Q1: What is eks?

Ans: Amazon Elastic Kubernetes Service (Amazon EKS) is a managed service that makes it easy for you to run Kubernetes on AWS without needing to stand up or maintain your own Kubernetes control plane.

Amazon EKS runs Kubernetes control plane instances across multiple Availability Zones to ensure high availability. Amazon EKS automatically detects and replaces unhealthy control plane instances, and it provides automated version upgrades and patching for them.



Getting started with Amazon EKS is easy:

1. First, create an Amazon EKS cluster in the AWS Management Console or with the AWS CLI or one of the AWS SDKs.

2. Then, launch worker nodes that register with the Amazon EKS cluster. We provide you with an AWS Cloud Formation template that automatically configures your nodes.

3. When your cluster is ready, you can configure your favorite Kubernetes tools (such as **kubectl**) to communicate with your cluster.

4. Deploy and manage applications on your Amazon EKS cluster the same way that you would with any other Kubernetes environment.

# Benefits:-

# 1) High Availability

# 2) Server less option

# 3) Secure

# 4) Built with the Community

# ****Task:-****

For this task, we are going to deploy our pods of Wordpress and a Mysql database on the top of Amazon EKS cluster.

Basically, what we are trying to do is:

: Create a user with admin access

: Create a cluster using eksctl.

: Integrate EKS with EC2,ELB,EBS,EFS.

: Create an EFS that has the ability to attach more than one container/pods.

: Create deployment of the application Word Press and mysql

: Create a secret that contains wordpress root and mysql db password.

