Seneca logo


Lab 2: Linux refresher and automation with bash scripts

The goal of this Lab is to familiarize ourselves with the main Linux commands and tools that we will use.

We will cover the following:

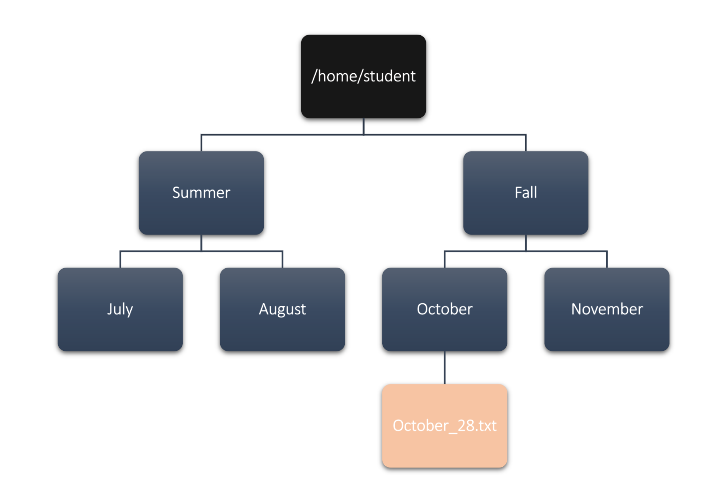
1. Create a folder structure using basic Linux commands
2. Working with package manager on CentOS based VM
3. Connecting to Amazon EC2 from your local environment using custom SSH key.
4. Create a service on EC2 instance
5. Linux Networking – Troubleshooting Webserver connectivity
6. Securely connect to Amazon EC2 with AWS SSM Session Manager

Submission requirements:

* Submit screenshots with functionality of tasks 2, 3, 4, 5.
* Submit screenshot that verifies your identity

Task 1: Create a folder structure using basic Linux commands

In your CLoud9 environment, create the basic folder structure below. Assume that your home directory is /home/student.



mkdir -p Fall/October

mkdir -p Fall/November

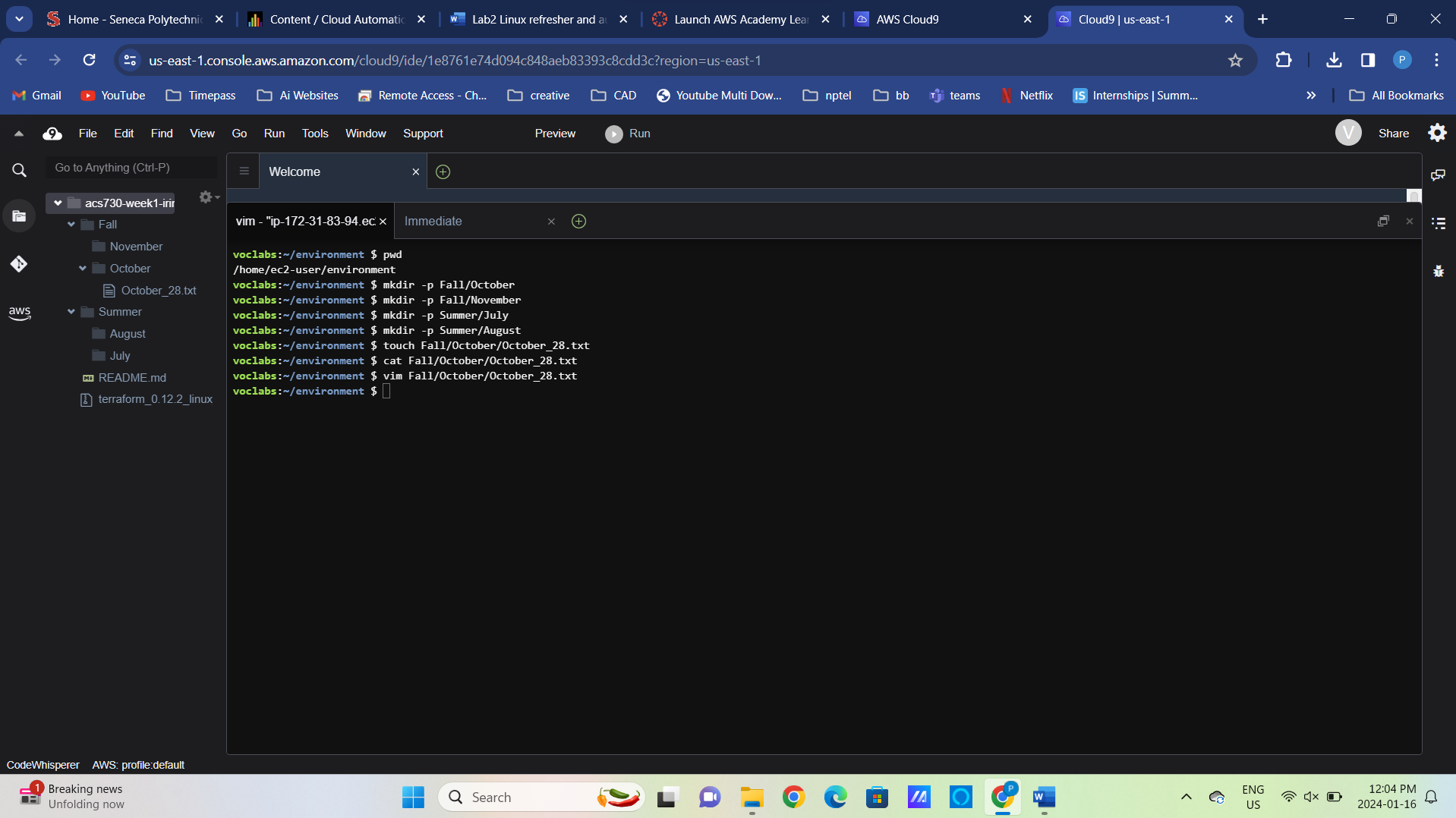
mkdir -p Summer/July

mkdir -p Summer/August

touch Fall/October/October\_28.txt

cat Fall/October/October\_28.txt

vim Fall/October/October\_28.txt



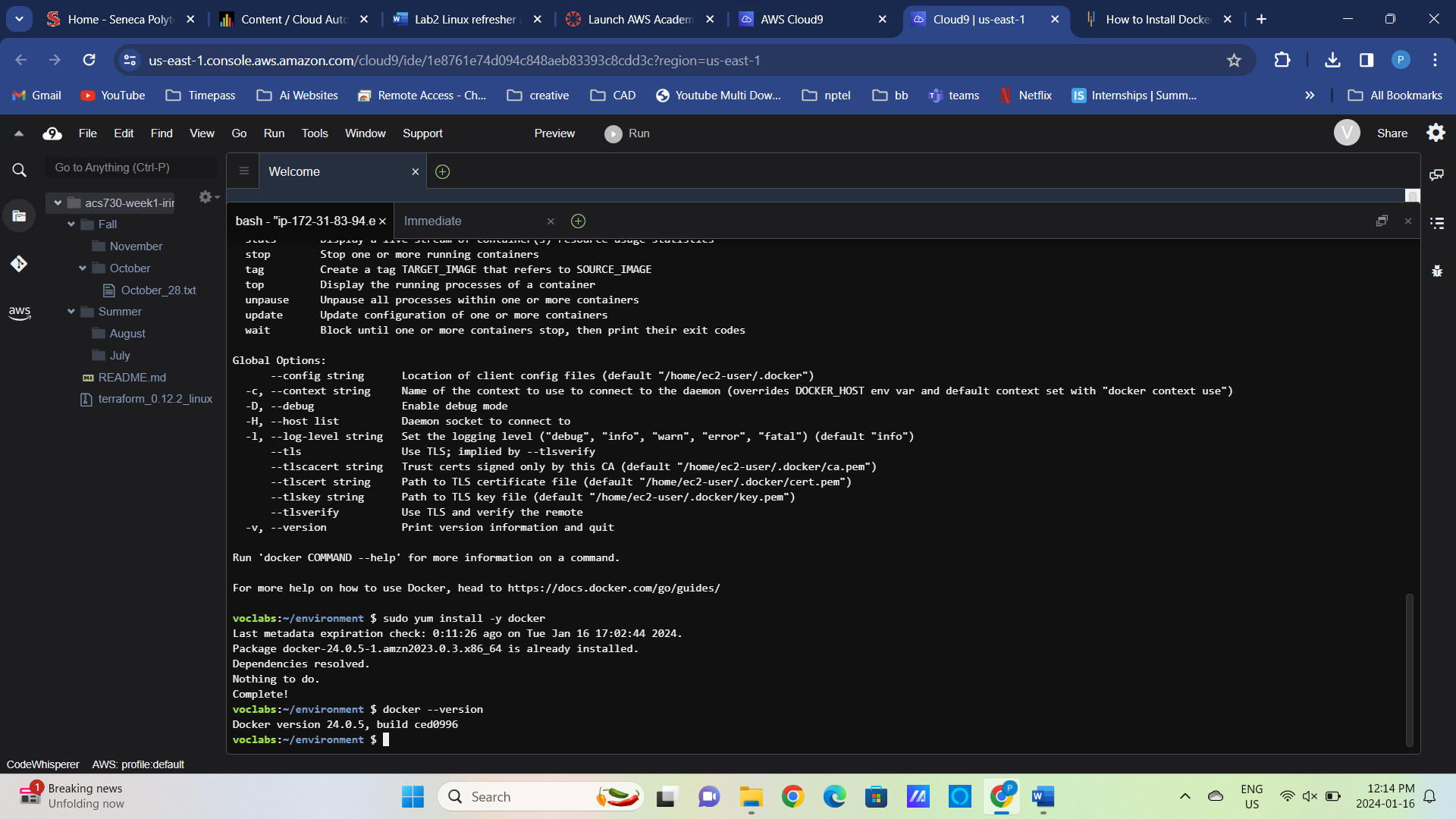
Task 2: Working with the Linux package manager

In your Cloud9 environment, install the latest docker package and make sure the docker service is running when the VM is restarted.  
<https://docs.aws.amazon.com/cloud9/latest/user-guide/sample-docker.html#sample-docker-install>

Install Docker from the above given link.  
  
sudo yum install -y docker

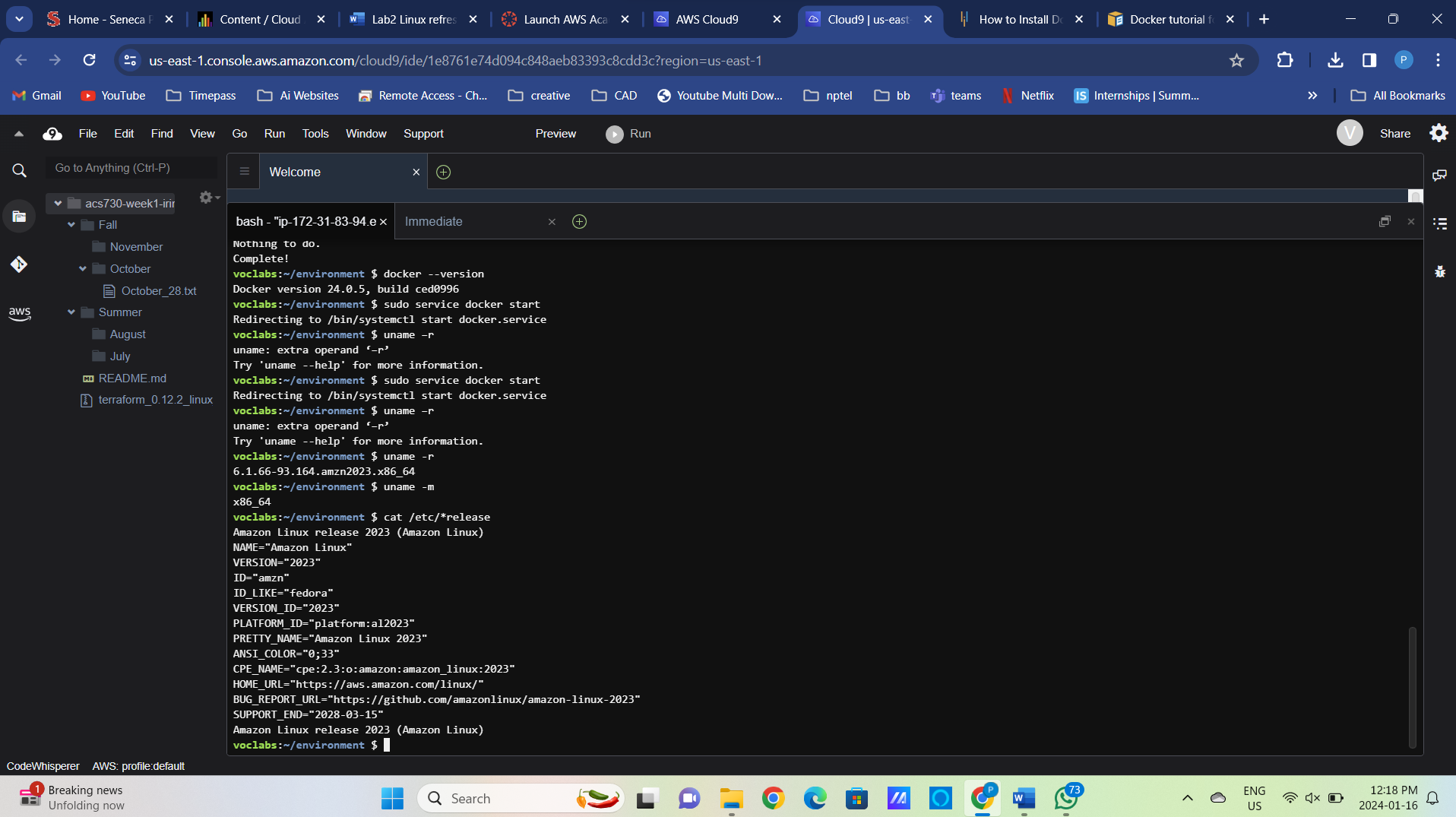
docker –version

sudo service docker start



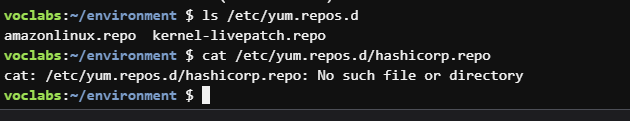
1. Find out if this is a Debian or an RMP distribution. What is the version of this distribution? What chip manufacturer and architecture?

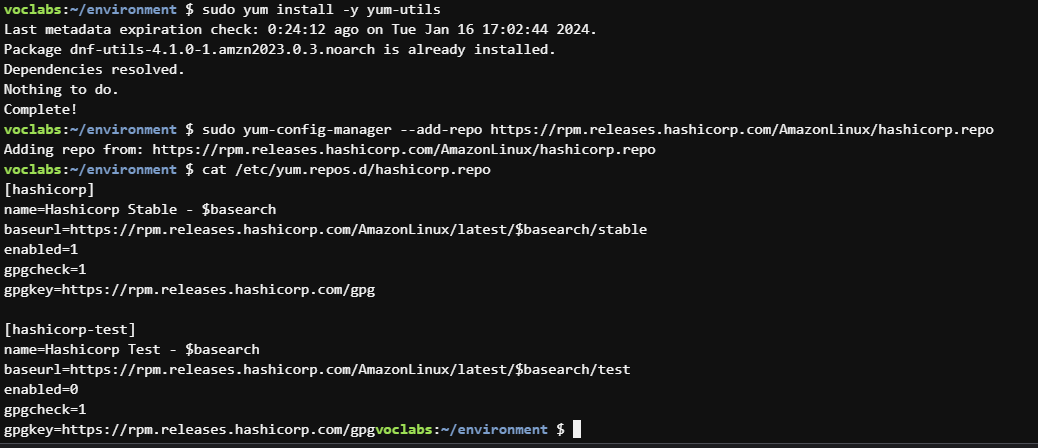
|  |  |
| --- | --- |
| Version | uname –r |
| Manufacturer | uname –m |
| Distro | cat /etc/\*release |

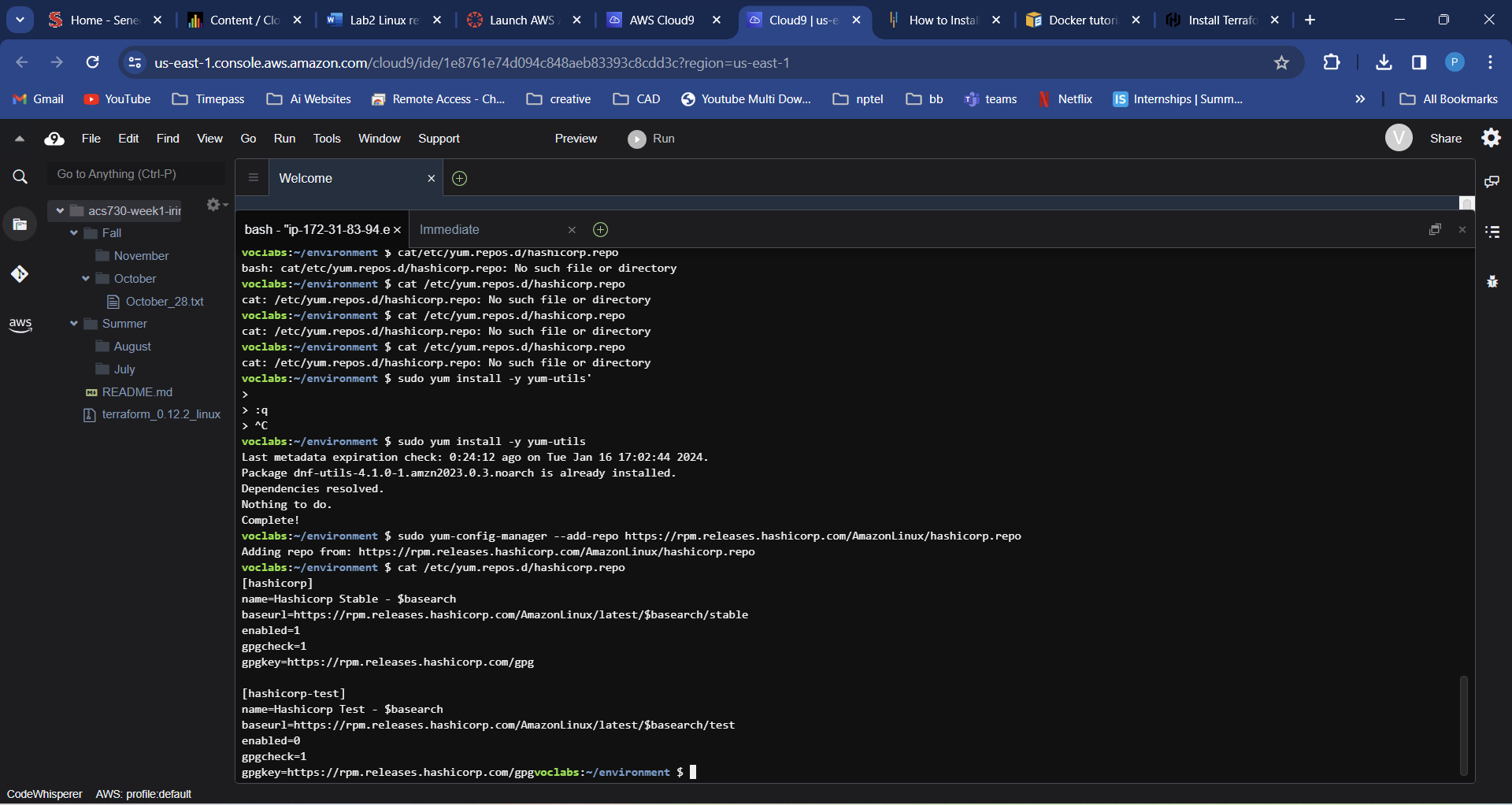


1. Find all the repos available on Cloud9 VM

|  |  |
| --- | --- |
| All repos | ls /etc/yum.repos.d  sudo yum install -y yum-utils  sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo |
| Print the details of the specific repo | cat /etc/yum.repos.d/hashicorp.repo |

