**Step 3: Installation of Jenkins and Docker on RHEL 9 EC2 Instance**

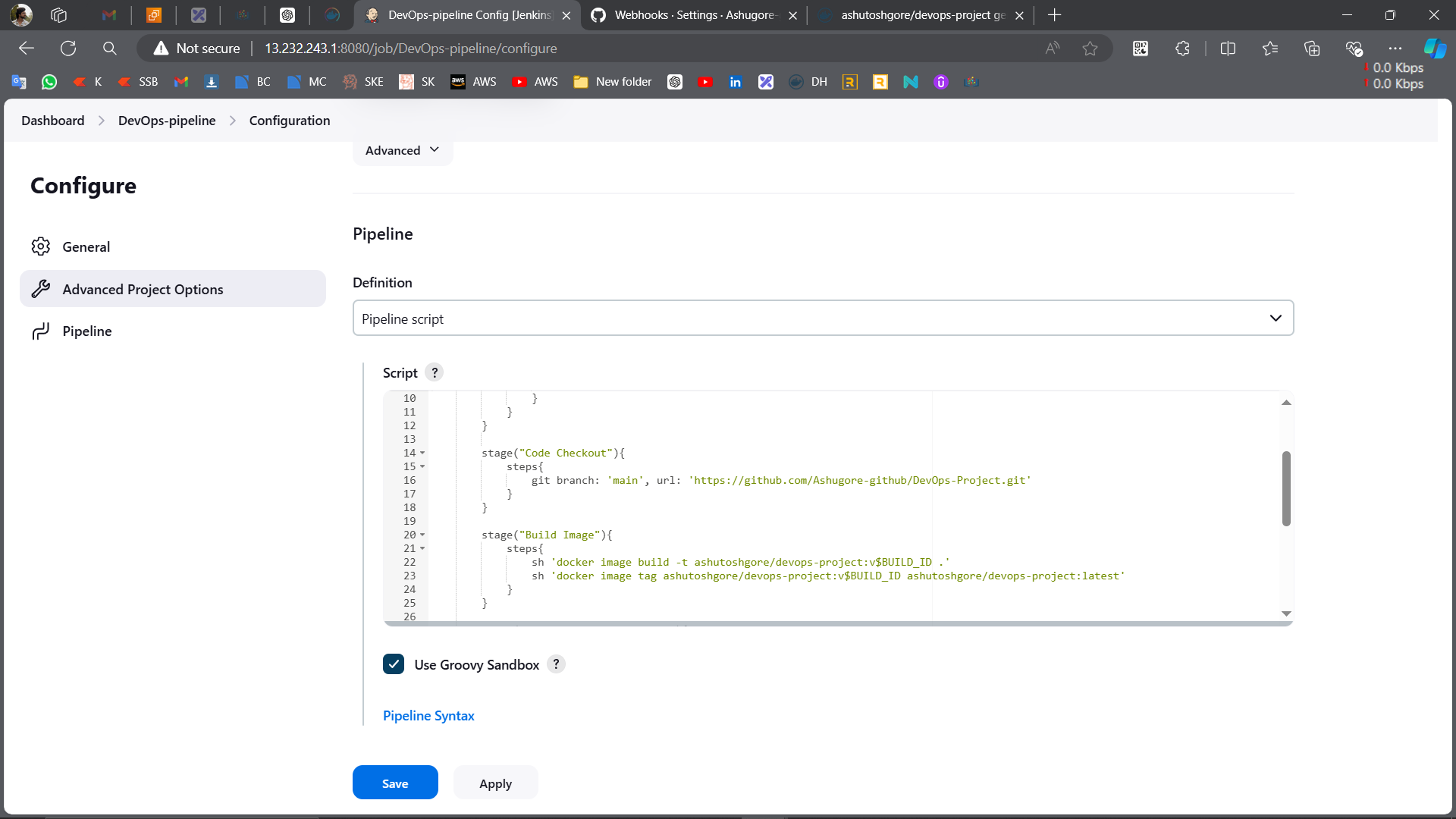