**Prerequisite:-**

1) AWS account

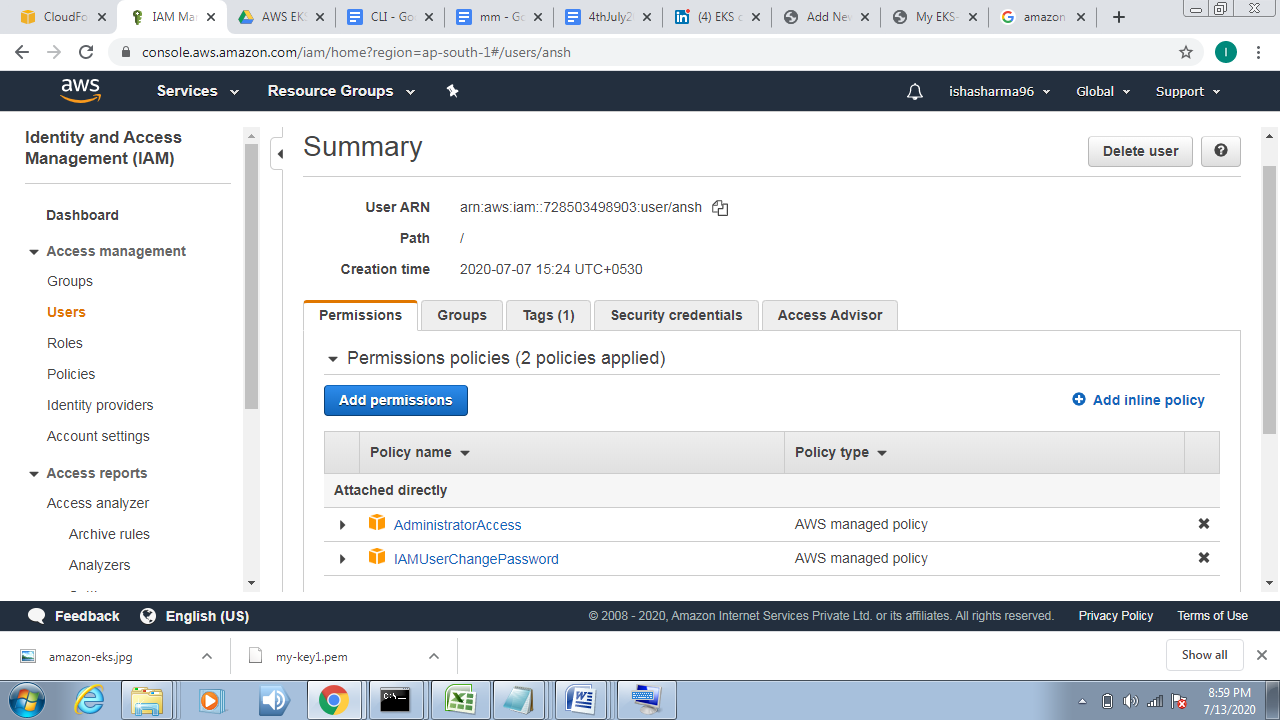
2) AWS CLI

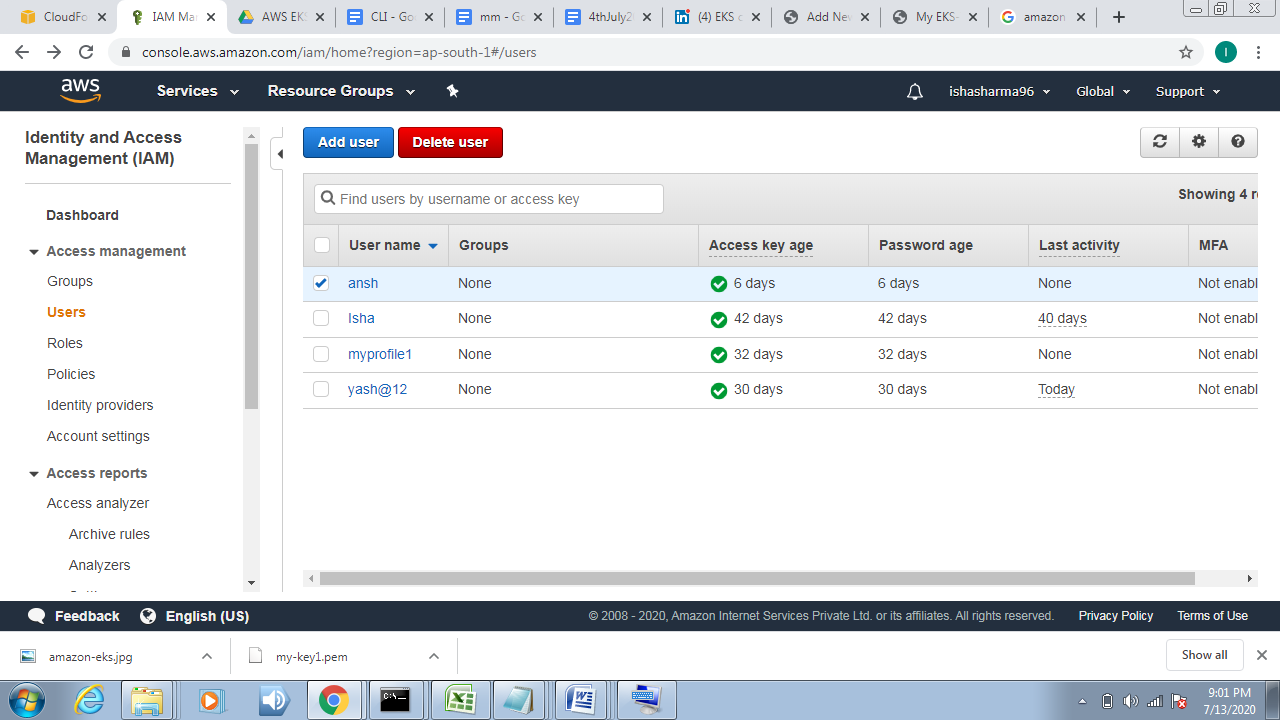
