**AWS Resource Tracker**

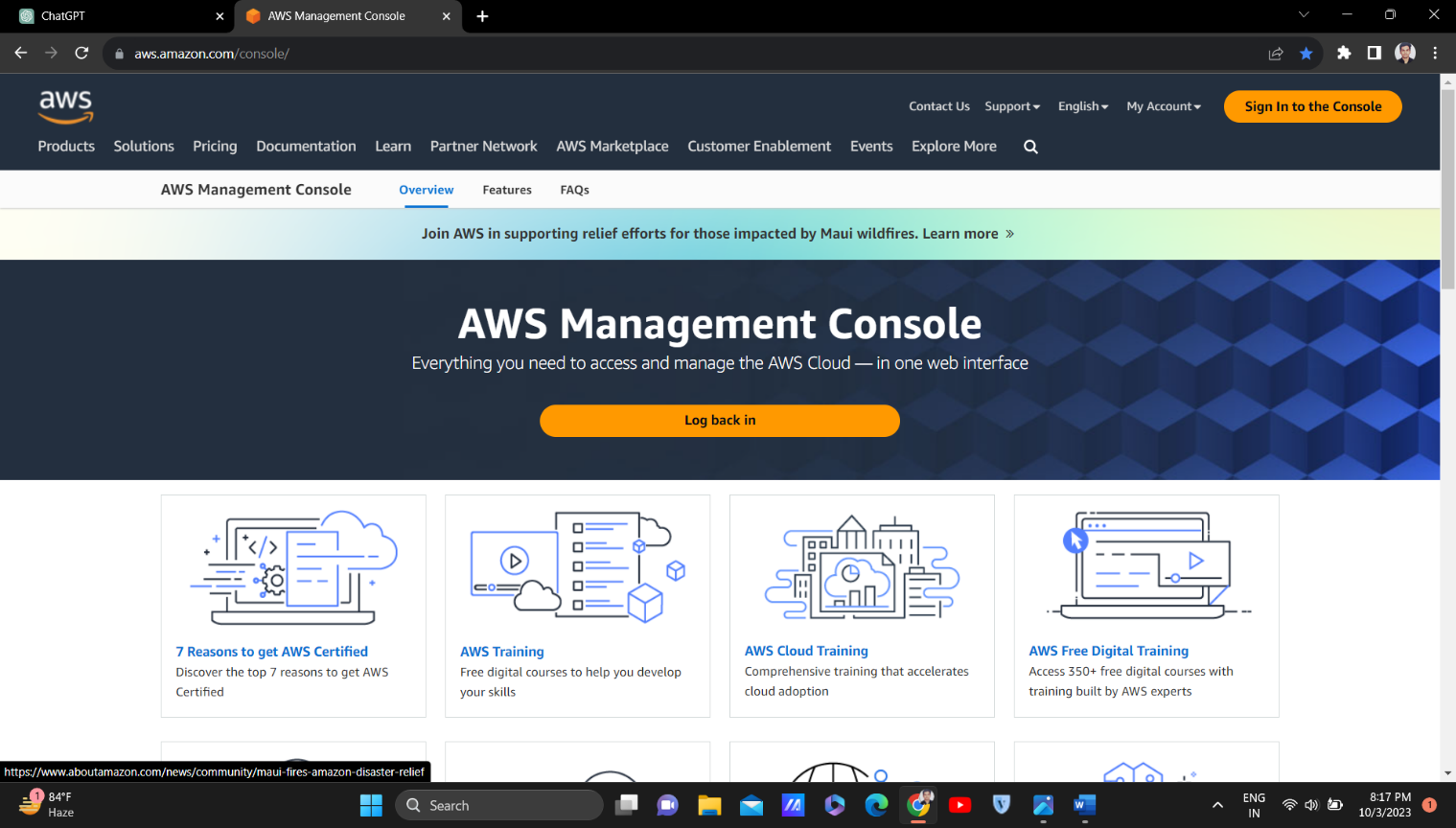
In this documentation we will see how we can track down ongoing services in our Aws account and integrate it with cronjob to send the email of usage report.

If you already have a Linux machine you can skip step 1 and directly jump on step 2.

**Step 1. Launch an EC2 Instance**

* <https://aws.amazon.com/console/> using this website login to AWS Console, if you do not

have an AWS account please sign up first and then login.



* After you successfully login to your aws account, Search for EC2 and press enter, you will see the interface as shown in fig 1.1
* Now click on Launch Instance
* Now name your instance (for eg. Testec2)
* Now in Application and OS image, select Amazon Linux.
* In Amazon Machine Image, select Amazon Linux 2023 AMI free eligible tier. Remember, always select the free tier to avoid getting charged for EC2 instance.
* In Instance type, select t2 micro free eligible tier. Remember, always select the free tier if you do not have a heavy task to perform. Amazon will charge for other instance types except the free ones.