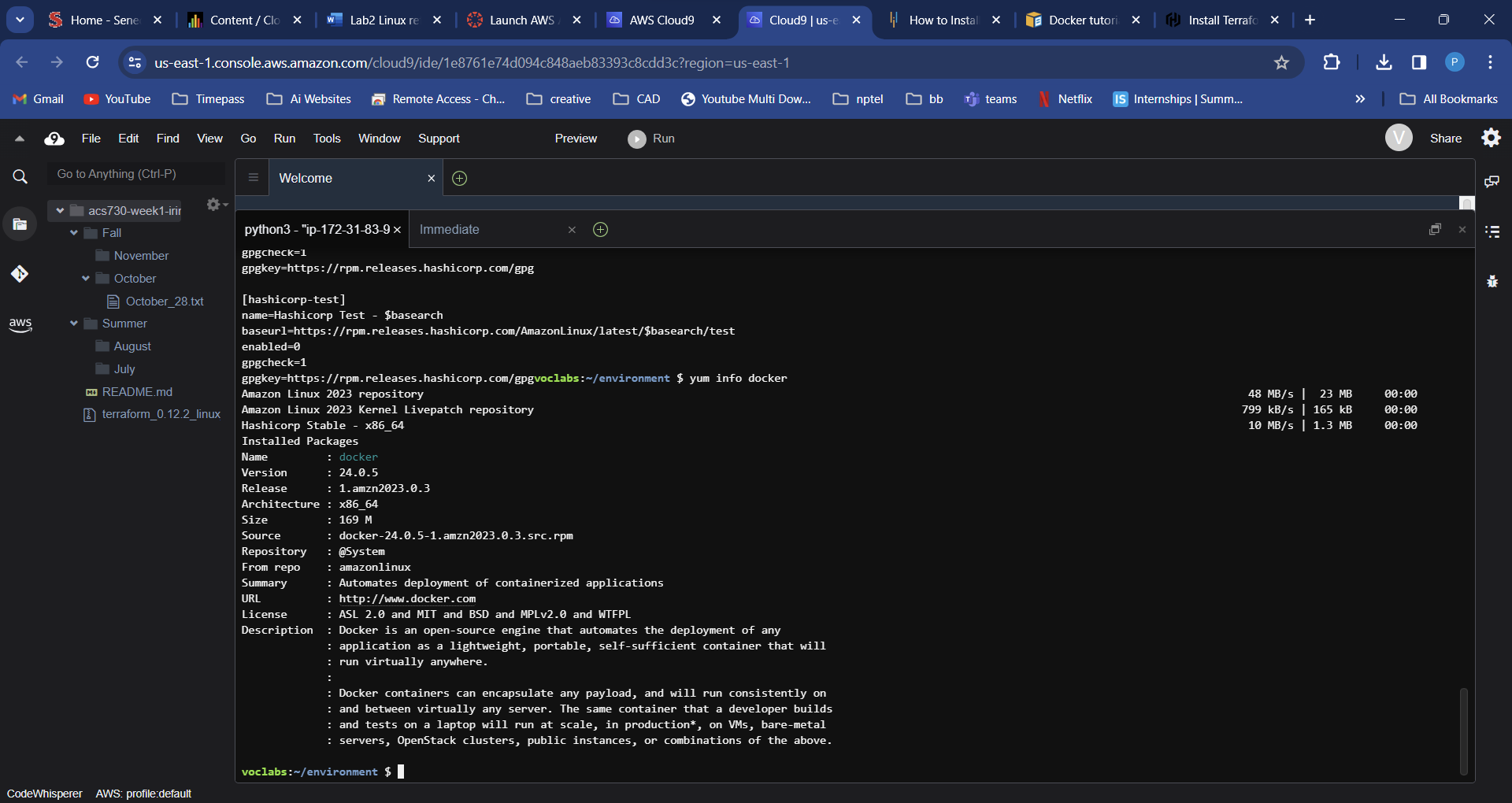






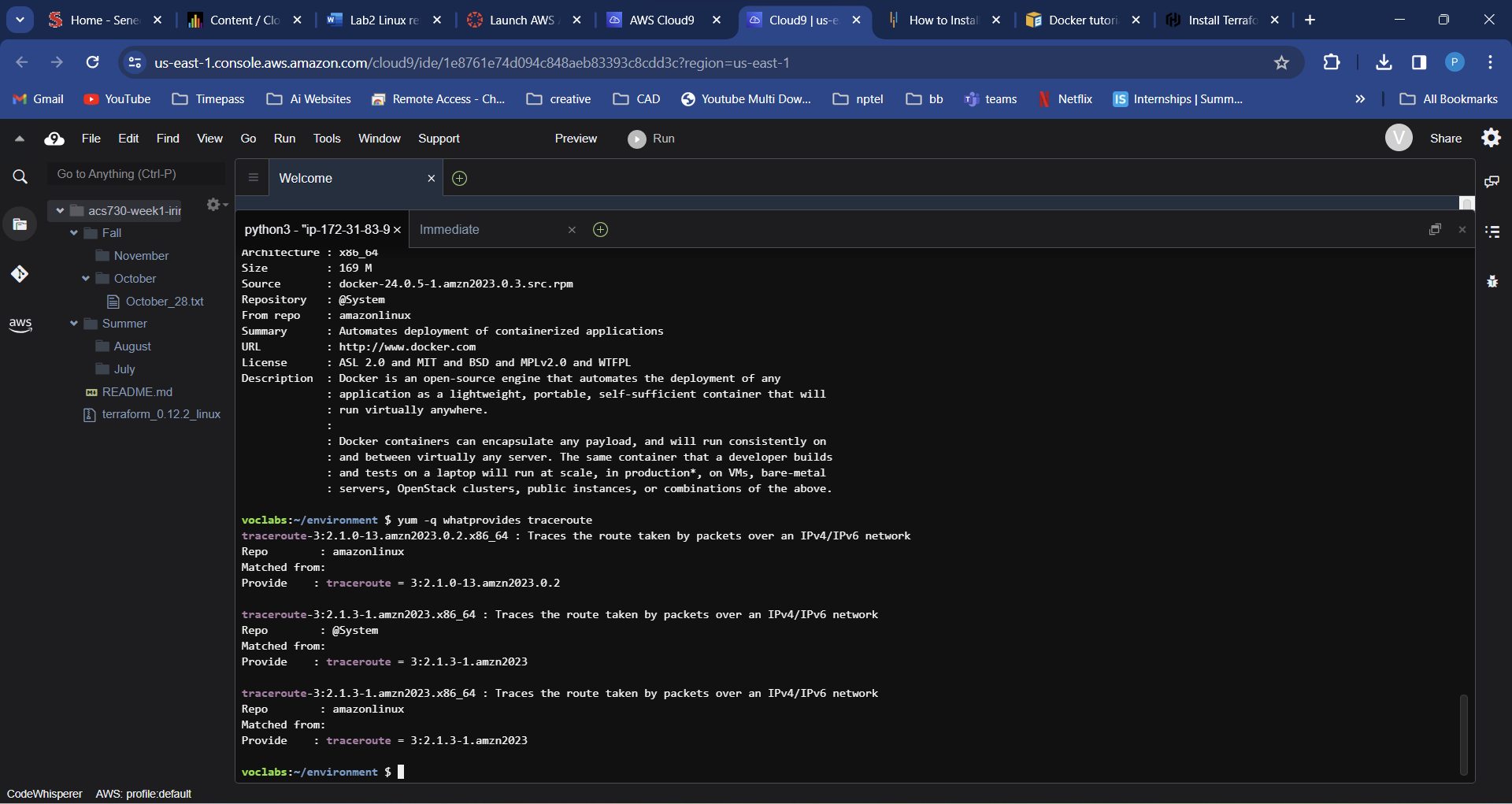
1. What version of docker package is installed on Cloud9 VM?

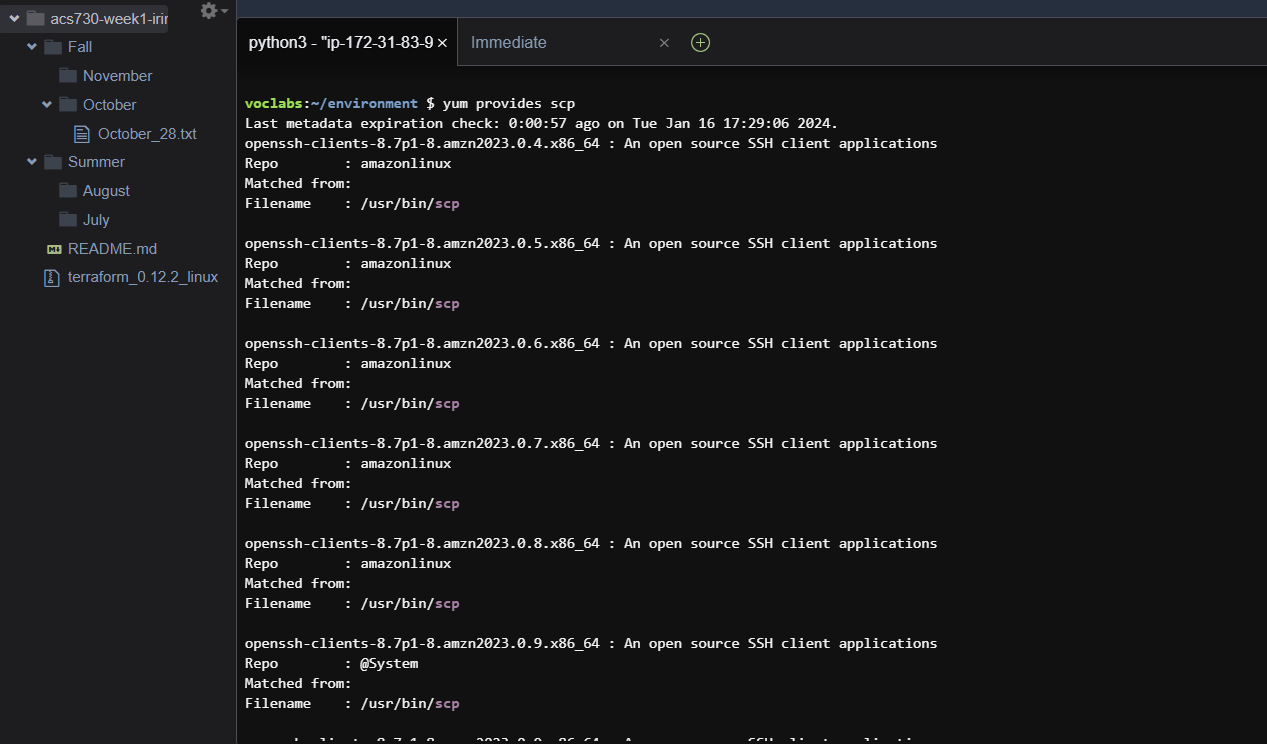
|  |  |
| --- | --- |
| Using YUM | yum info docker |



1. Identify the package that provides traceroute command

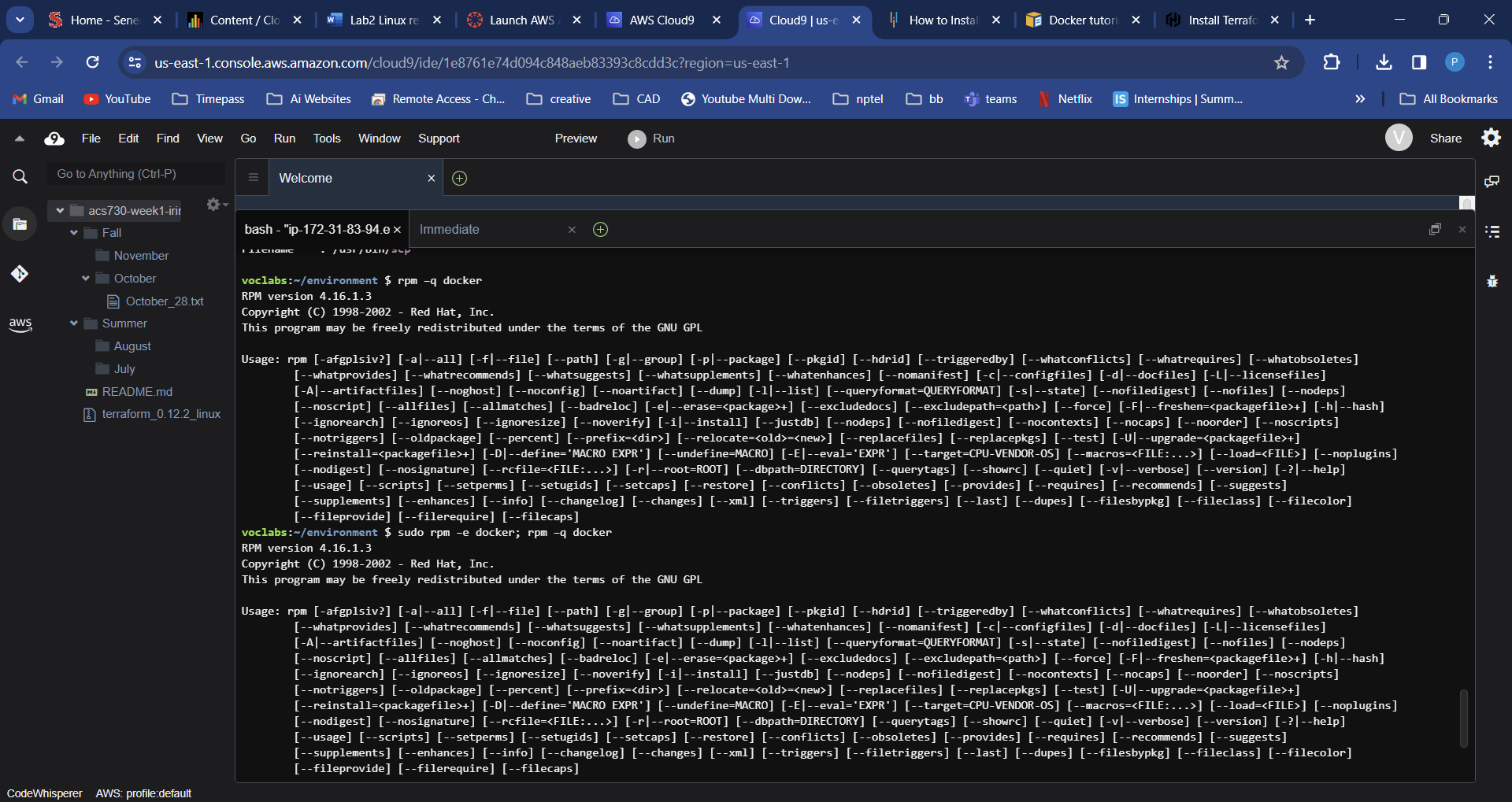
|  |  |
| --- | --- |
| Using YUM | yum -q whatprovides traceroute  yum provides scp |