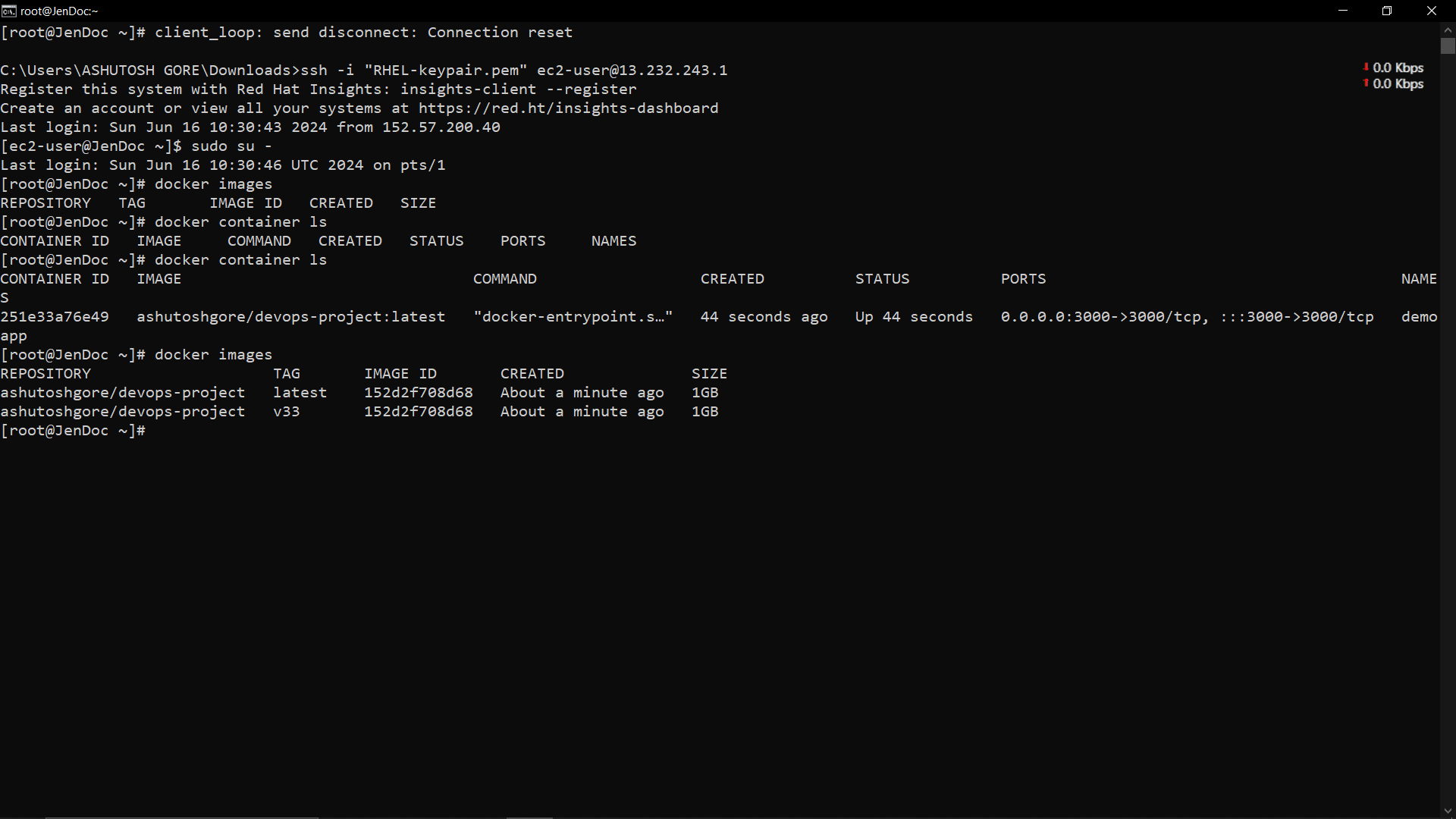
* Install Jenkins and create an account.
* Install docker.