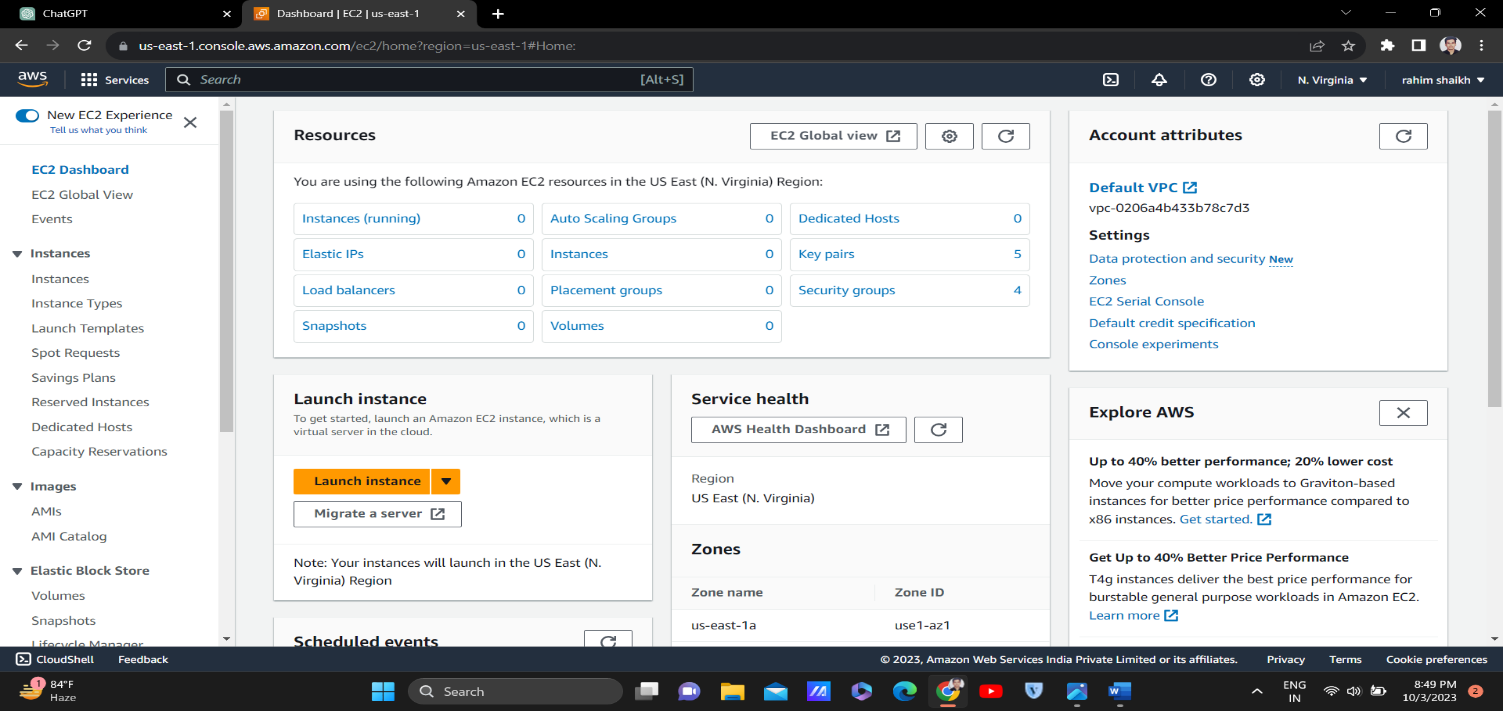


Fig 1.1

* Now in Key Pair, click on create on a new key pair, name the key pair (for eg. ec2key), in key pair type click RSA and in Private key file format click on .ppk and then click on Create Key Pair. After creating, your key pair will automatically get downloaded.
* Keep other settings by default and now click on Launch Instance and your Testec2 instance will be created.
* Now select that instance and click on Connect to launch the instance, once the instance is launched you can see the below interface as show in fig 1.2