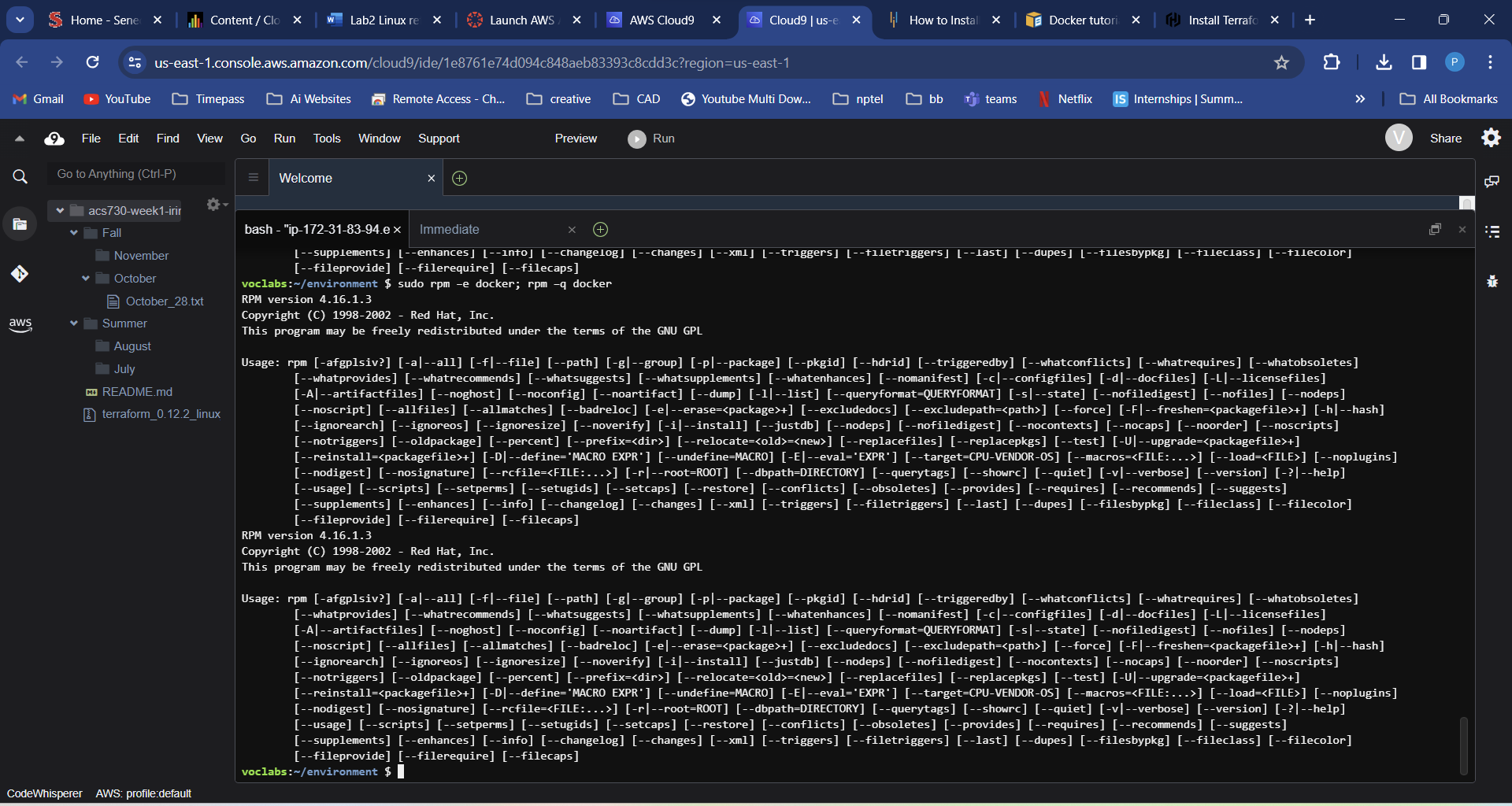


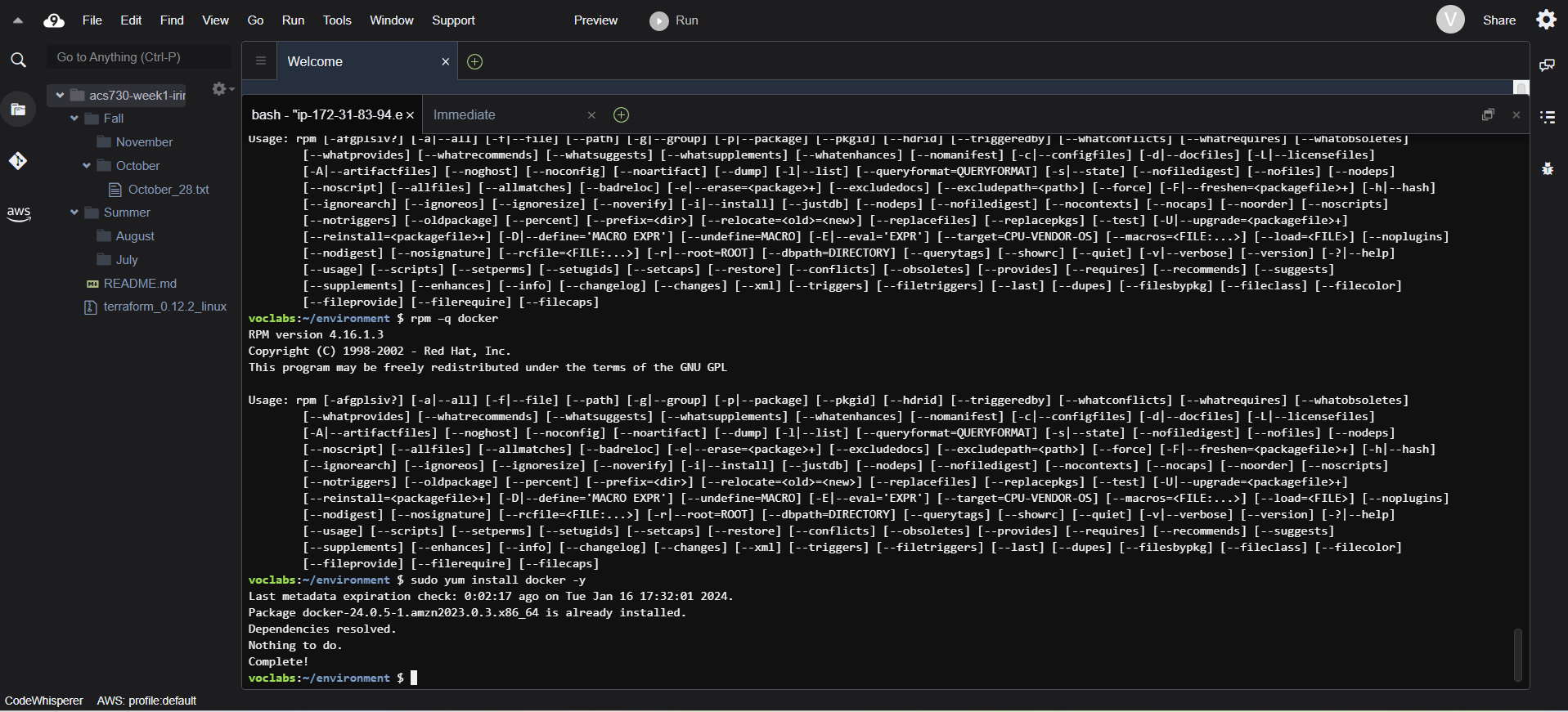


1. Is the docker package installed? If yes, uninstall the existing docker package and re-install it

|  |  |
| --- | --- |
| Is a docker installed? | rpm –q docker |
| Uninstall docker | sudo rpm –e docker; rpm –q docker |
| sudo yum -y remove docker |  |
| Re-install docker | sudo yum install docker -y |

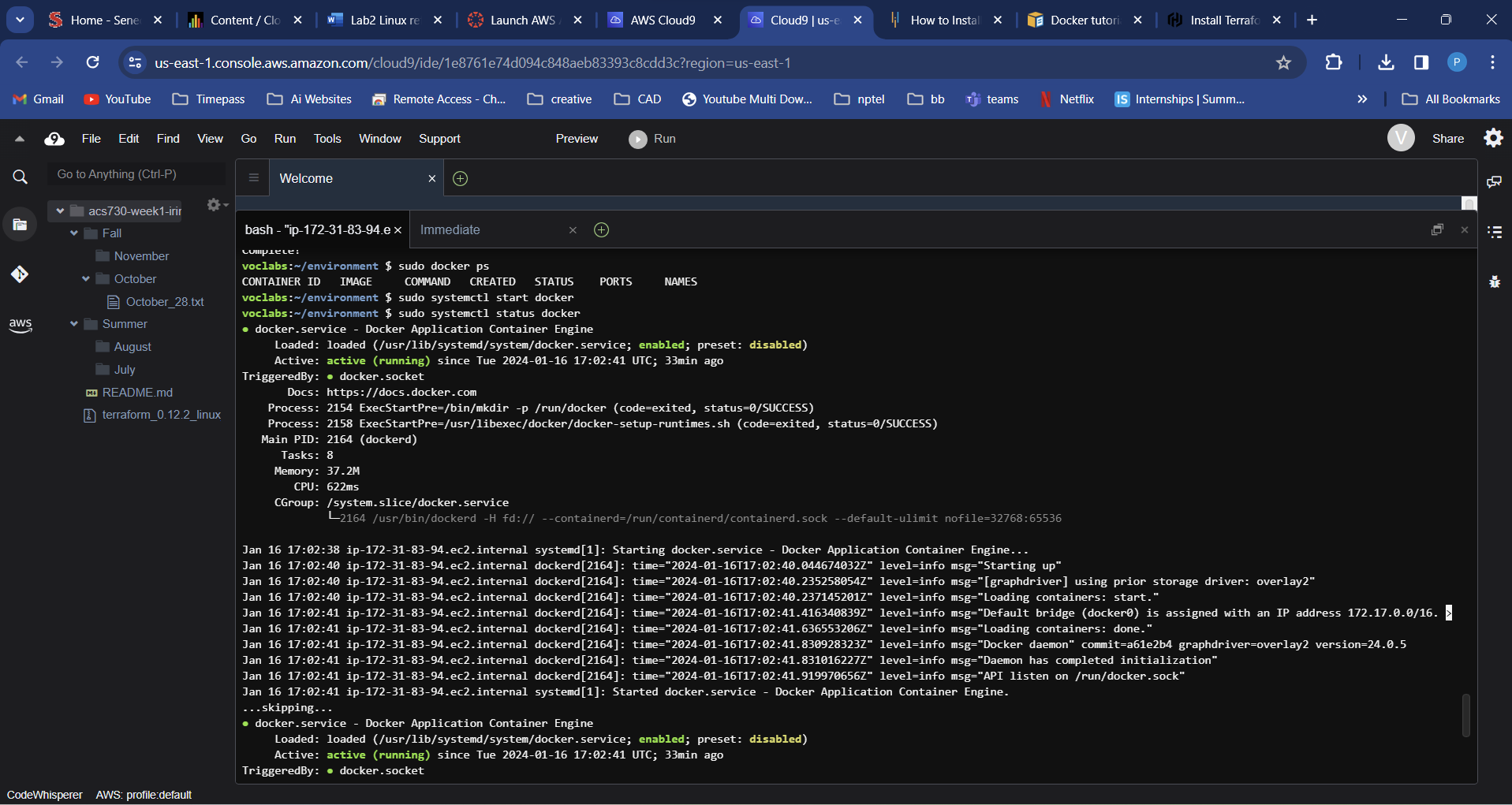


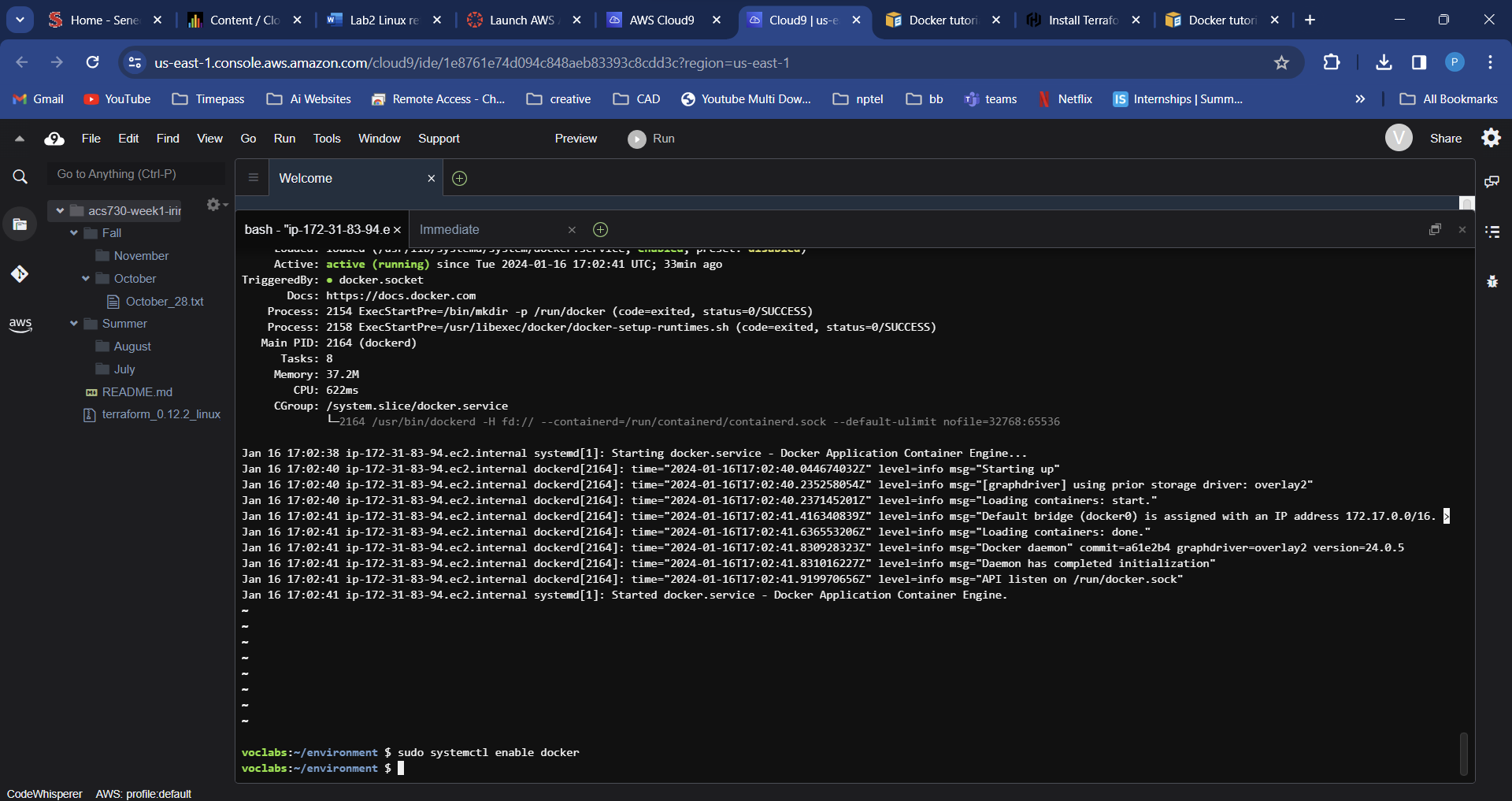




1. Try running docker commands. Does it work? Use systemctl commands to start and enable docker daemon

|  |  |
| --- | --- |
| Is docker running? | sudo docker ps |
| Start docker | sudo systemctl start docker |
| Verify docker is running | sudo systemctl status docker |
| Make sure docker is running after reboot | sudo systemctl enable docker |

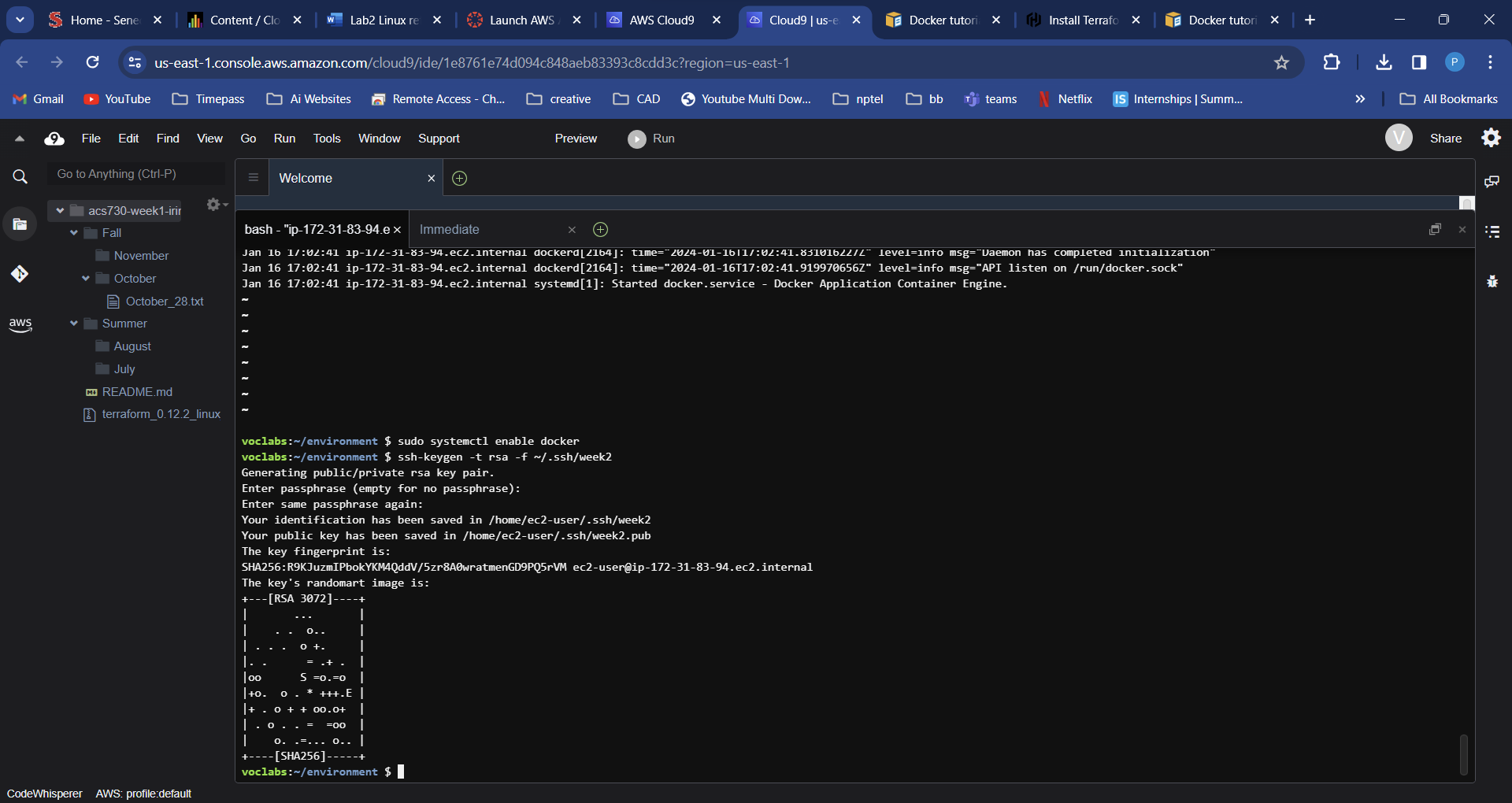


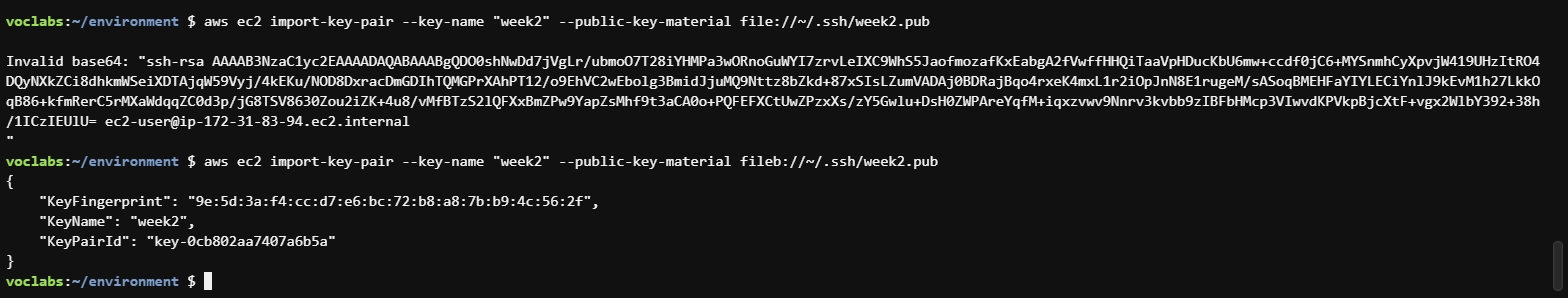


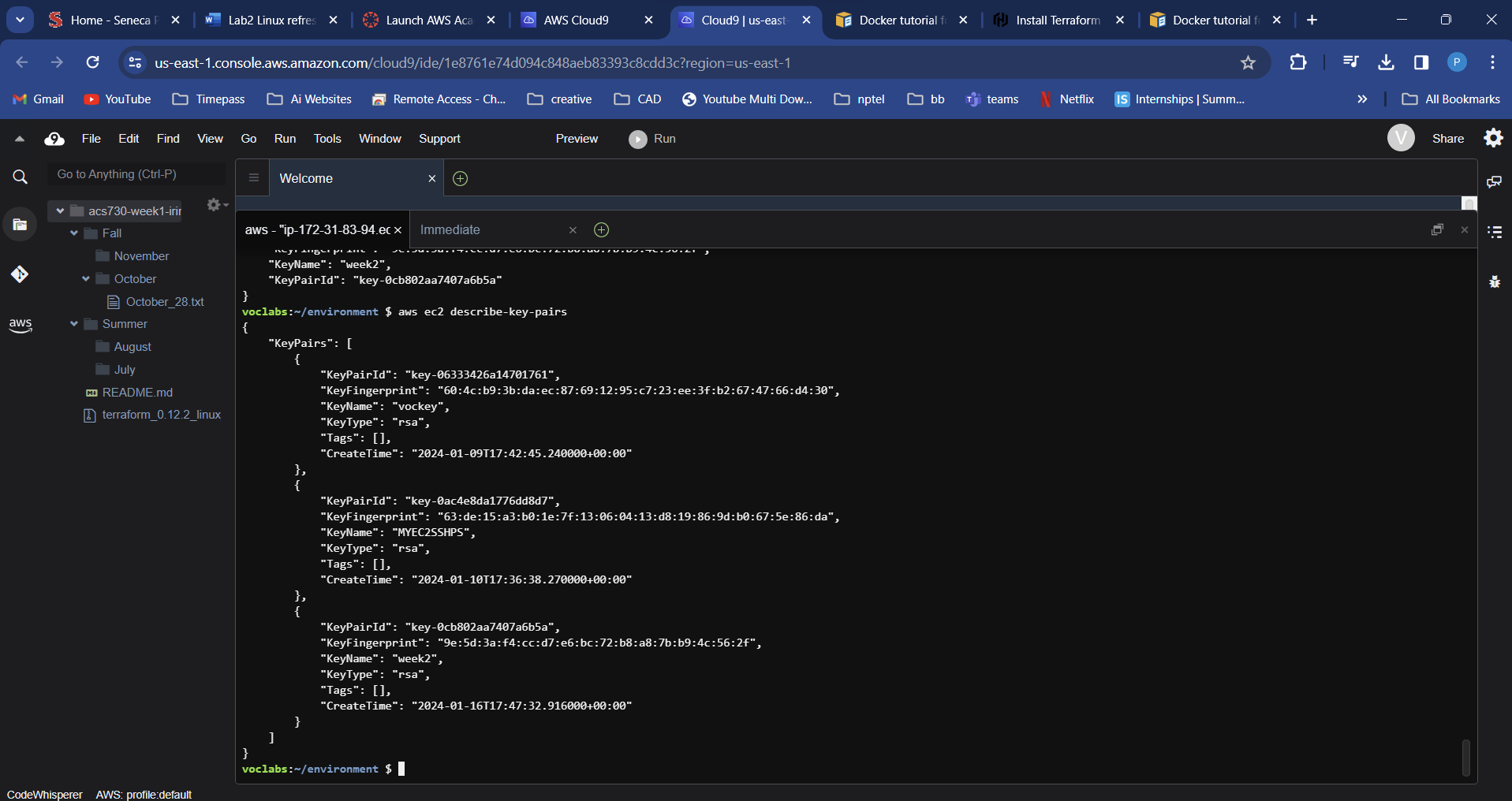
Task 3: Connecting to Amazon EC2 from Cloud9 environment using custom SSH key