**Step 4: Integrating GitHub and Jenkins**

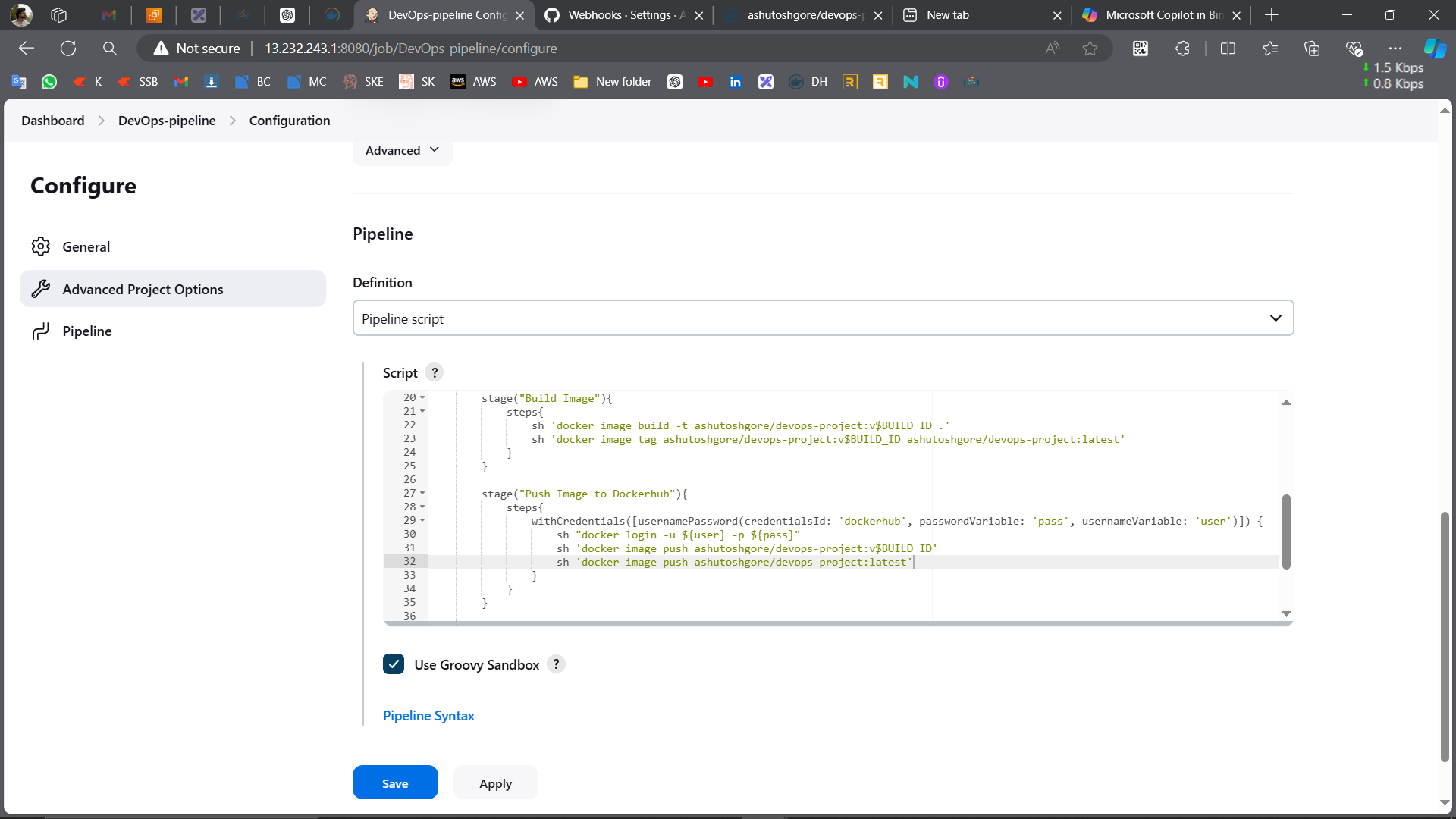
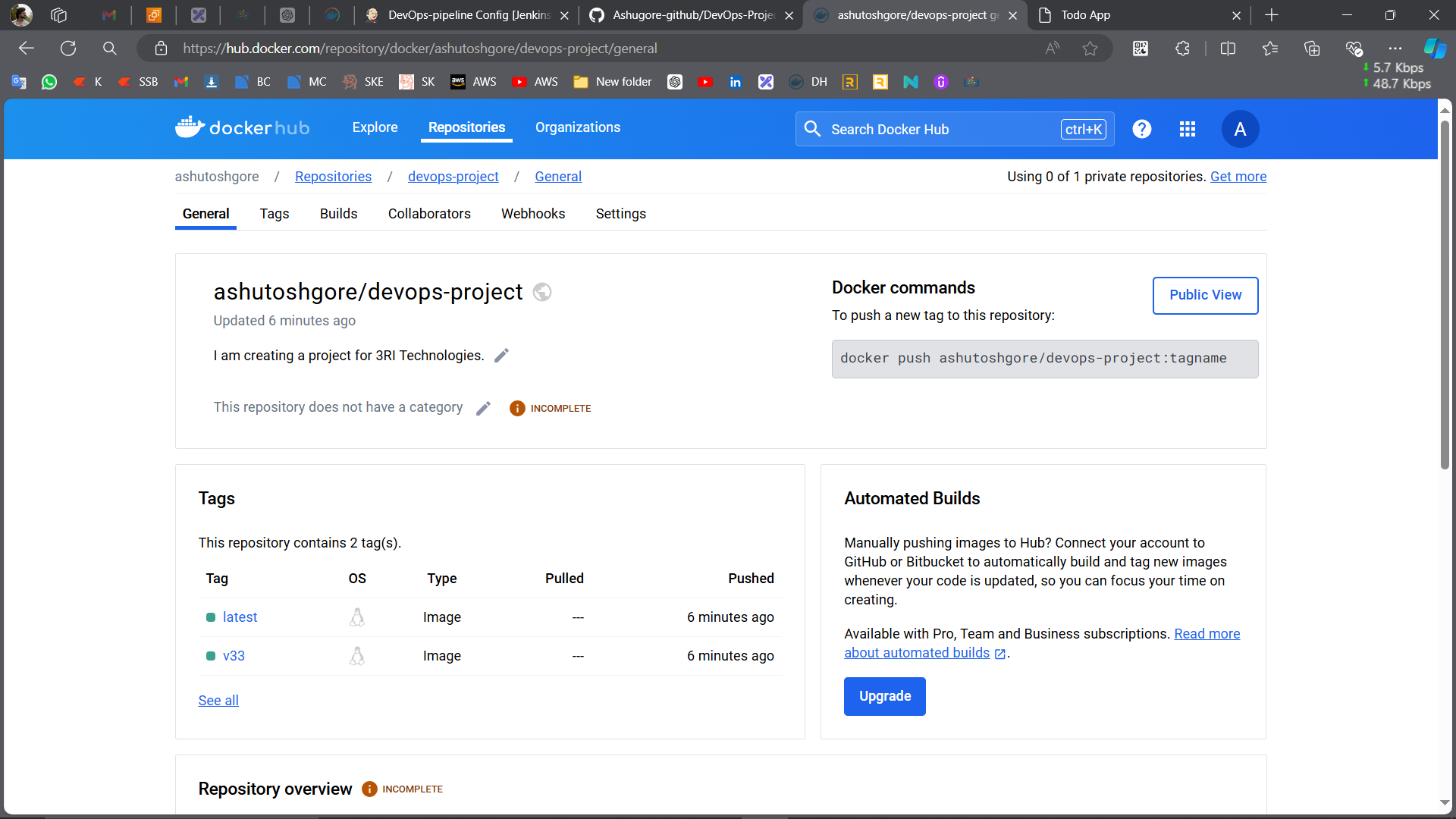
* Creating webhook on the GitHub repository and integrating it with Jenkins.
  + 
* Enable the checkbox of pipeline🡪Build Triggers🡪GitHub hook trigger for GITScm polling
* Create a snippet (from Pipeline Syntax) to generate url.
  + 
* Now we can see the GitHub file in an instance.
  + 