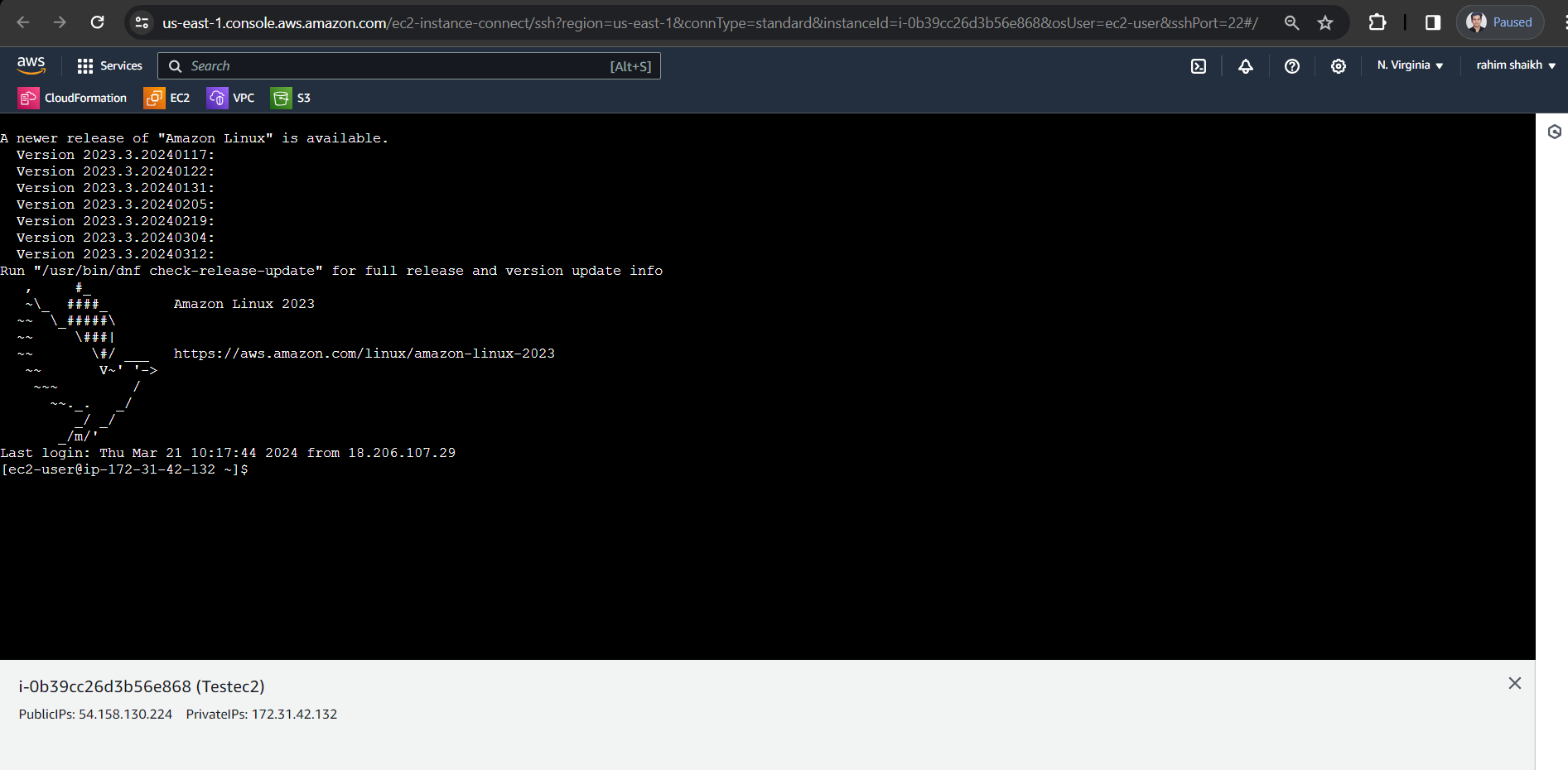
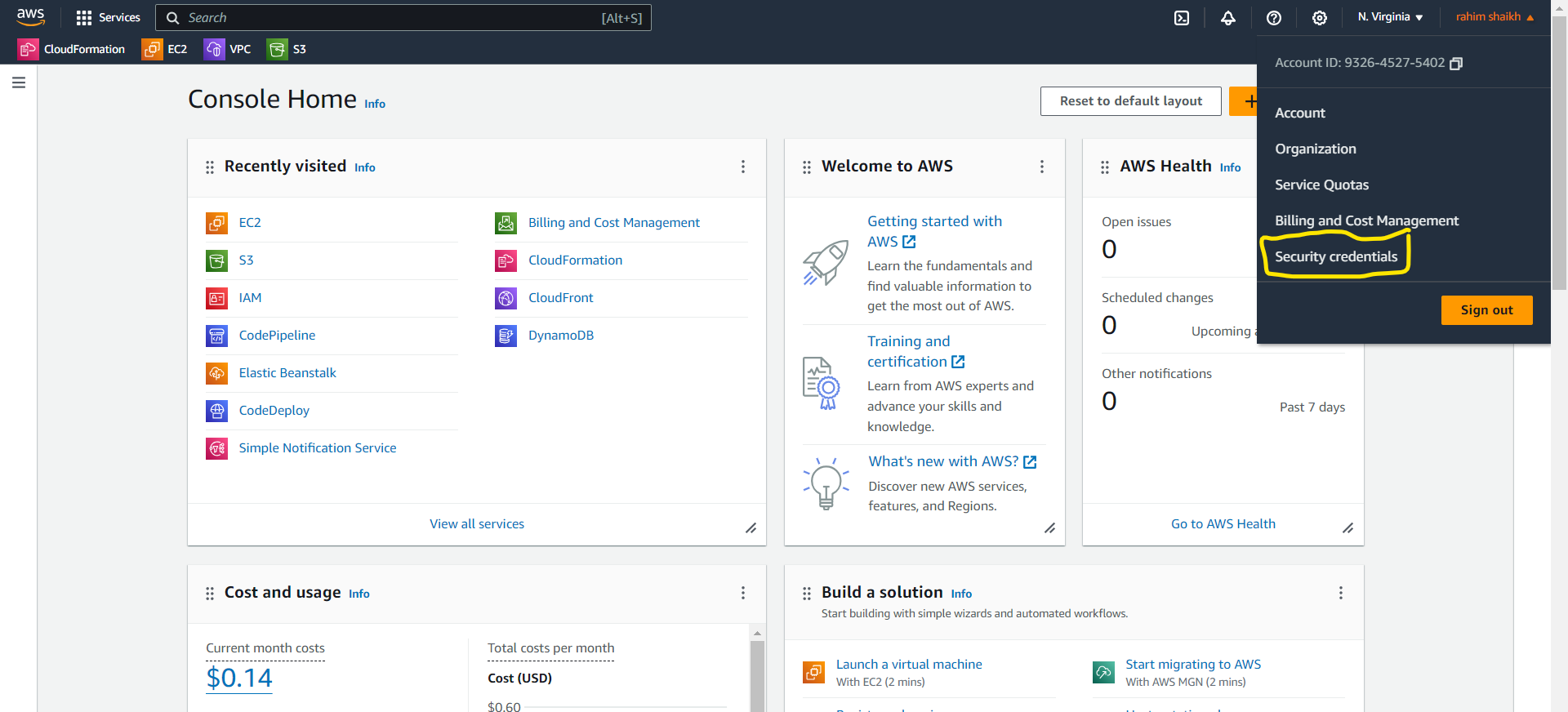