1. Create a new SSH key pair in your Cloud9 environment with commands below

|  |  |
| --- | --- |
| Create SSH key pair locally, press Enter when asked for the pass phrase | ssh-keygen -t rsa -f ~/.ssh/week2 |
| Create AWS Key Pair from your local key pair | aws ec2 import-key-pair --key-name "week2" --public-key-material file://~/.ssh/week2.pub |
| Verify the key was created | aws ec2 describe-key-pairs |



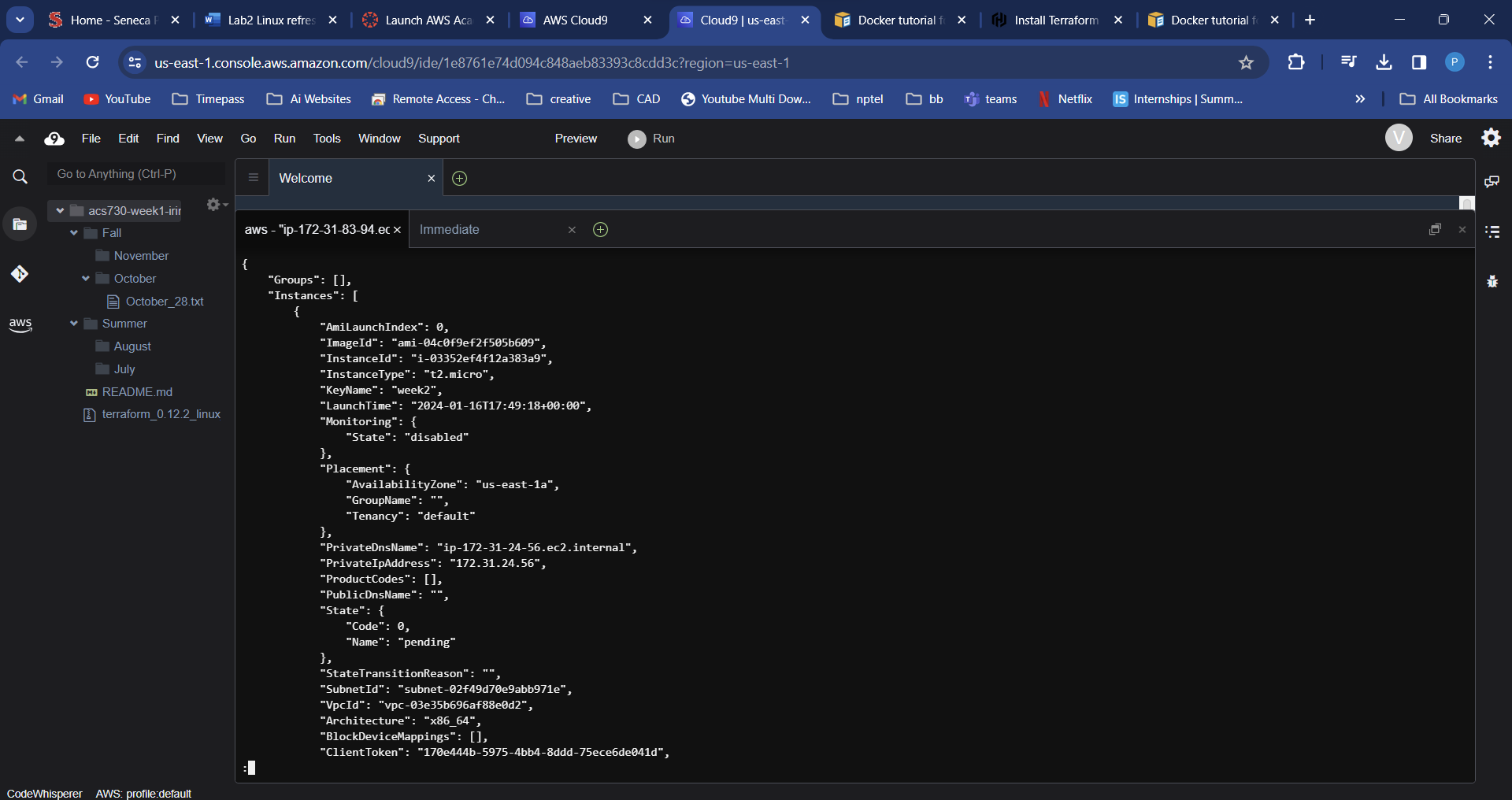




1. Create Amazon EC2 instance with ingress on port 22 with the instructions from [Lab1, Task 5. Grant](https://seneca.sharepoint.com/:w:/s/CAACurriculum-ACS730AutomationandControlSystems/EZCZgVJBGIRLmKwNklJz0s0B137XJPq_HQrDLvDpa1YgWA?e=O0pf4a) access to Amazon Cloud 9 public IPs.

# Create an EC2 instance that uses that Key pair

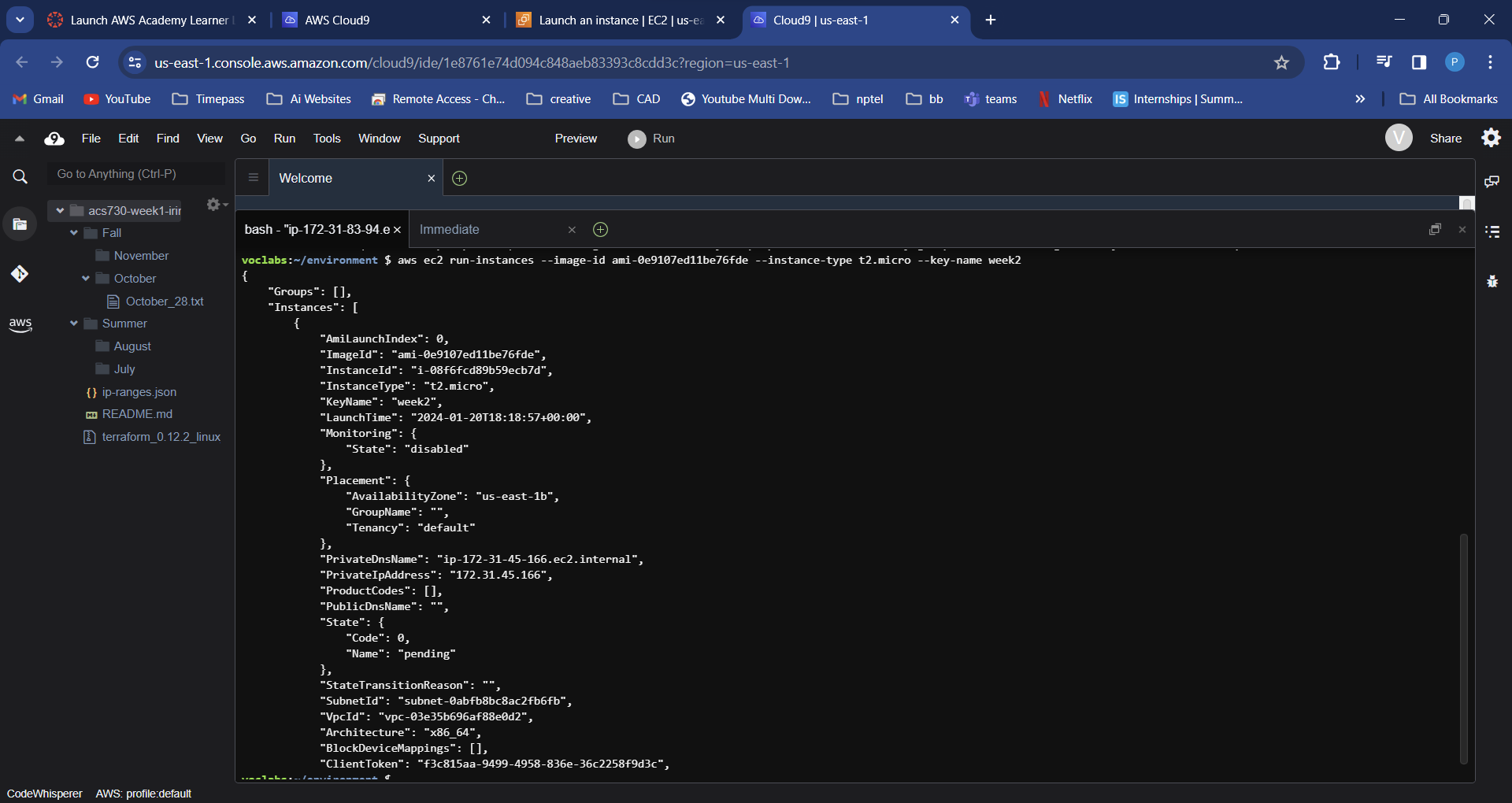
aws ec2 run-instances --image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86\_64-gp2 --instance-type t2.micro --key-name week2



**Note**: if you are getting SSM access error please run the command below:

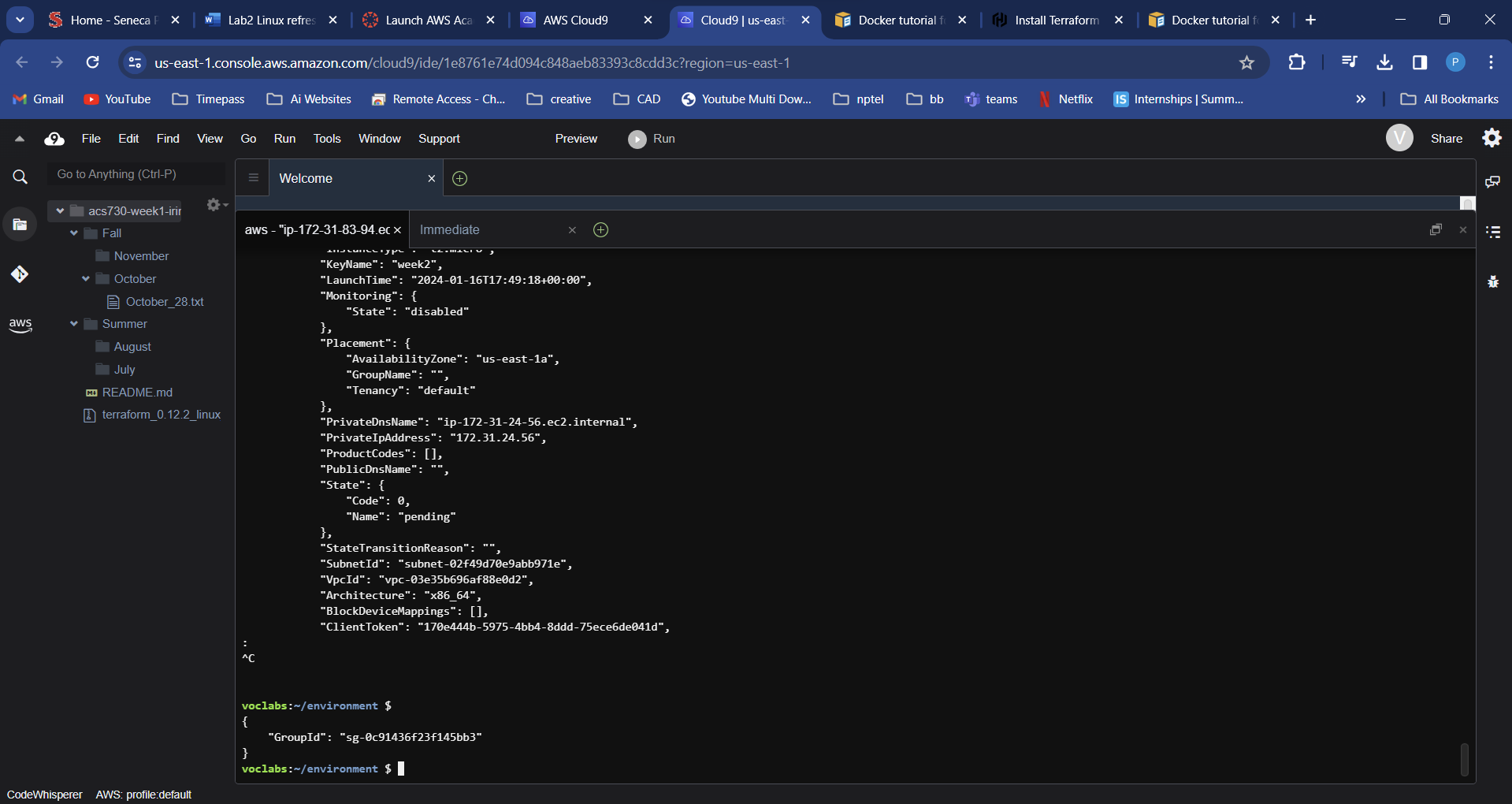
aws ec2 run-instances --image-id --instance-type t2.micro --key-name week2

ws ec2 run-instances --image-id ami-0e9107ed11be76fde --instance-type t2.micro --key-name week2



# Create the security group for admin traffic (ssh)

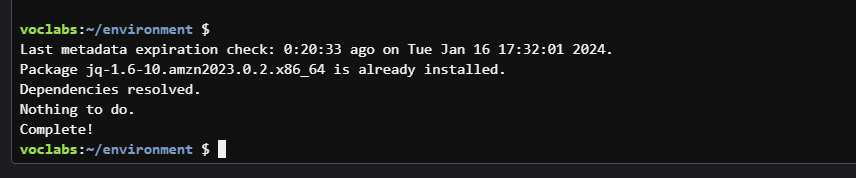
aws ec2 create-security-group --group-name acs730-week2-sg --description "Security Group for Admin Traffic"



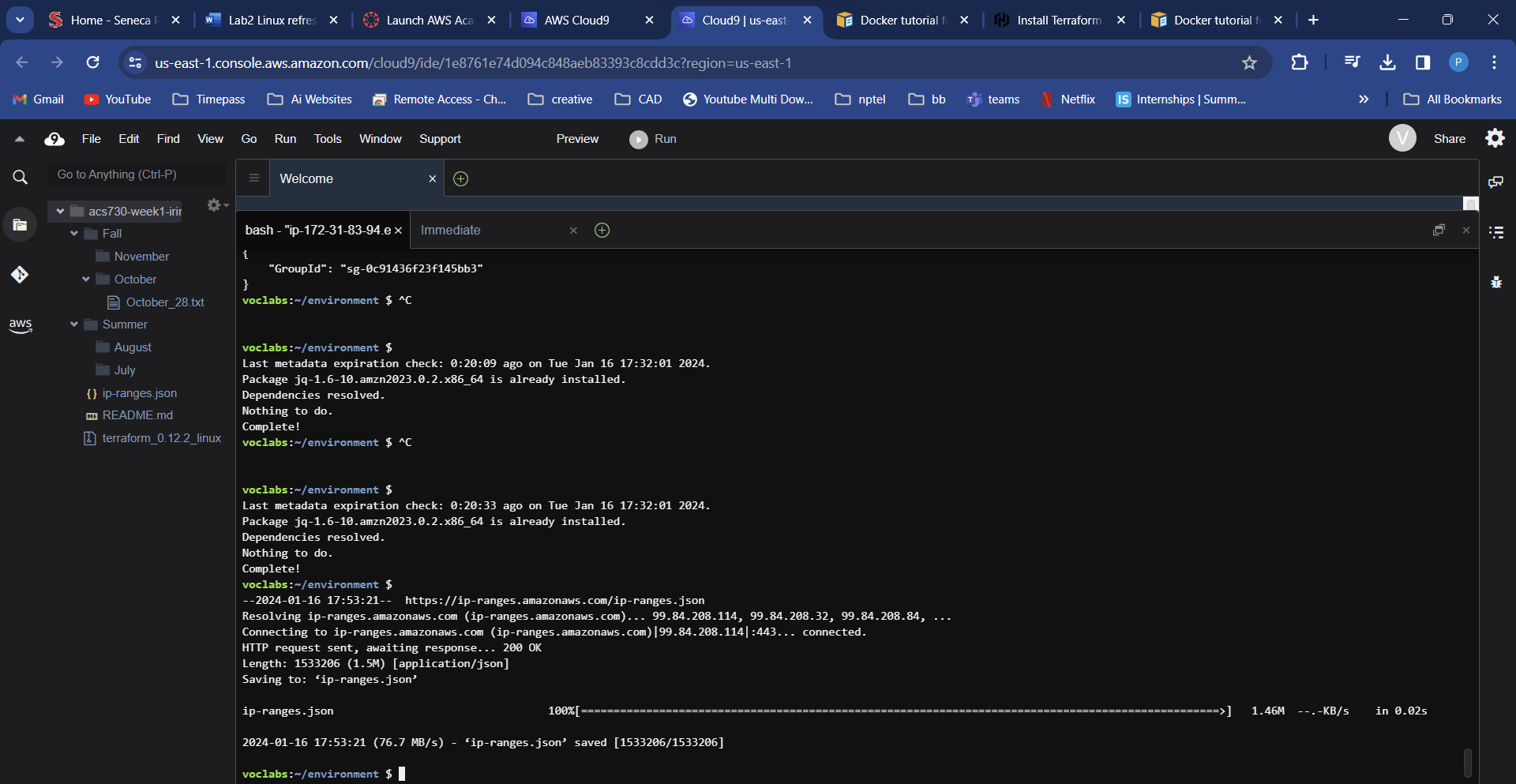
# Authorize SSH and PING access to [the public IP ranges of Cloud9 service](https://docs.aws.amazon.com/cloud9/latest/user-guide/ip-ranges.html) in us-east-1 region. Install jq in your Cloud9 environment