3) Eksctl

4) Basic of k8s

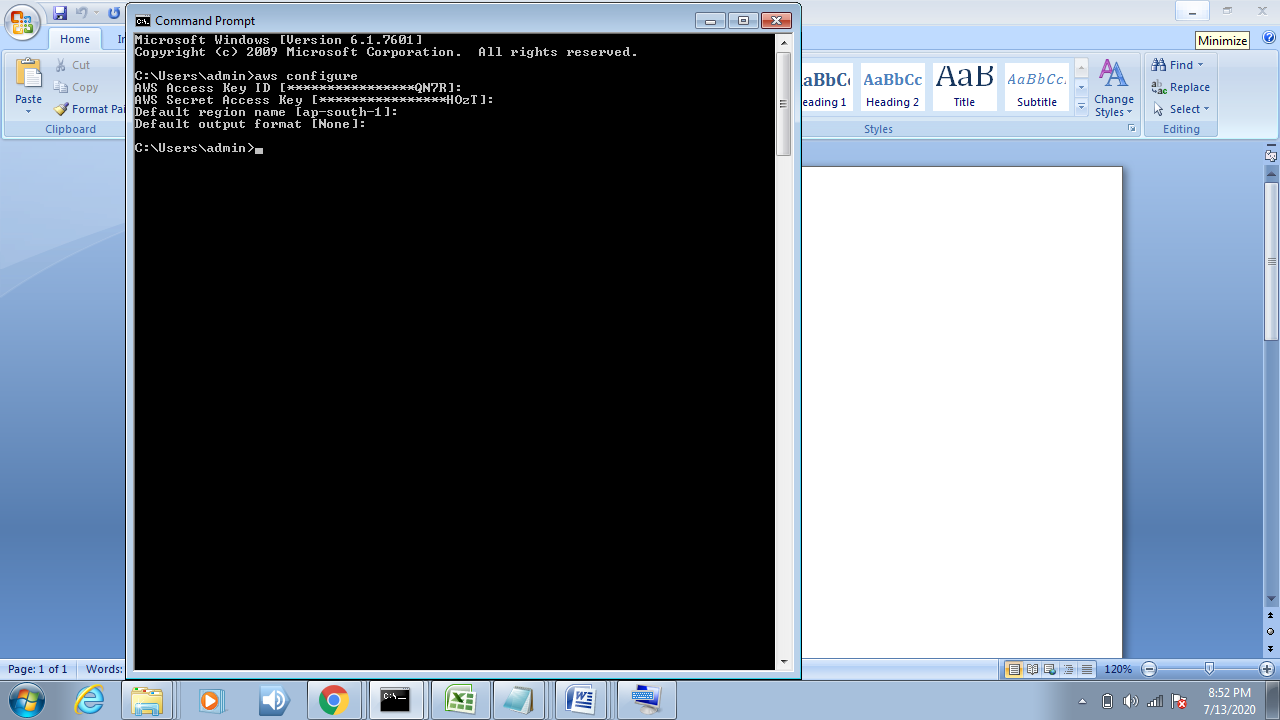
5) Kubectl

Step1:create iam user having administration power.