Fig 1.2

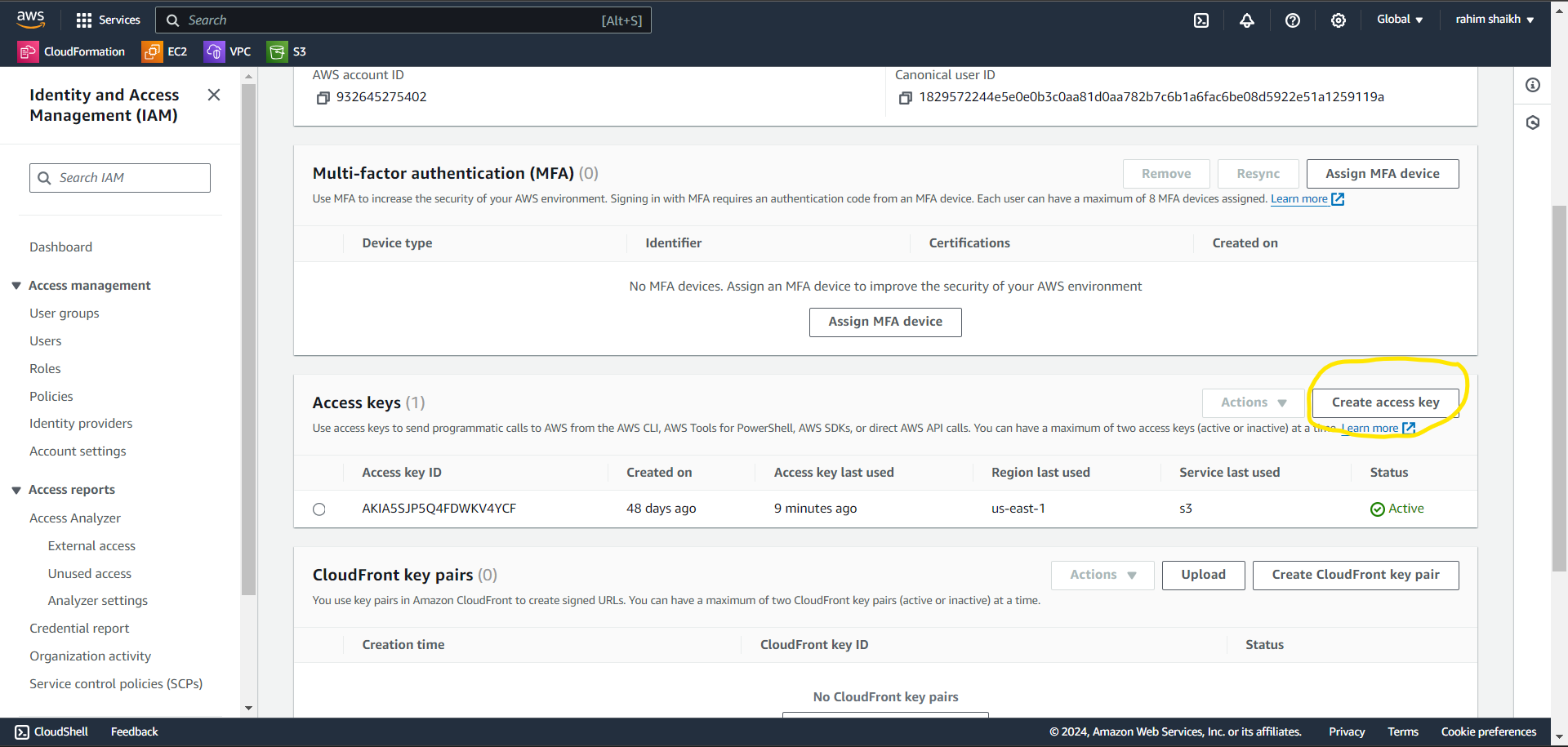
**Step 2. Login to your Aws Account from your instance**

Once you have logged into your ec2 instance or Linux machine, now we need to login to our aws account from the ec2 instance, so that our instance can get access to the ongoing services.

First, we need to create an access key, go to the home page of you aws account



Go to Security credentials and scroll down to Access keys section and create an access key.



After you create your access key copy the access key and secret access key.