# Install jq

sudo yum install jq -y

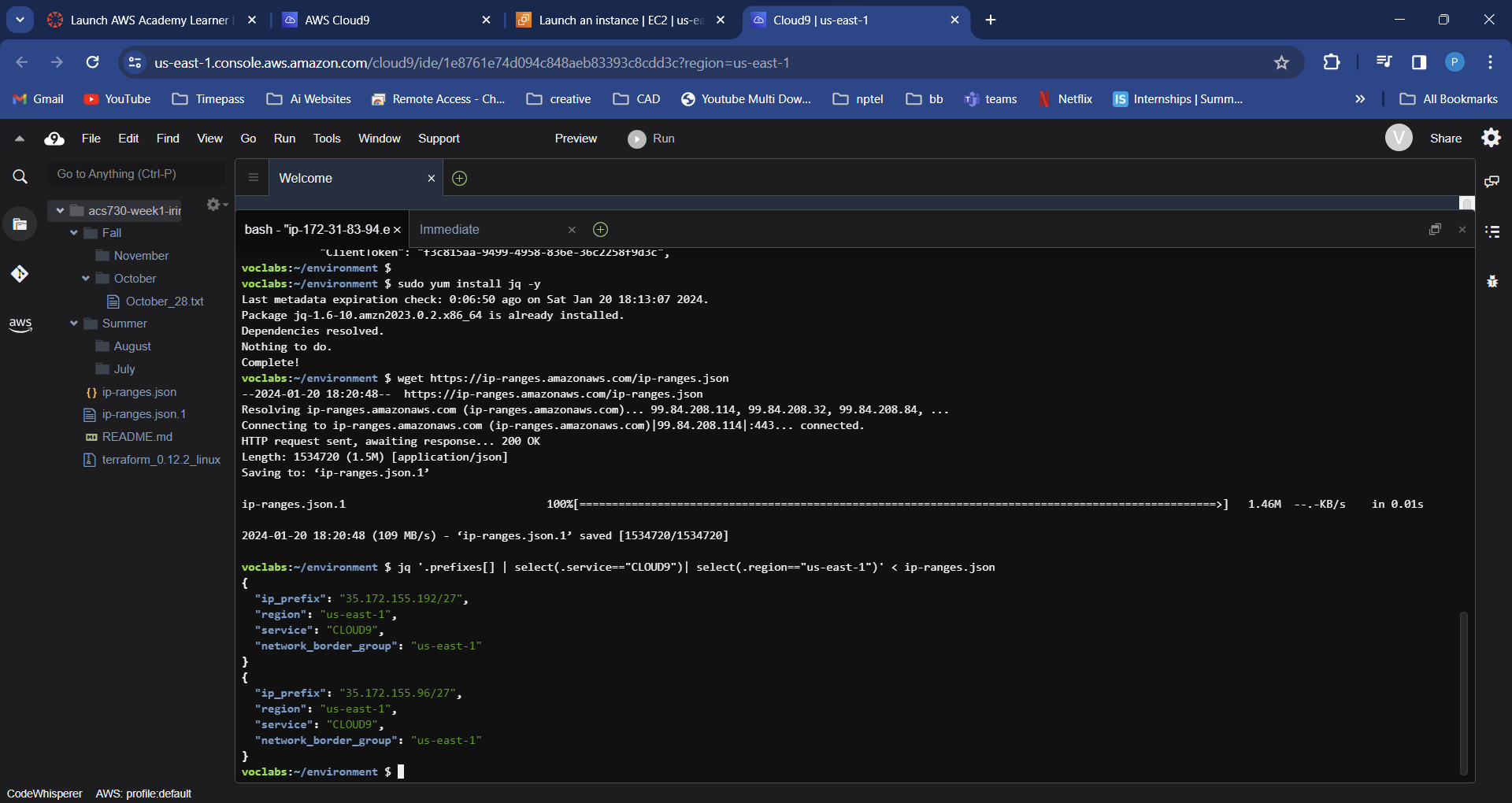
  
# Download IP ranges

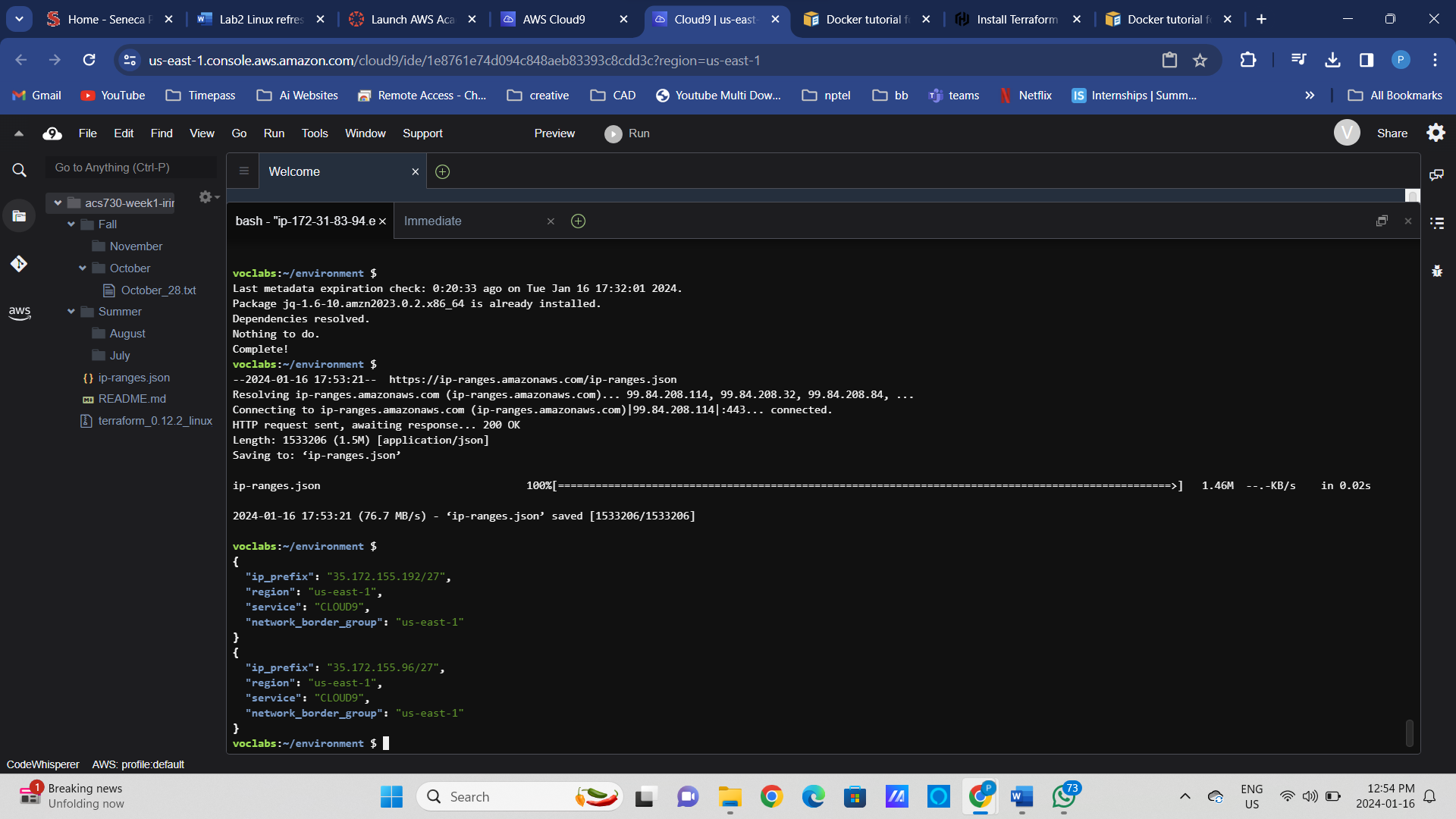
wget <https://ip-ranges.amazonaws.com/ip-ranges.json>



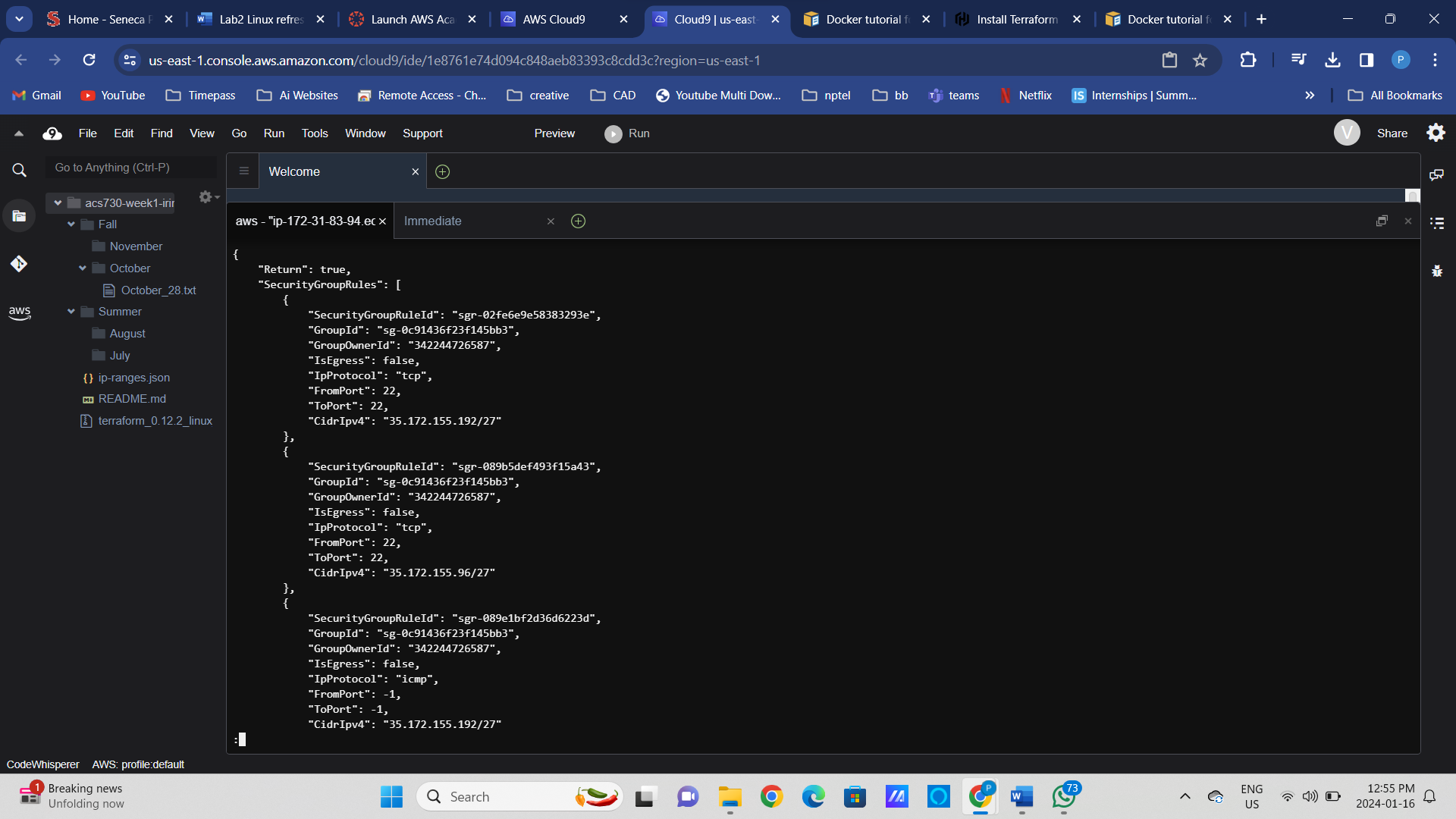
# Get all the IP ranges for CLOUD9 environment

jq '.prefixes[] | select(.service=="CLOUD9")| select(.region=="us-east-1")' < ip-ranges.json





aws ec2 authorize-security-group-ingress --group-name acs730-week2-sg --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges=[{CidrIp=35.172.155.192/27},{CidrIp=35.172.155.96/27}] IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=35.172.155.192/27},{CidrIp=35.172.155.96/27}]



export INSTANCE\_ID=i-039846ca048ac228a

export INSTANCE\_ID=i-04bef30b98ab34e3c

export SG\_ID=sg-03c8aacfc095f4070

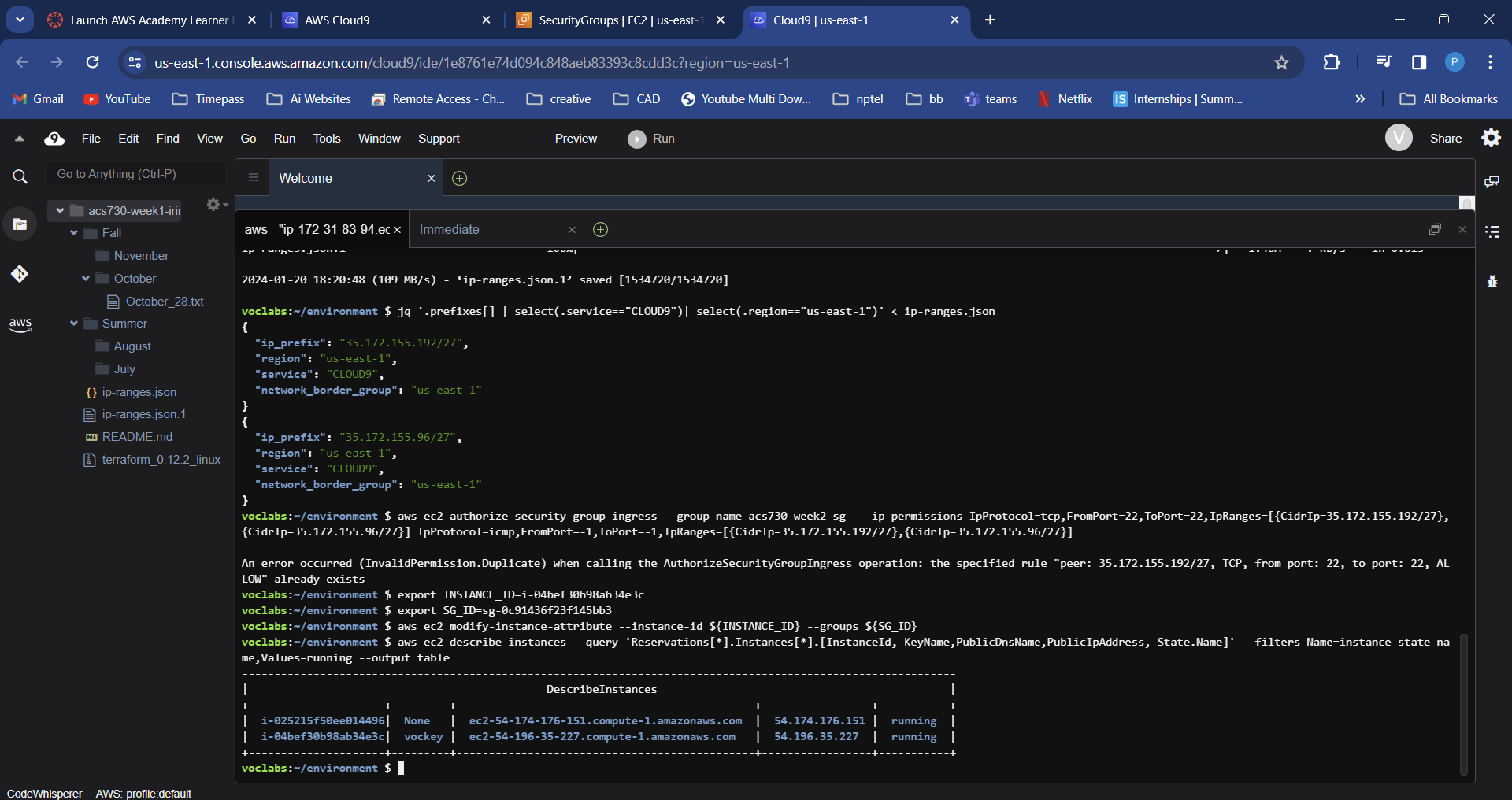
export SG\_ID=[sg-0c91436f23f145bb3](https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#SecurityGroup:groupId=sg-0c91436f23f145bb3)

aws ec2 modify-instance-attribute --instance-id ${INSTANCE\_ID} --groups ${SG\_ID}

aws ec2 modify-instance-attribute --instance-id ${INSTANCE\_ID} --groups ${SG\_ID}

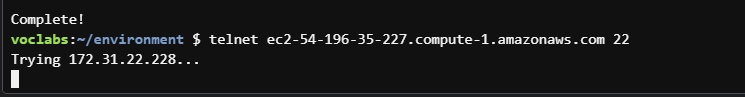
1. Verify the instance was created successfully

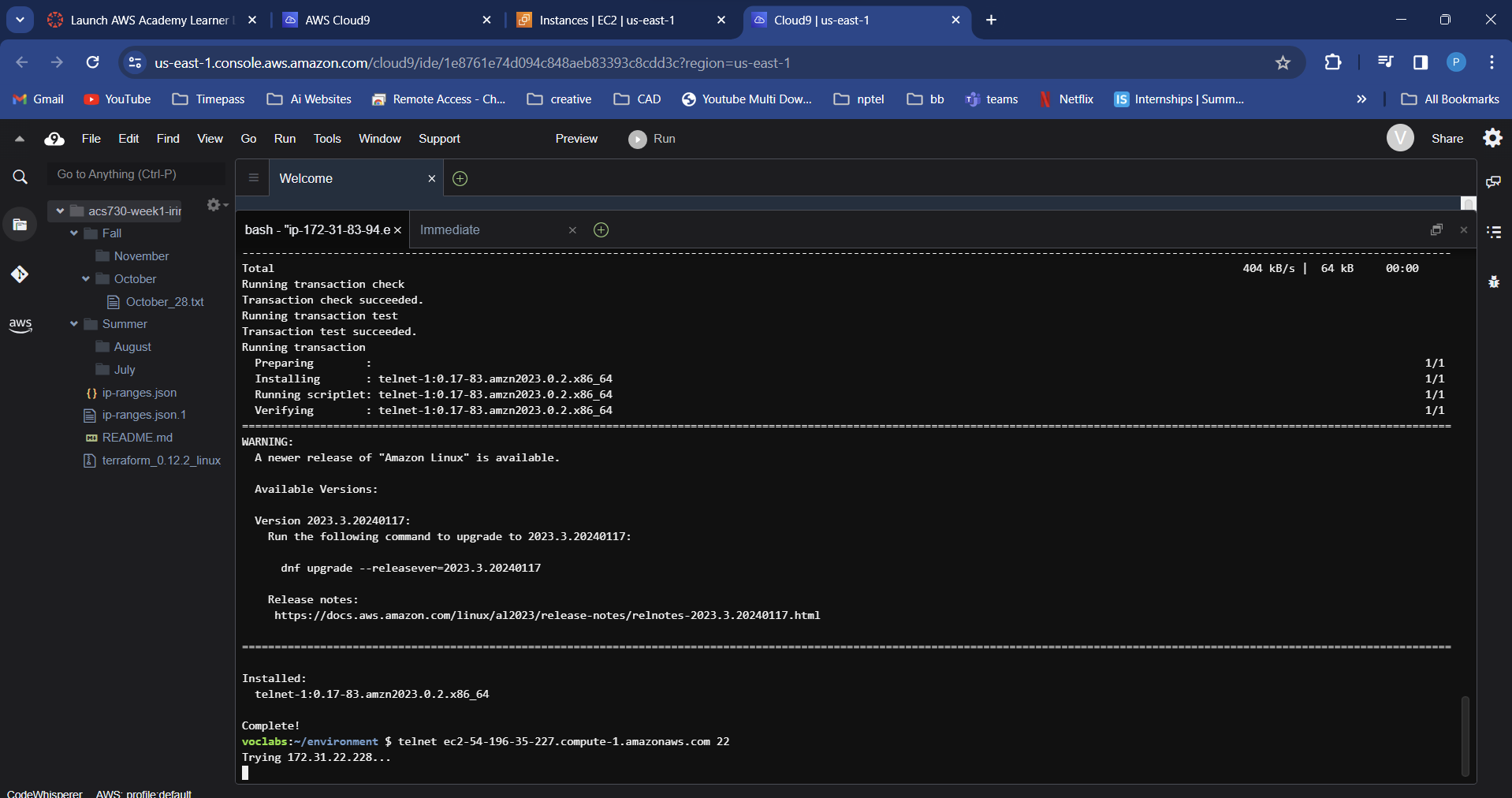
aws ec2 describe-instances --query 'Reservations[\*].Instances[\*].[InstanceId, KeyName,PublicDnsName,PublicIpAddress, State.Name]' --filters Name=instance-state-name,Values=running --output table