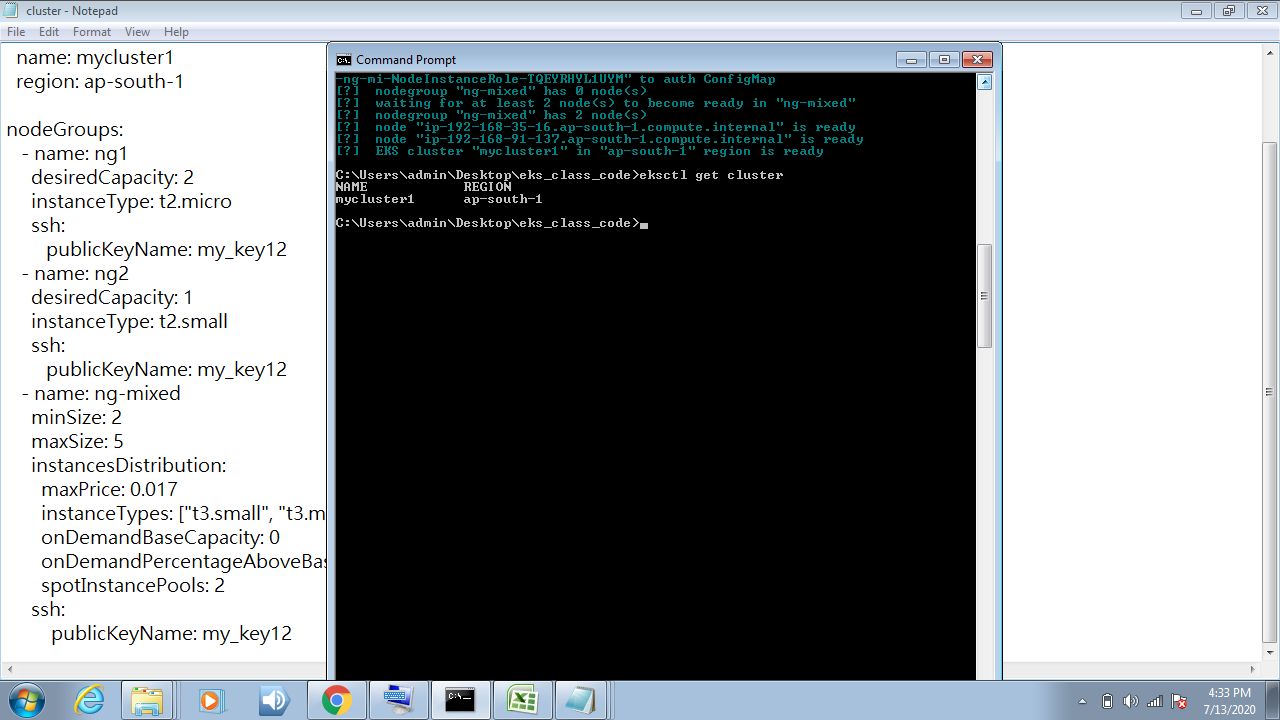




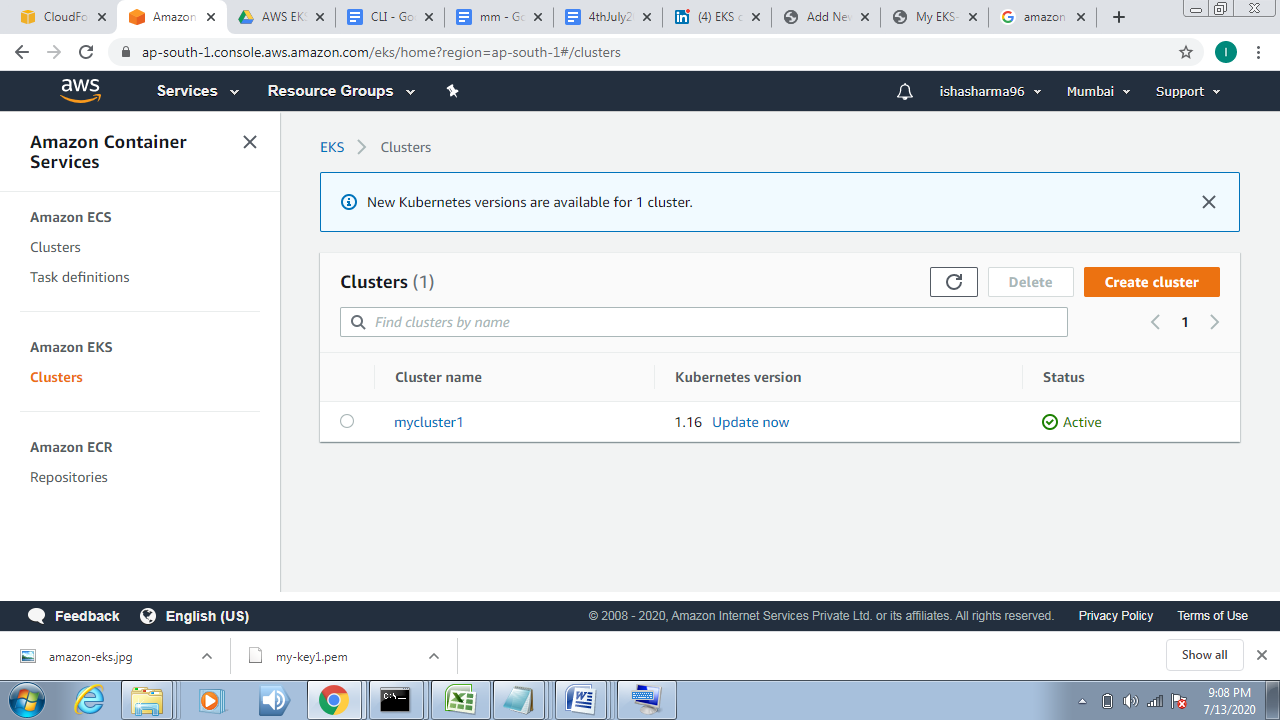
Step:2 configure the aws:



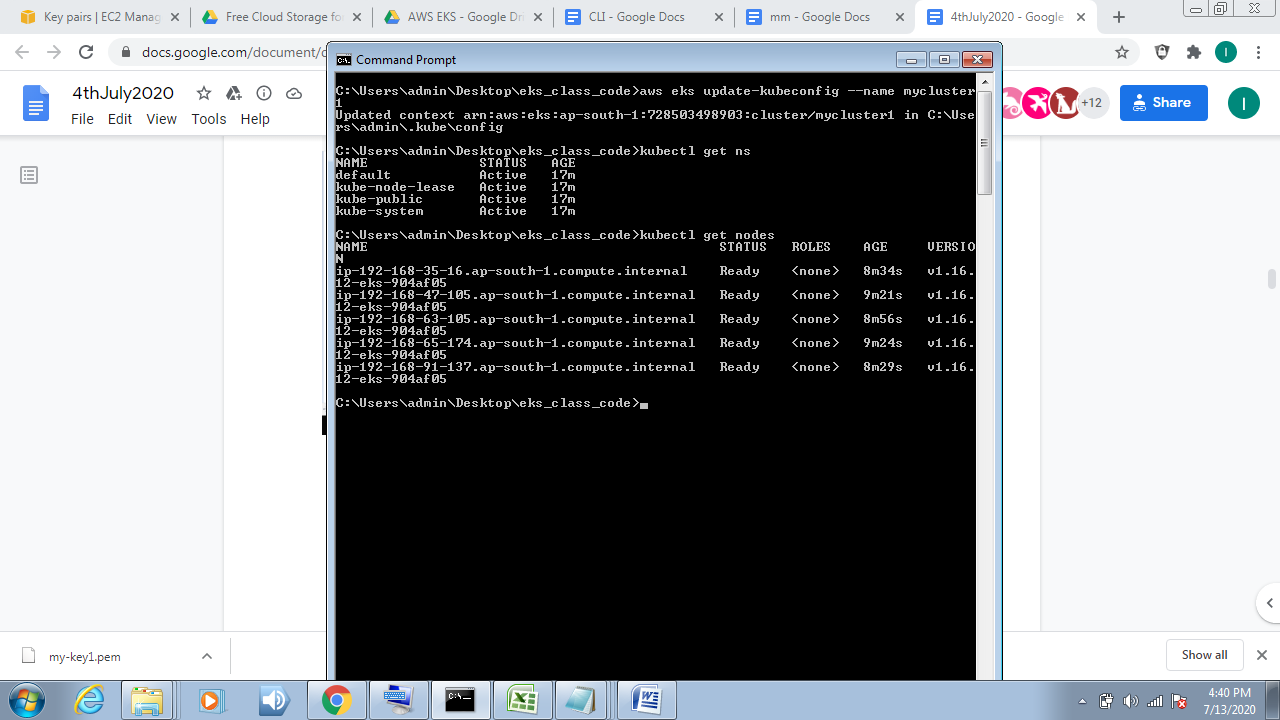
Step3: create cluster :