Go back to your instance and type the following commands:

* “sudo su –” to switch to root user.
* “yum update” update installed packages on the system.
* “aws configure” to login to your aws account from your instance, it will ask you for access key and secret access key, paste it and press enter now your instance have the access to your aws account.

**Step 3. Write the shell script**

Create and open the file by the following command: vi awsresourcetracker.sh, a file named awsresourcetracker.sh will be created and opened. Now we can write our bash shell script in it. First press “esc” and “i” to enter into the insert mode.

I have written the below shell script so you can copy paste it, this shell script tracks only 4 services i.e. EC2, S3, VPC, IAM USER, if you want you can edit the script for more services.

---------------------------------------------------------------------------------------------------------------------

#!/bin/bash

########################################################

#Author: Naveed Shaikh

#Date: 02/02/2024

#Purpose: This script tracks the aws resources in your aws account.

#Version: v1

########################################################

set -x #debug mode

#This command list all the EC2 instances in your Aws Account

echo "Name InstanceId State"

aws ec2 describe-instances | jq -r '.Reservations[].Instances[] | [(.Tags[]? | select(.Key == "Name").Value // "NoName"), .InstanceId, .State.Name] | "\(.[0]) \(.[1]) \(.[2])"'

echo "-------------------------------------------------------------------------------------------------------"

#This command list all the S4 buckets in your Aws Account

echo "Name of S3 Buckets"

aws s3 ls

echo "-------------------------------------------------------------------------------------------------------"

#This command list all the VPC in your Aws Account

echo "Name of all the VPCs"

aws ec2 describe-vpcs --query 'Vpcs[\*].[Tags[?Key==`Name`].Value | [0], VpcId]' --output text | awk '{$1=$1;print}'

echo "-------------------------------------------------------------------------------------------------------"

#This command list all the IAM Users in your Aws Account

echo "List of IAM Users and UserId"

aws iam list-users | jq '.Users[] | "\(.UserName) \(.UserId)"

---------------------------------------------------------------------------------------------------------------------

Once you have pasted the above shell script you can save it by pressing “esc” and “:wq!” and press enter to save it.

**Step 4. Installation of Postfix and Mailx**

Postfix: Open-source mail transfer agent (MTA) for routing and delivering electronic mail efficiently and securely.

Mailx: Command-line utility for sending and receiving email, commonly used in Unix-like operating systems for automation and system administration tasks.

Command to install postfix: “**yum install postfix -y**”

Command to install mailx: “**yum install mailx**”

**Postfix configuration:** Now we have to add some lines in postfix main.cf file. So go to this location by entering this command **“cd /etc/postfix/”**

Open main.cf file and search for relayhost and add this line **relayhost = [smtp.gmail.com]:587**

Now save this file and check your hostname by **hostname -f** and copy your host name. Open main.cf file search for myhostname and paste **myhostname= “*hostname\_you\_just\_copied*”** and at the end for this main.cf file paste these lines:

**#Location of sasl\_passwd we saved**

**smtp\_sasl\_password\_maps = hash:/etc/postfix/sasl/sasl\_passwd**

**#Enables SASL authentication for postfix**

**smtp\_sasl\_auth\_enable = yes**

**smtp\_tls\_security\_level = encrypt**

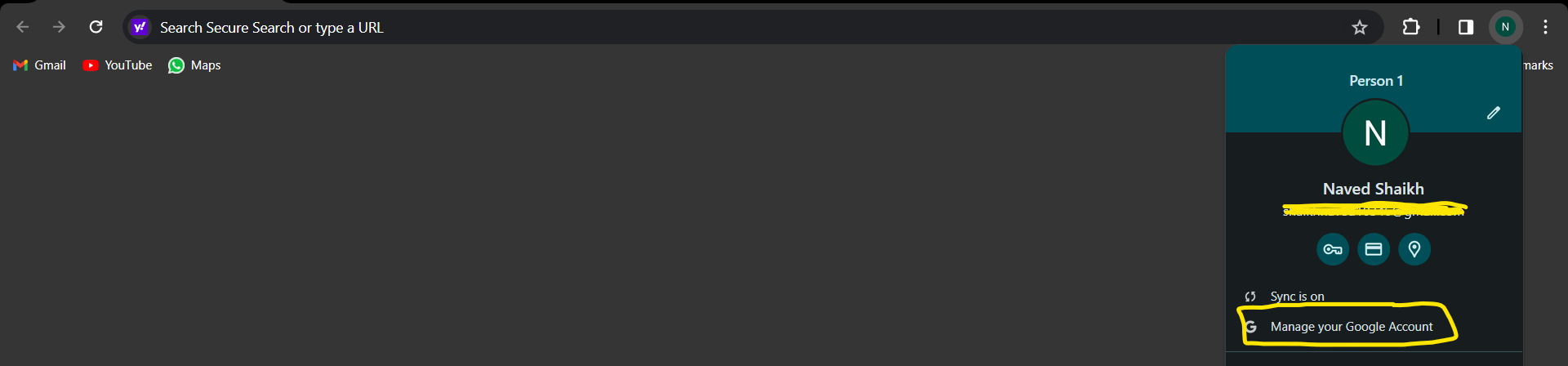
**#Disallow methods that allow anonymous authentication**