1. Verify network connectivity to port 22 of the provisioned instance using its public DNS

|  |  |
| --- | --- |
| Can we resolve the DNS? | nslookup ec2-54-161-142-127.compute-1.amazonaws.com  nslookup ec2-54-196-35-227.compute-1.amazonaws.com    dig ec2-54-161-142-127.compute-1.amazonaws.com  dig ec2-54-196-35-227.compute-1.amazonaws.com |
| Is the local link up? | ip link |
| Is the remote server responding? | sudo yum install telnet      telnet ec2-54-161-142-127.compute-1.amazonaws.com 22  telnet ec2-54-196-35-227.compute-1.amazonaws.com 22 |

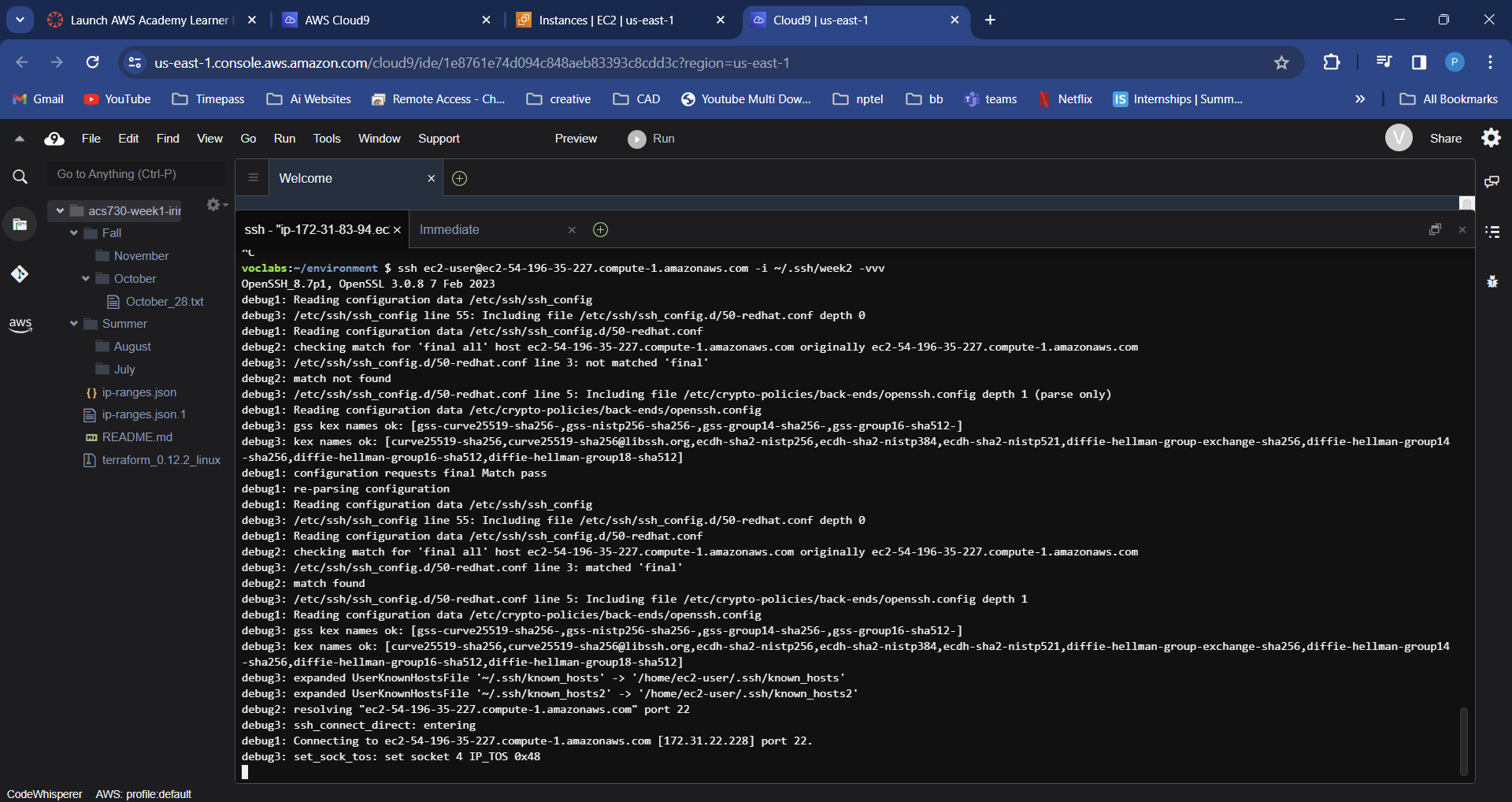




**Note the IP address returned by DNS lookups. Why is it different from the IP address in the EC2 public DNS name?**

1. The username ec2-user is pre-provisioned by AWS when the CentOS based EC2 instance is created. Attempt connecting the EC2 using your local Linux user and ec2-user.

|  |  |
| --- | --- |
| Find out your local username | whoami    Id |
| Try connecting to the EC2 instance  Did it work?  Let's debug | ssh ec2-user@ec2-54-161-142-127.compute-1.amazonaws.com -i ~/.ssh/week2  ssh ec2-user@ec2-54-196-35-227.compute-1.amazonaws.com -i ~/.ssh/week2    ssh ec2-user@ec2-54-161-142-127.compute-1.amazonaws.com -i ~/.ssh/week2 -vvv  ssh ec2-user@ec2-54-196-35-227.compute-1.amazonaws.com -i ~/.ssh/week2 -vvv |
|  |  |

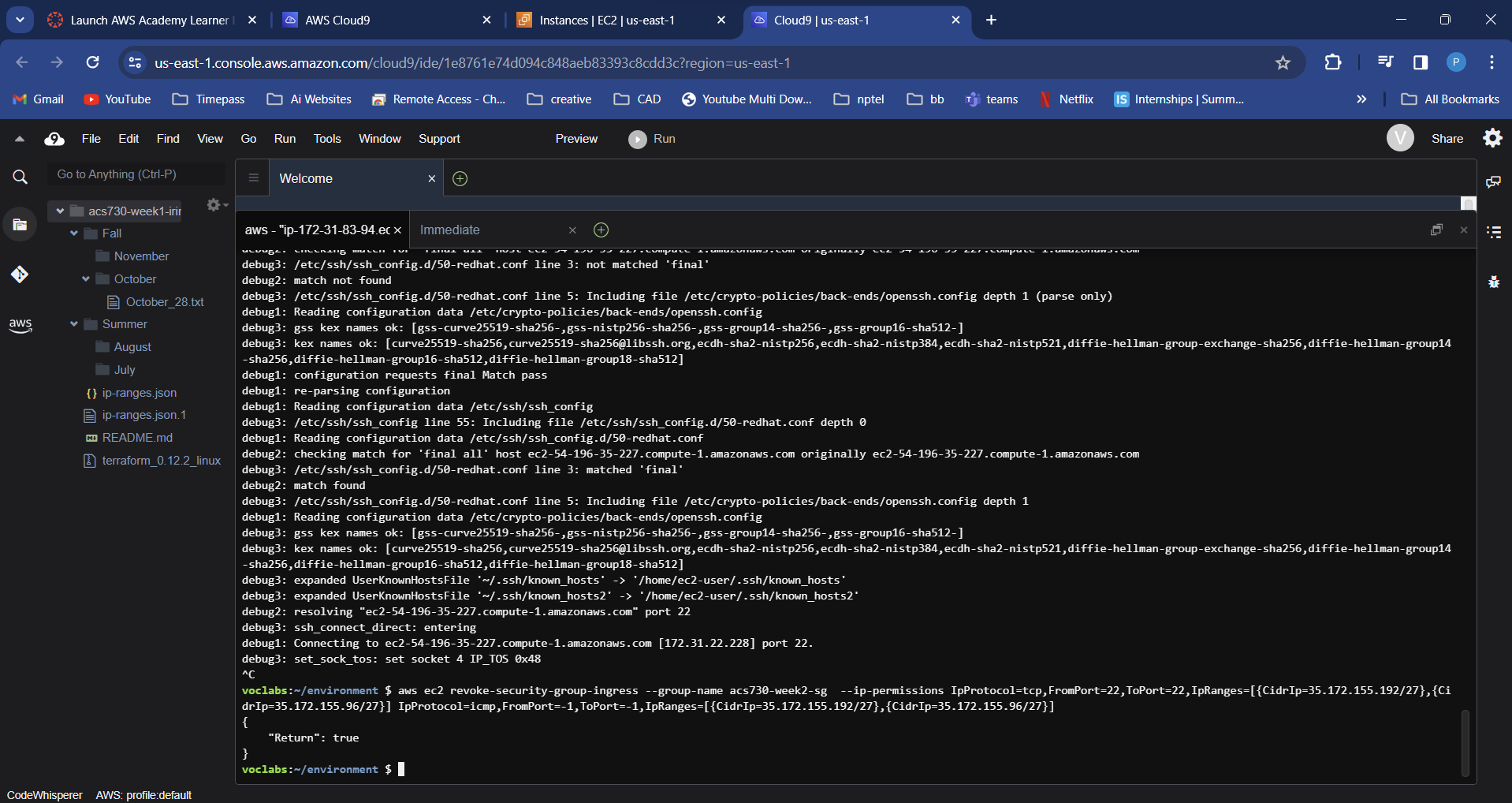


1. Cloud 9 environment connects to our EC2 instance using its private IP address. Let’s fix our security group.

# Revoke previously defined access rule

aws ec2 revoke-security-group-ingress --group-name acs730-week2-sg --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges=[{CidrIp=35.172.155.192/27},{CidrIp=35.172.155.96/27}] IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=35.172.155.192/27},{CidrIp=35.172.155.96/27}]

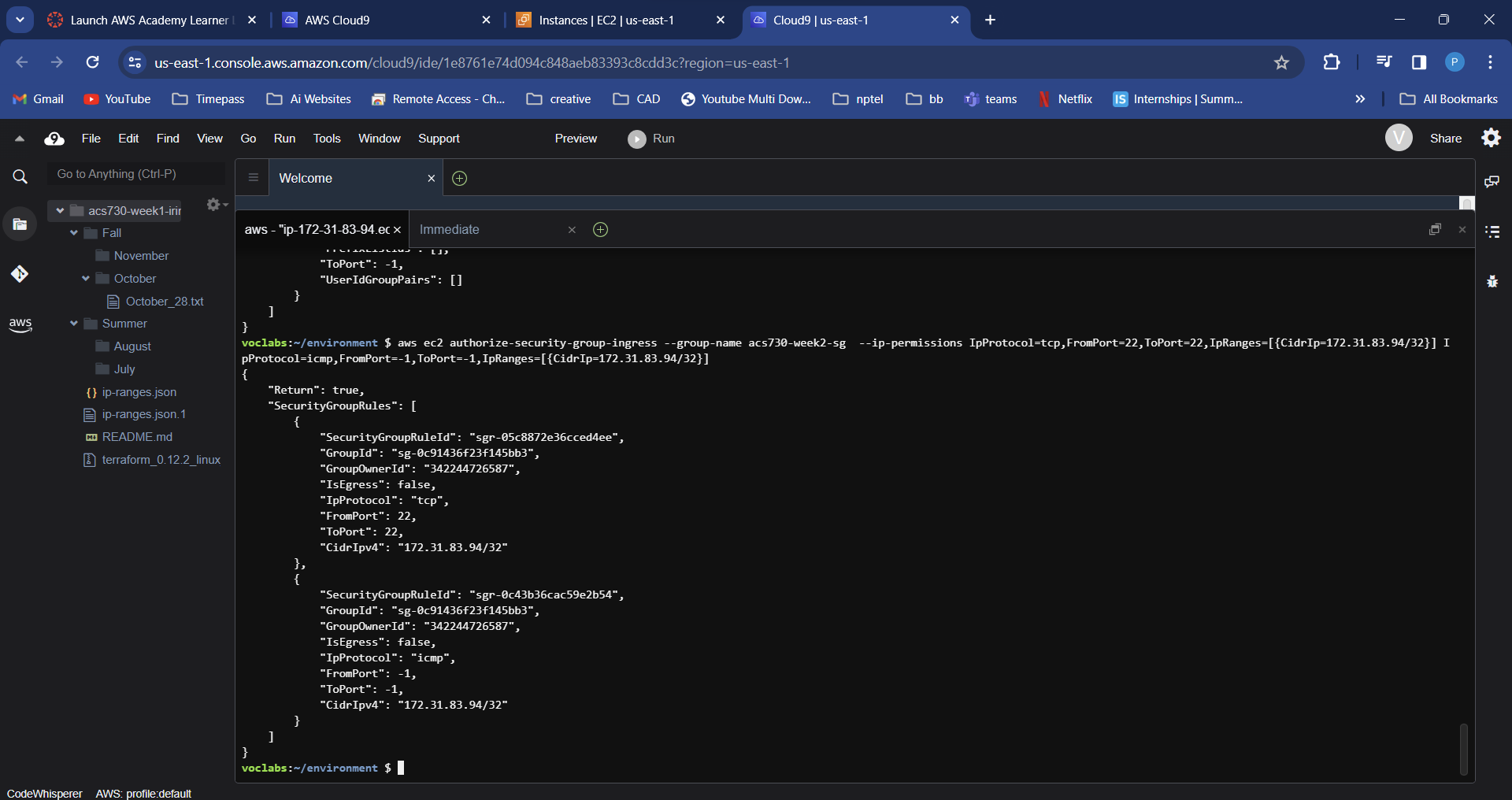
aws ec2 revoke-security-group-ingress --group-name acs730-week2-sg --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges=[{CidrIp=172.31.83.94/27},{CidrIp=172.31.83.94/27}] IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=172.31.83.94/27},{CidrIp=172.31.83.94/27}]

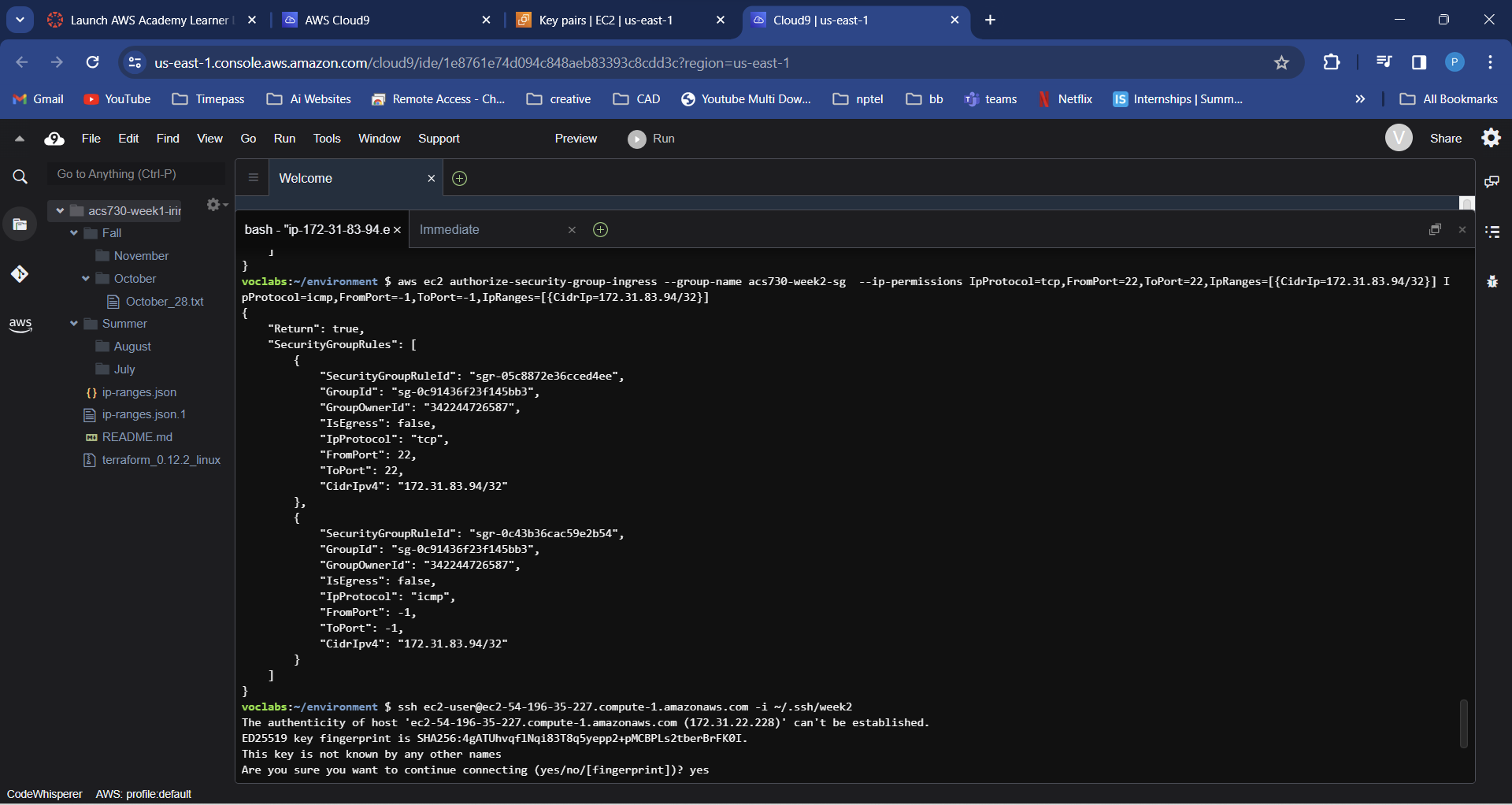


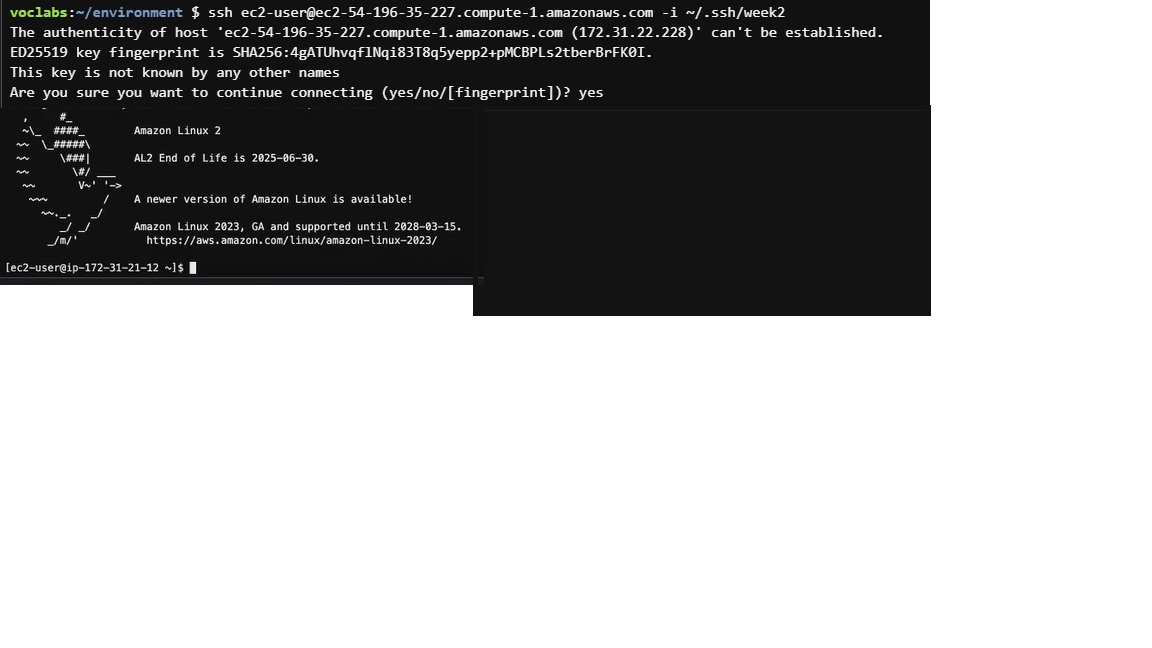
# Create new inbound rules to allow traffic from a private IP of Cloud 9 instance