**Step 5: Build an Image and Push it in Docker Hub.**

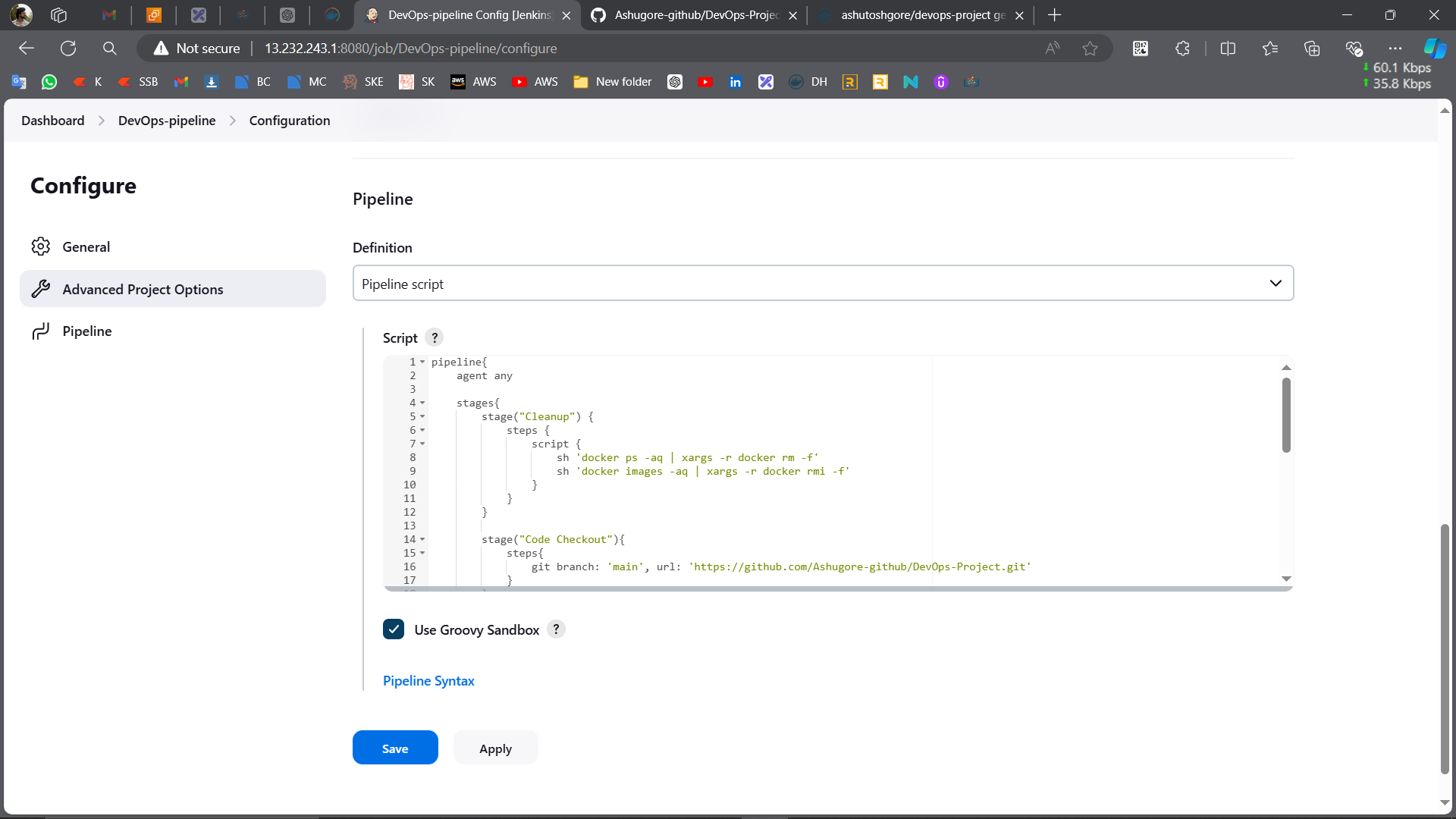
1. **Installing images**

* These commands will 2 images one is with the latest tag and another one is with the build no.
* The $BUILD\_ID is a placeholder for a build identifier, which you would replace with an actual value during the build process.
* 
* 

1. **Pushing image to Docker Hub**.