**smtp\_sasl\_security\_options = noanonymous**

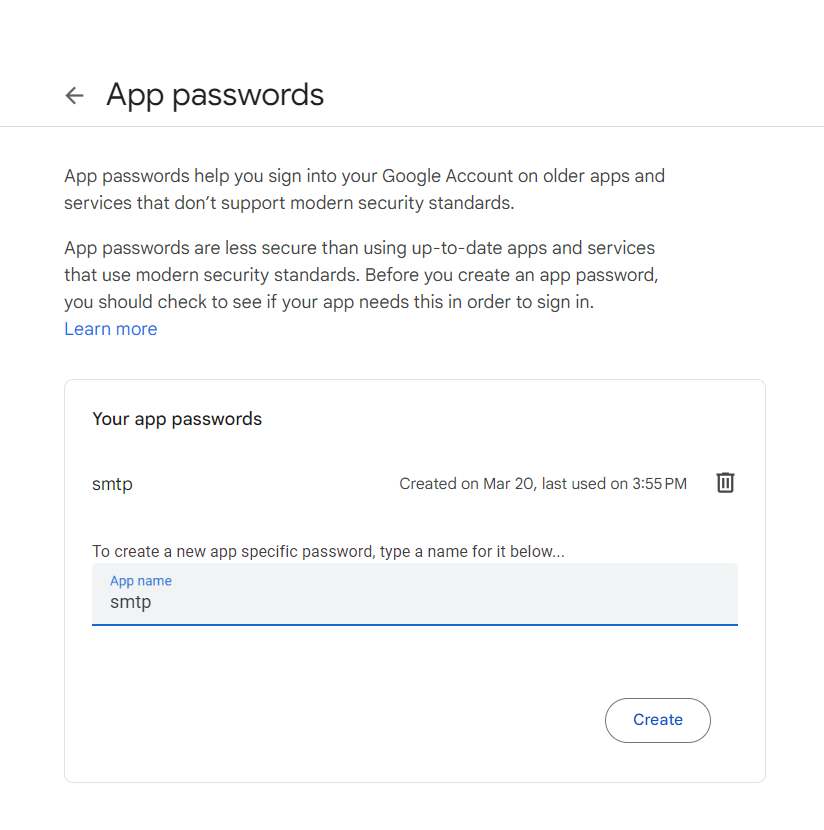
Now save this file.

Now go to cd /etc/postfix/ and make a directory by **mkdir sasl**, go in this dir by **cd sasl** and create a file inside by **touch sasl\_passwd**

Now go to you google account and click on manage your google account



Click on security in left column, enable 2 step verification and in above search bar search for apps, click on apps password, type name smtp and create one. You will get a password copy that password



Now open the sasl\_passwd file that we have created in /etc/postfix/sasl/ and add this line: **[smtp.gmail.com]:587** [**example@gmail.com:password**](mailto:example@gmail.com:password), replace [example@gmail.com](mailto:example@gmail.com) with that email you just created a password and replace the password with the password we just copied and save the file.

Convert sasl\_passwd file into db file by this command **postmap sasl\_passwd**, now you can see two files in sasl directory sasl\_passwd and sasl\_passwd.db.

**If you are unable to follow step 4 you can go through this YouTube video:** [**https://www.youtube.com/watch?v=blYx6VQEPXY**](https://www.youtube.com/watch?v=blYx6VQEPXY)

**Step 5. Schedule Cronjobs**

Cron jobs are scheduled tasks that run automatically at specified times or intervals on Unix-like operating systems such as Linux. These tasks are managed by the cron daemon, which is a background process that executes commands or scripts according to predefined schedules.

Now we need to schedule cronjobs to automatically execute the script and save the output in other file, to send emails and to delete the file after the email is sent, if we do not delete the file after the email is sent it will continue to write in the same file and the content in the file will be repetitive, so it will be difficult to understand.

To schedule the cronjob you need to type the following command **“crontab -e”** and file will be opened and now we can schedule our cronjobs over there by pressing “esc” and “i” to enter into the insert mode.

I have written the below cronjobs so you can copy paste it.

* \*/3\* \* \* \* /root/naved/AwsResourceTracker.sh >> /root/naved/usagereport.txt
* \*/3 \* \* \* \* sleep 90 && /bin/bash -c "echo 'HELLO, please find the attachment below which shows all the ongoing AWS services in your AWS account' | mutt -s 'ONGOING SERVICES IN YOUR AWS ACCOUNT' -a /root/naved/usagereport.txt – example@gmail.com"
* \*/3 \* \* \* \* sleep 105 && /bin/bash -c "rm -f /root/naved/usagereport.txt"

These cronjobs will execute in every 3 minutes but the difference is the 1st cronjob will execute in every 3 minutes.

2nd cronjob will also execute in every 3 minutes but the sleep 90 command will delay it by 90 seconds so that the first cronjob gets ample of time to execute successfully and store the output in the usagereport.txt file, you need to replace [example@gmail.com](mailto:example@gmail.com) with your email address at the end of 2nd cronjob.