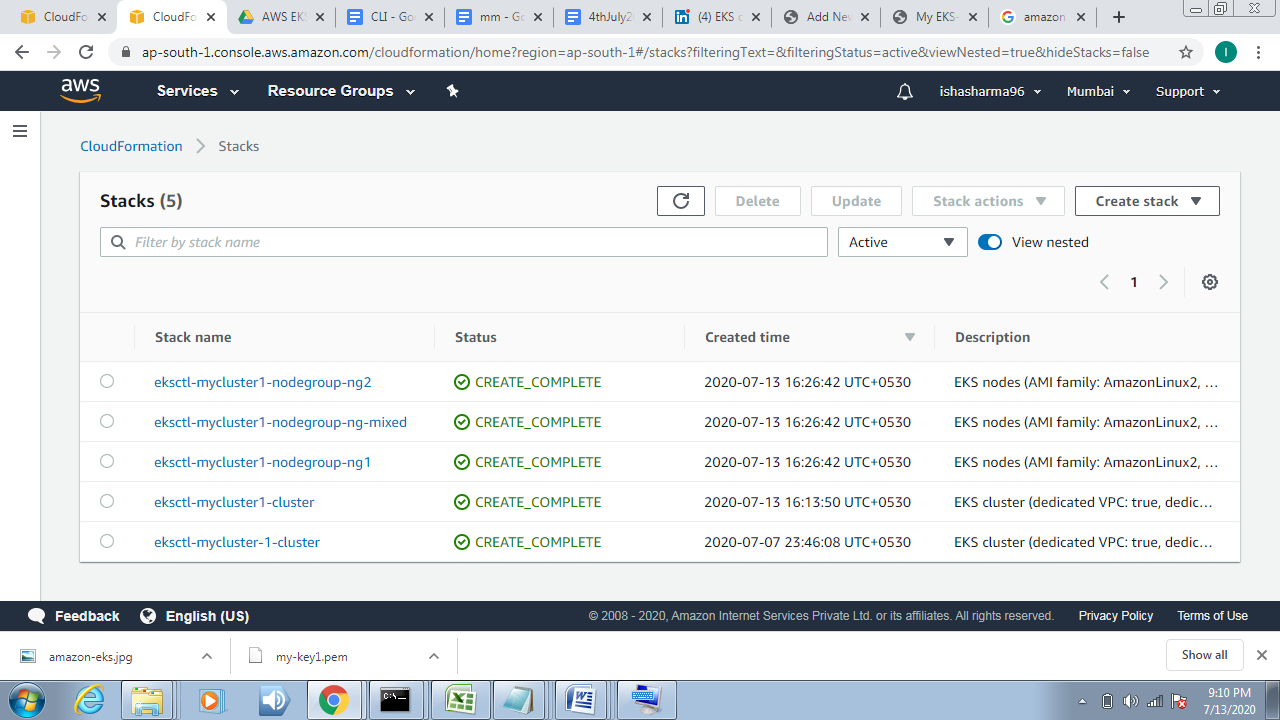
Cluster has been created



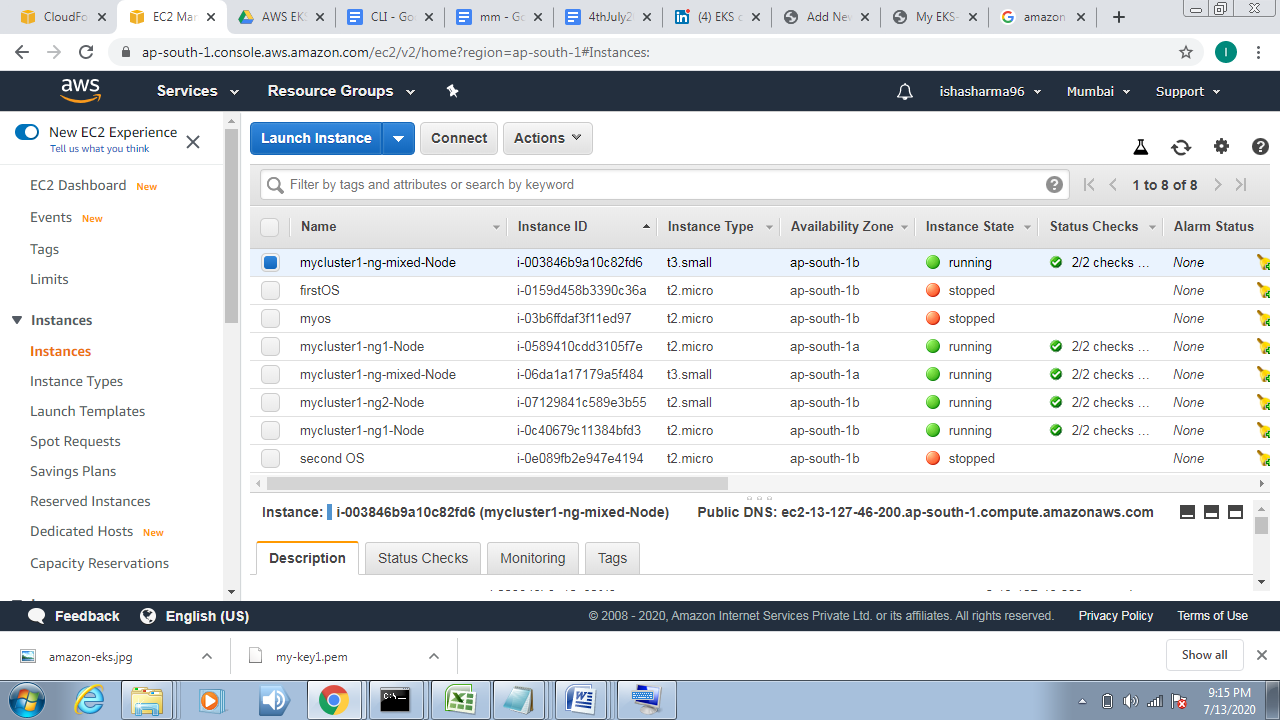