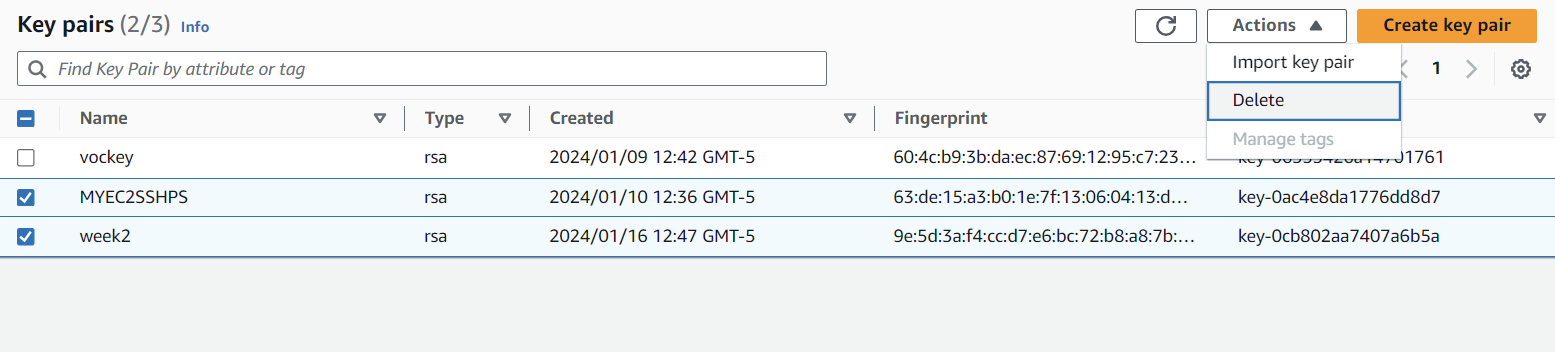
aws ec2 authorize-security-group-ingress --group-name acs730-week2-sg --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges=[{CidrIp=<Private IP of Cloud9 instance>/32}] IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=<Private IP of Cloud9 instance>/32}]

aws ec2 authorize-security-group-ingress --group-name acs730-week2-sg --ip-permissions IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges=[{CidrIp=172.31.83.94/32}] IpProtocol=icmp,FromPort=-1,ToPort=-1,IpRanges=[{CidrIp=172.31.83.94/32}]









Task 4: (OPTIONAL)Create a Linux service on AWS EC2

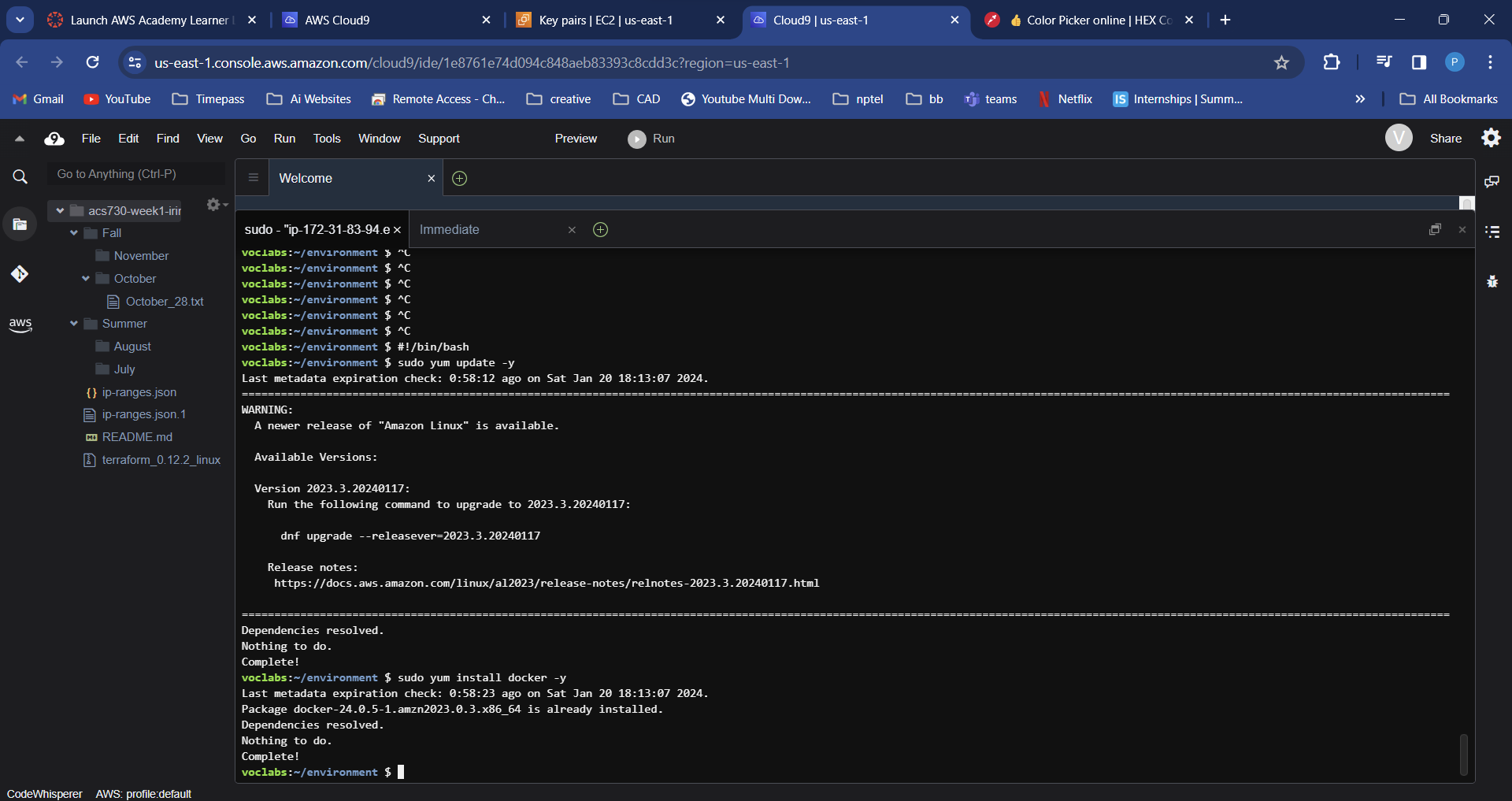
1. Create a bash script called run\_docker.sh in your Cloud9 IDE.

The content of the script is below. You can find this script in the Lab2 folder.

#!/bin/bash

sudo yum update -y

sudo yum install docker -y

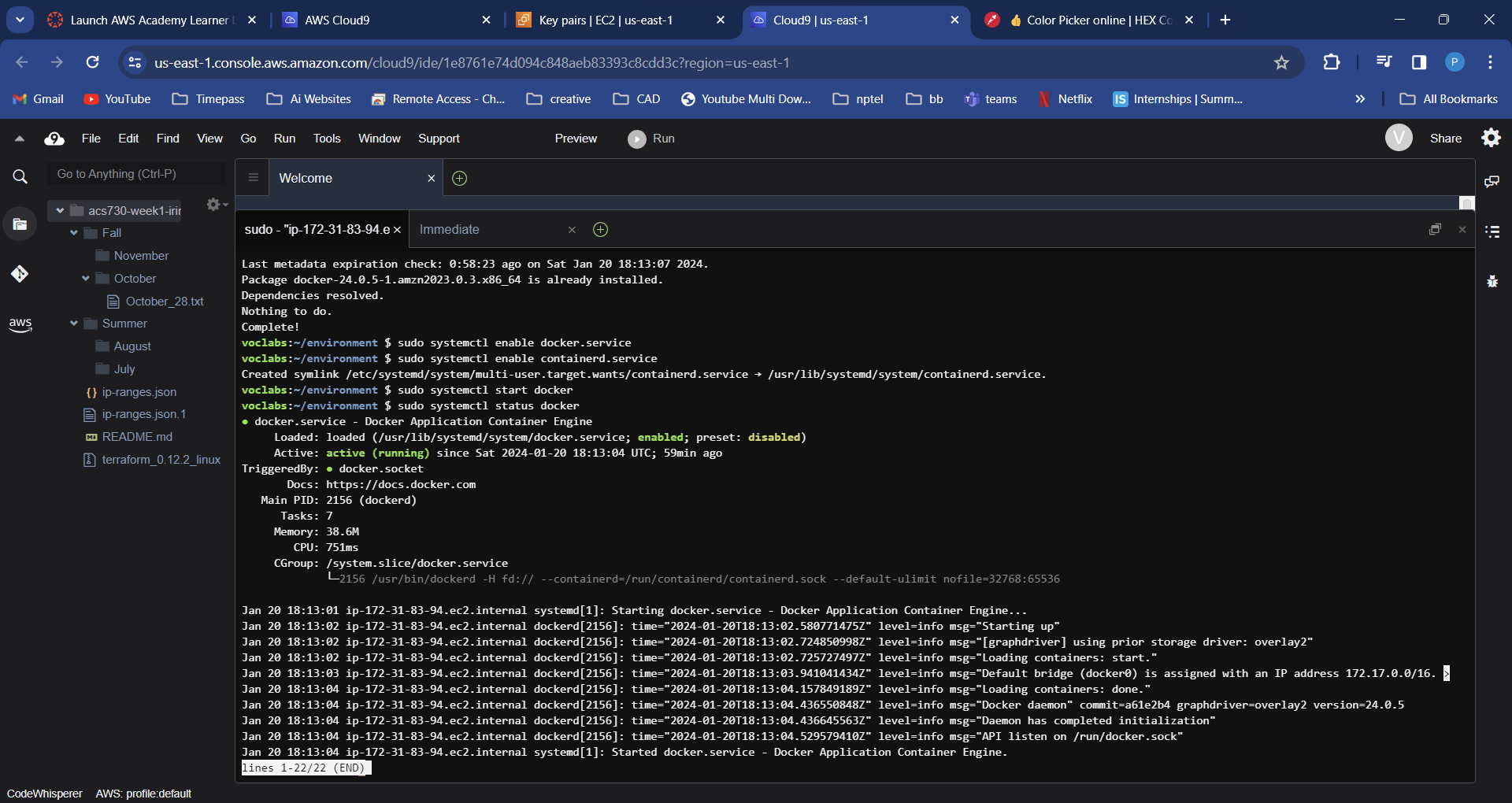


sudo systemctl enable docker.service

sudo systemctl enable containerd.service

sudo systemctl start docker

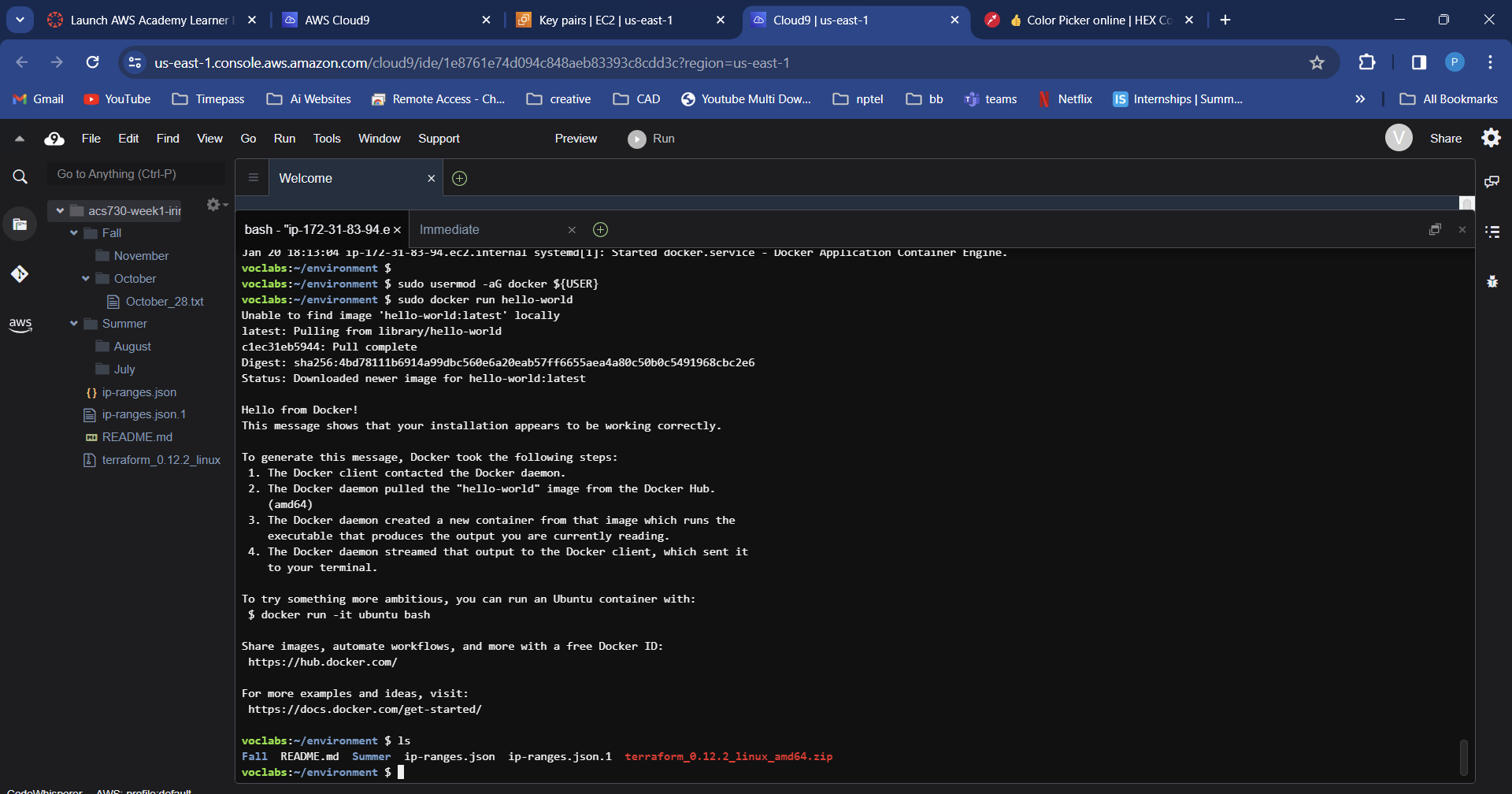
sudo systemctl status docker



sudo usermod -aG docker ${USER}

sudo docker run hello-world

Make sure the file was created

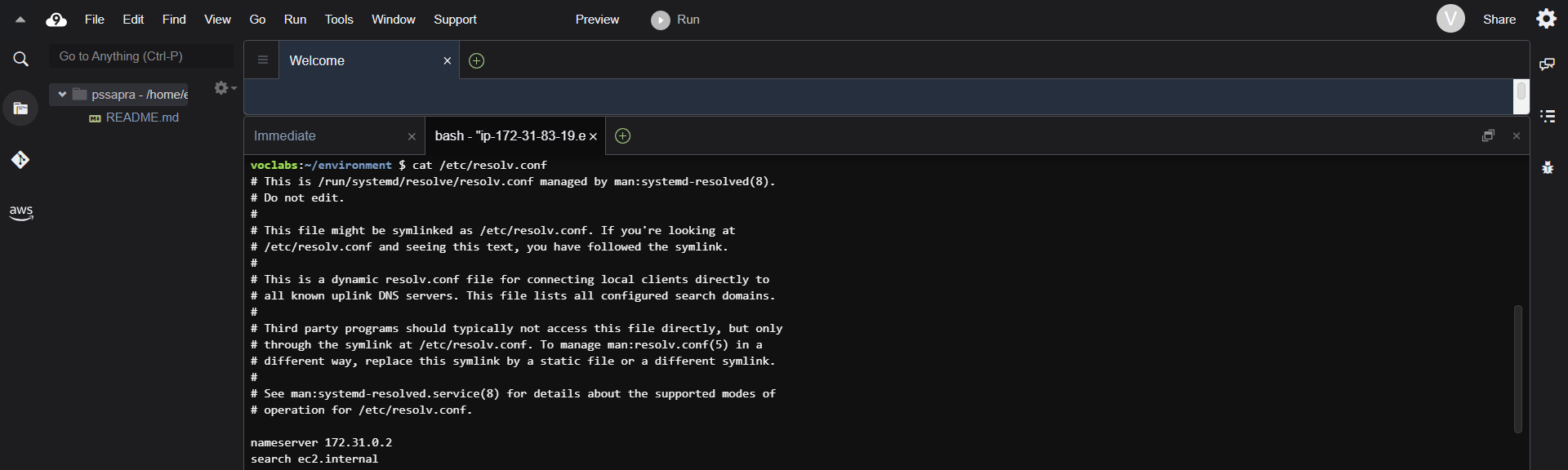


**Slide 25**

* What is the IP of the nameserver on the current VM?