The 3rd cronjob will delay by 105 seconds from first and 30 seconds from second cronjob to delete the usagereport.txt, if you want you can create a backup for usagereport.txt before it is deleted, for that you need to schedule 4th cronjob it is optional so I’m not doing it.

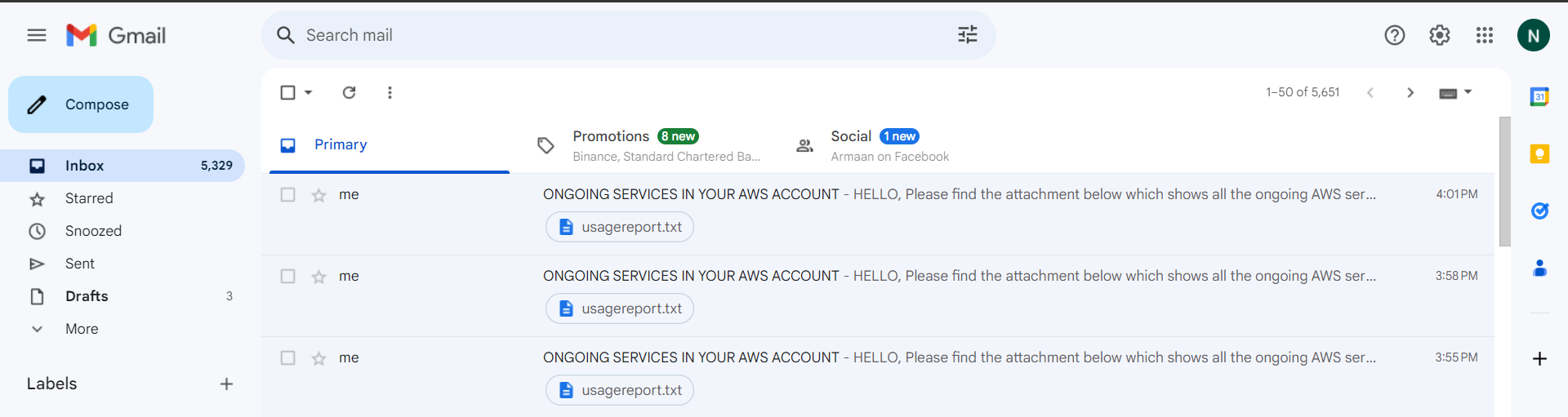
After you have pasted the above cronjobs you can save it by pressing “esc” and “:wq!” and press enter.