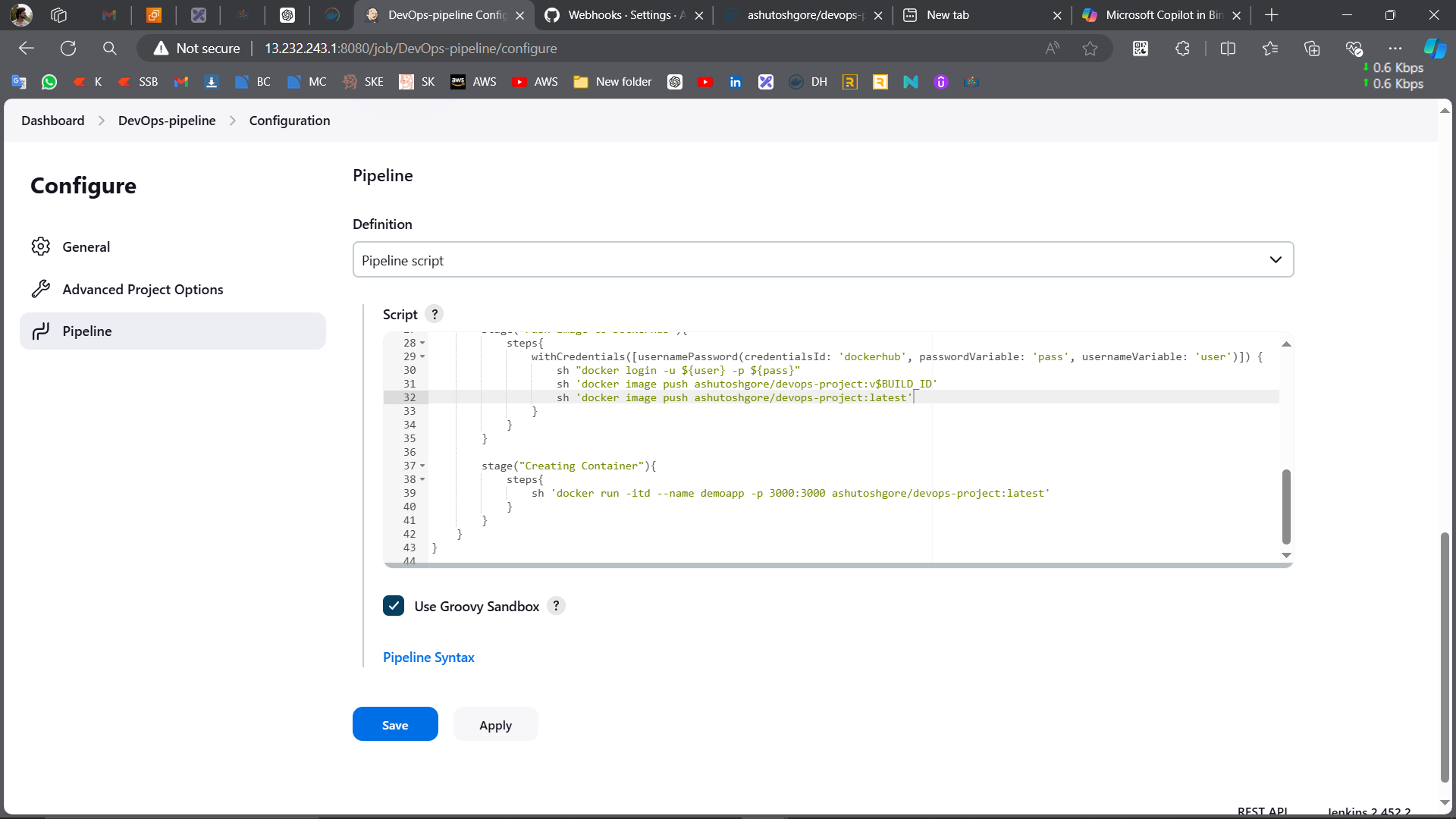
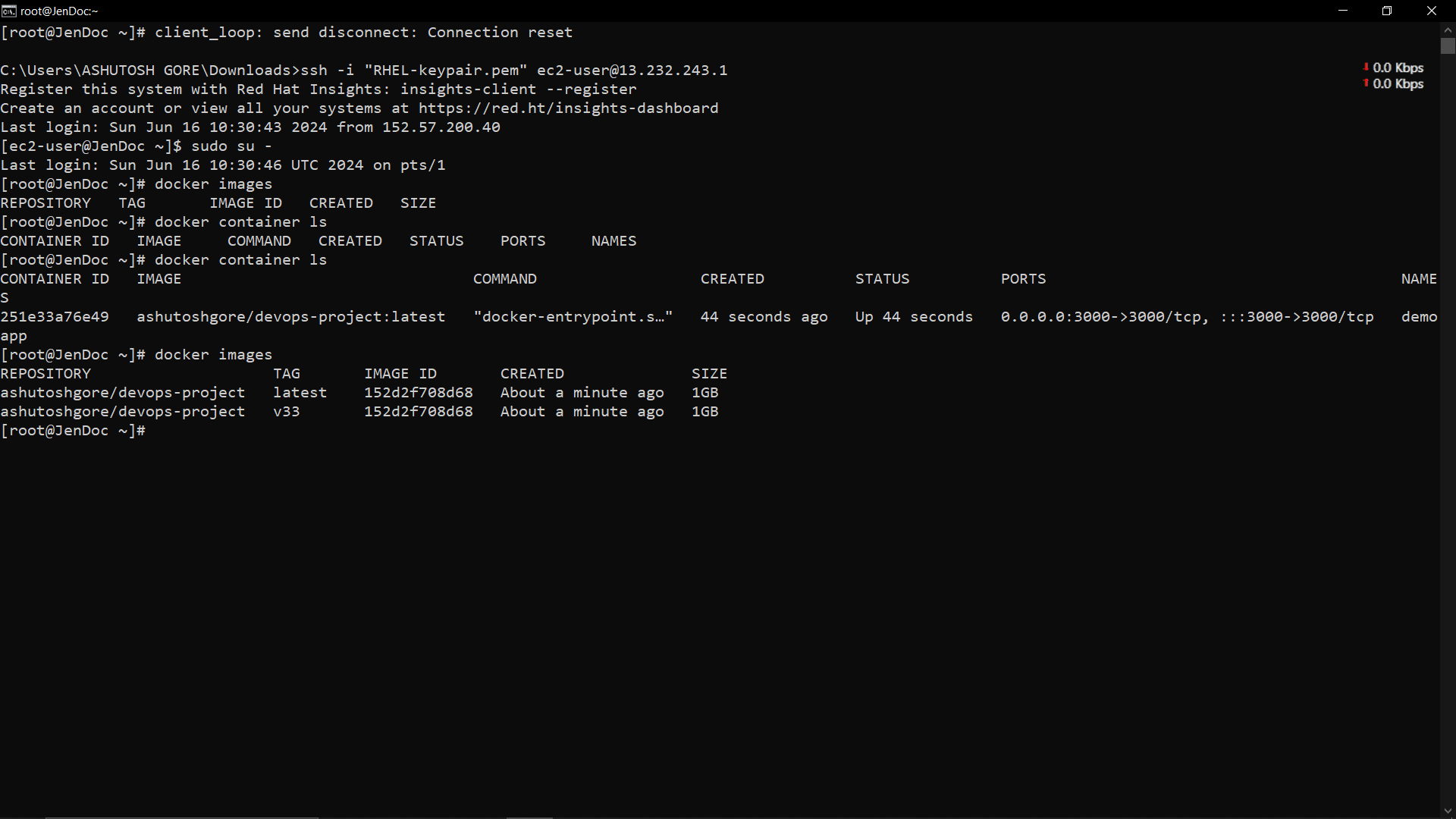
* First create credentials
* Secondly create a snippet to generate url.
* And now write commands to push images.
* 
* 

1. **Clean up**

* To delete preinstall images and docker containers.
* 

**Step 6: Create a Container and Access the application**

**1. Create a Container:**

* I give the ‘demoapp’ name to the container and the image will be ‘latest’ and exposed on 3000:3000 port.
* 
* 

**2. Access the application:**

* Copy the instance IP and with 3000 and run it on browser. <instanceIP:containerport>.
* 