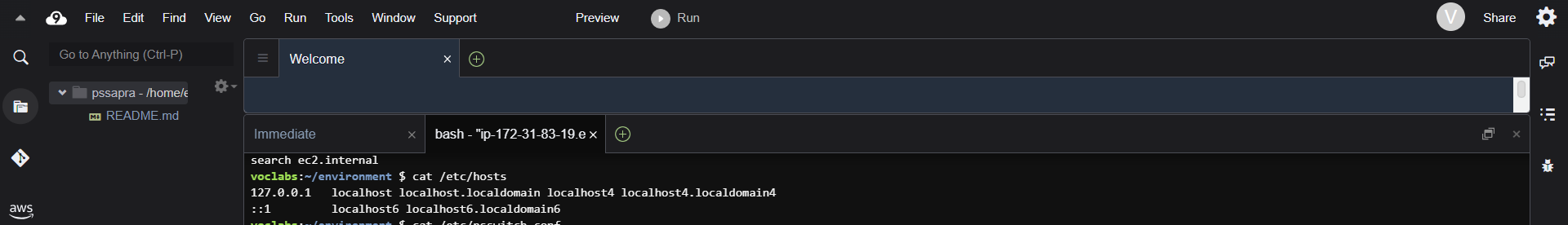
check /etc/resolv.conf

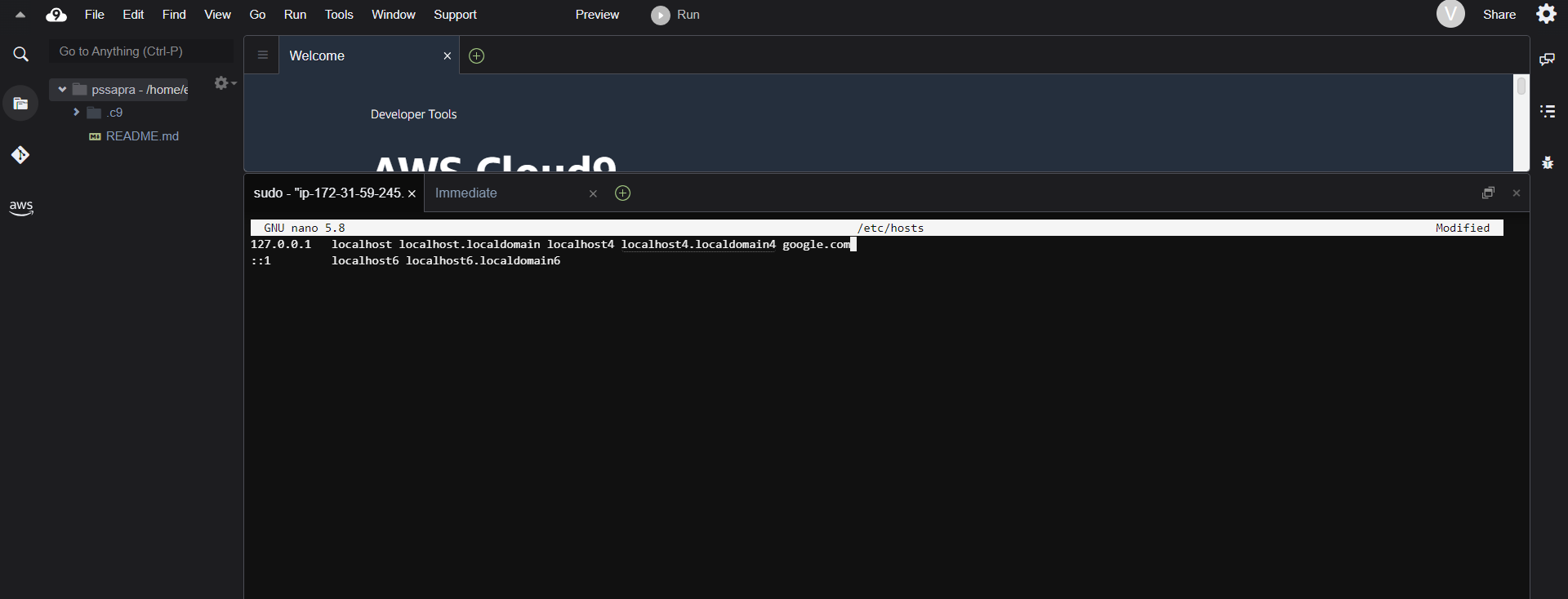
cat /etc/resolv.conf

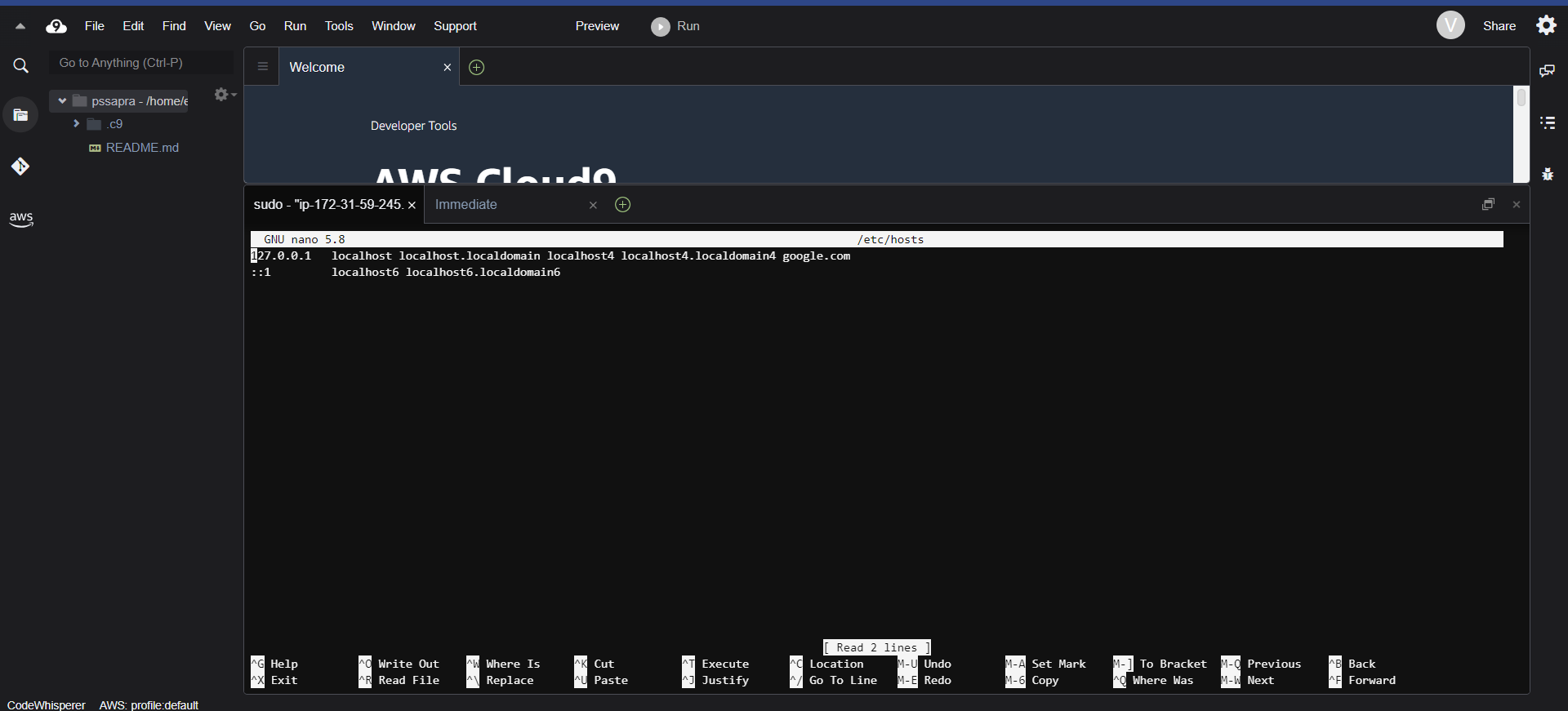


* Override the IP of google.com locally to point to 127.0.0.1

edit /etc/hosts





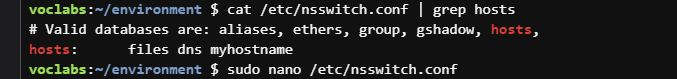


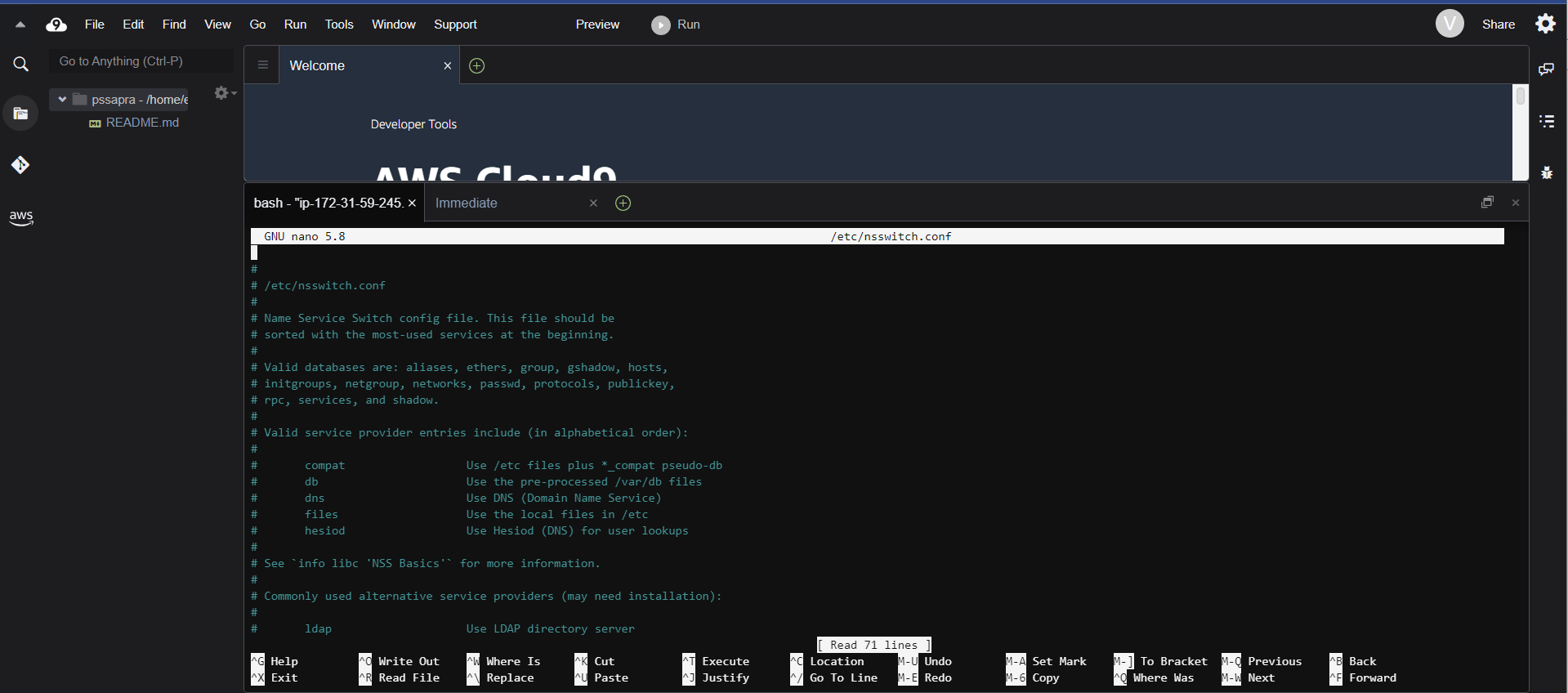
* What is the order that DNS resolution is configured on this VM?
  + Local file first, name server second or the other way around?
  + Change the configuration

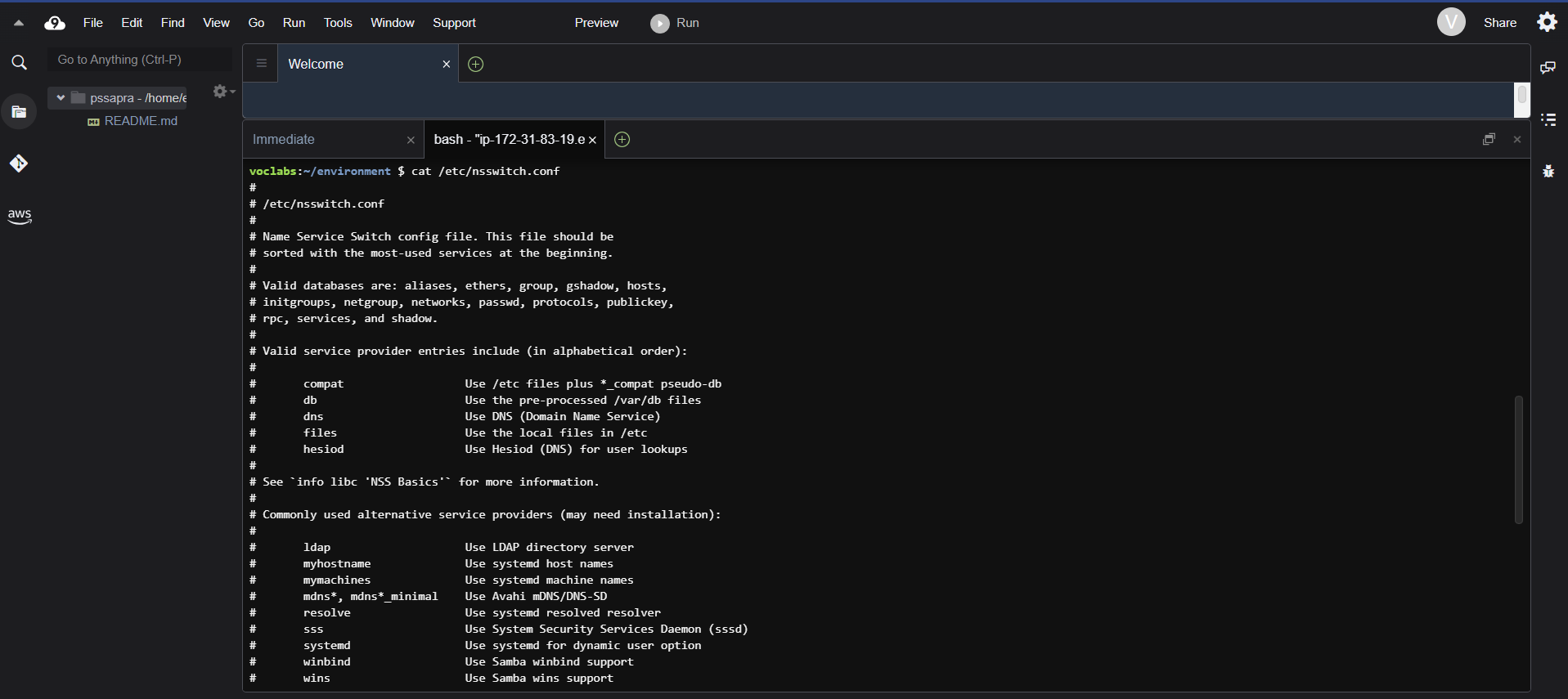
/etc/nsswitch.conf

cat /etc/nsswitch.conf | grep hosts

sudo nano /etc/nsswitch.conf

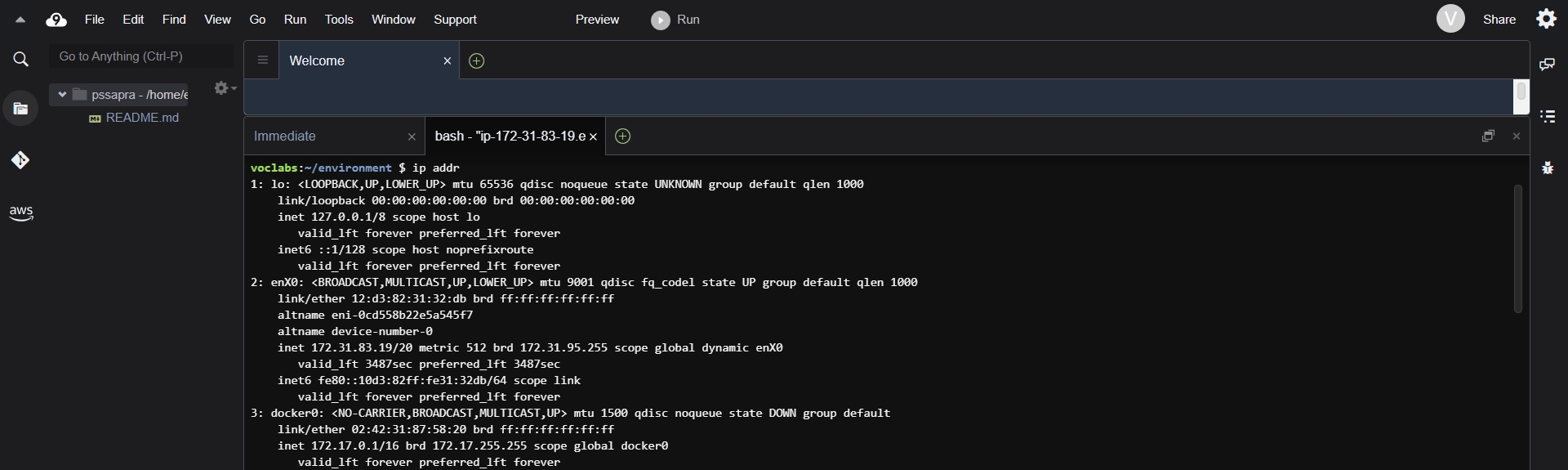






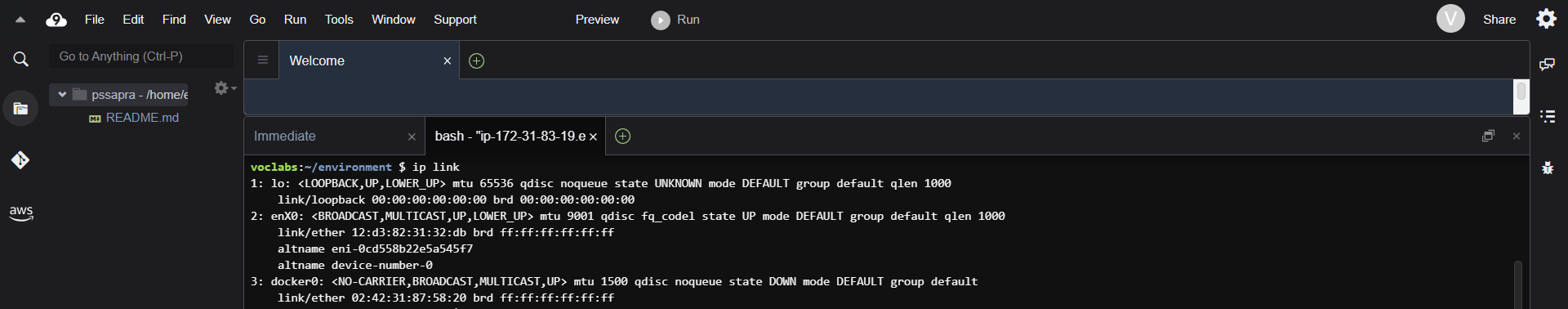
* What is the IP address of your VM?

ip addr



* Inspect the interfaces of the VM. Are all of them up?

ip link



* What is the default gateway of your VM?

route | grep default