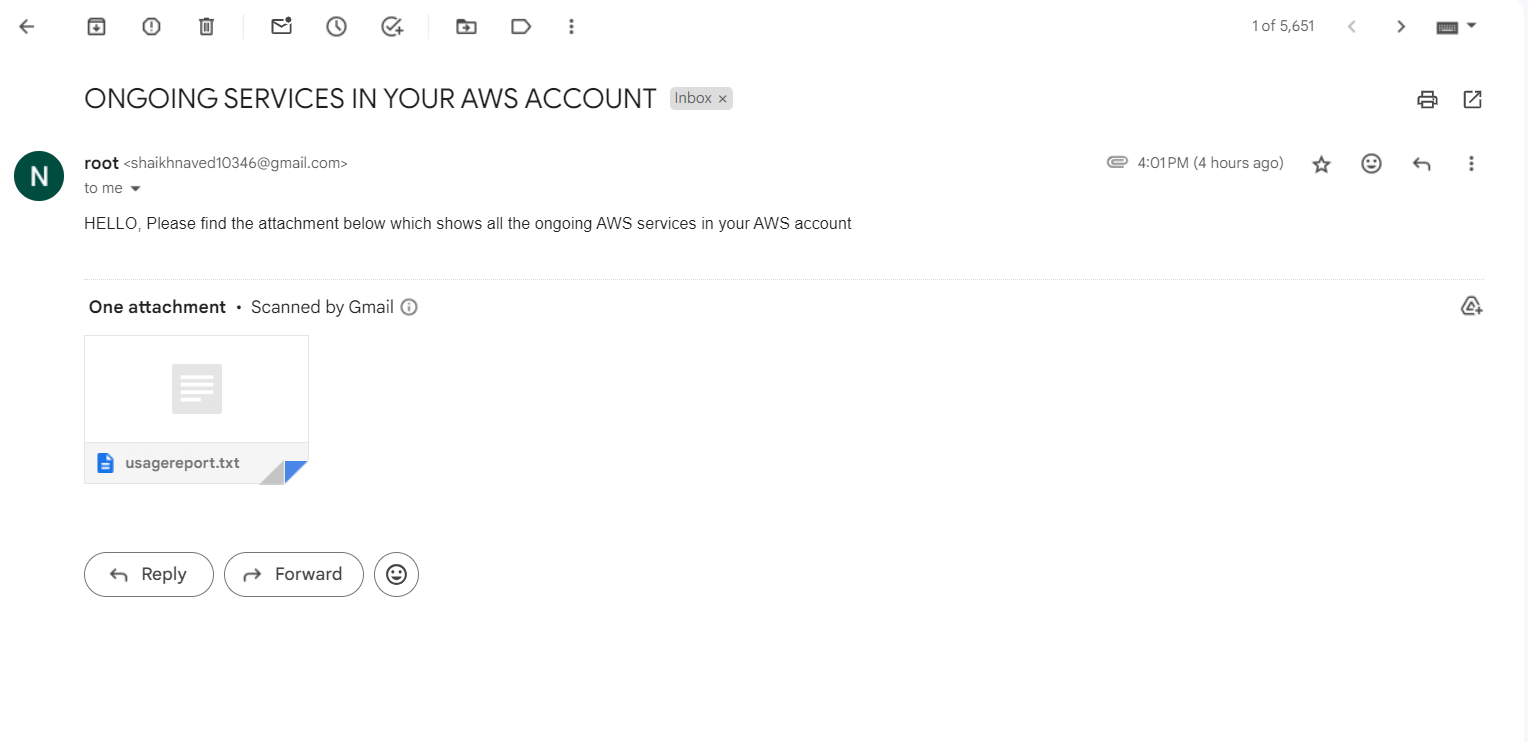
**Step 6. Start Postfix**

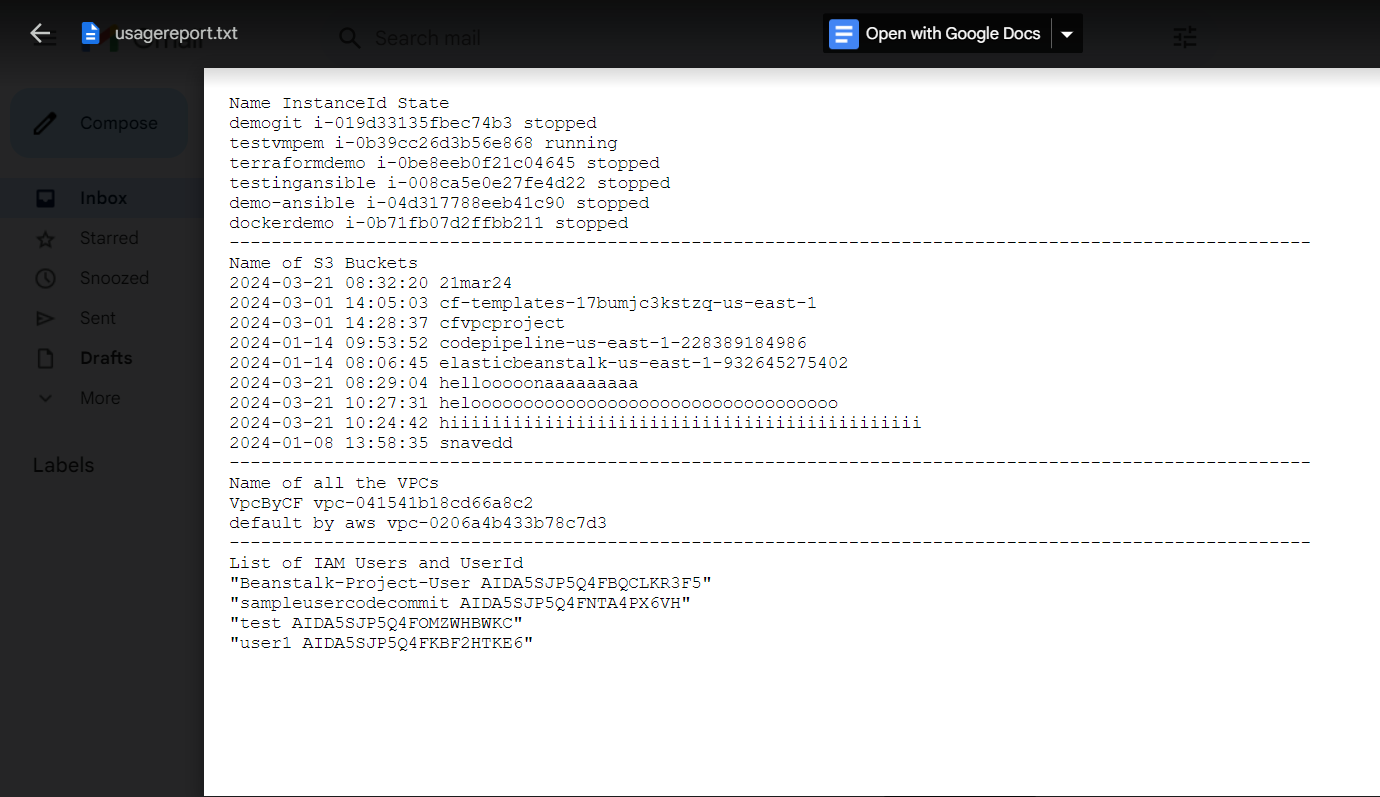
We need to start postfix to receive emails as scheduled by cronjob.

To start postfix: **systemctl start postfix**, after starting you can check status of postfix by: **systemctl status postfix.**

Once postfix is started you will start receiving emails in every 3 minutes or you can edit the time of cronjob as per your need.



****

****

As we can see above emails are received in every 3 minutes, if you want to add more aws services you can modify the script as you want and you can also modify the timing of cronjob as you want.

If you want to stop the emails you need to stop postfix by this command: **systemctl stop postfix**

By this way we can track down the ongoing aws services in our account and integrate it with cronjob.