Step4: update config file



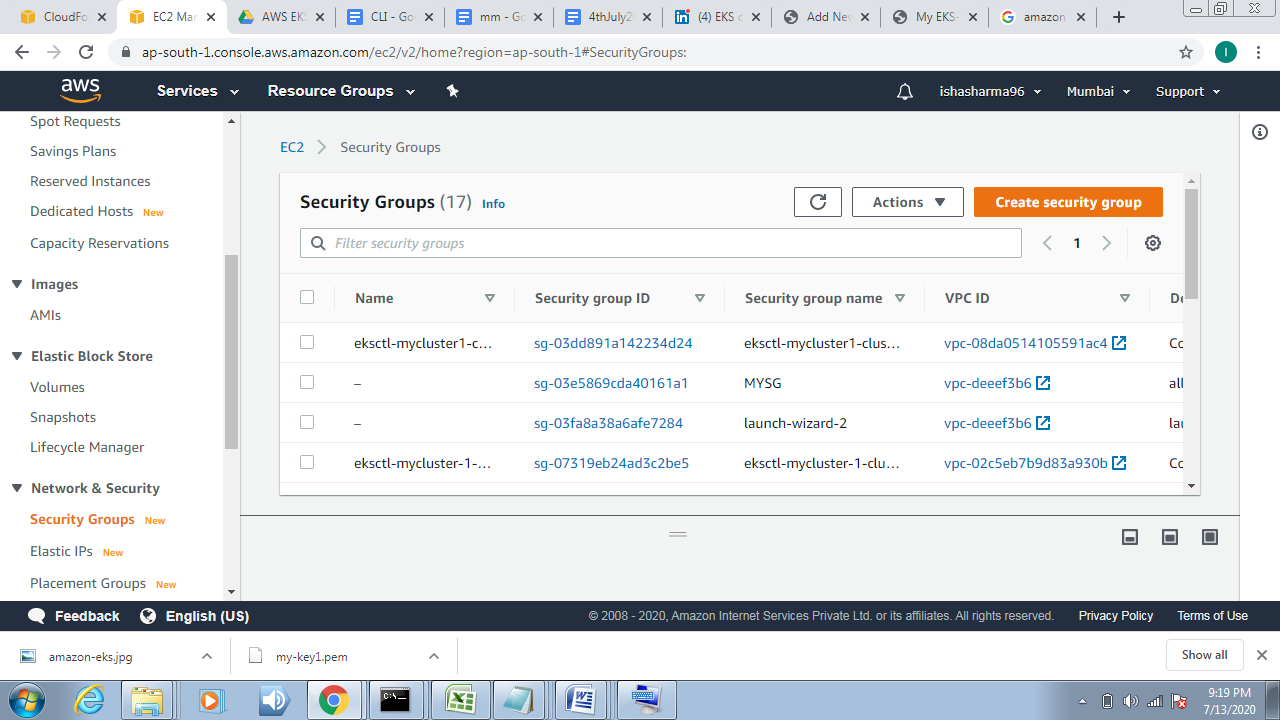
# cloudformation>stack:



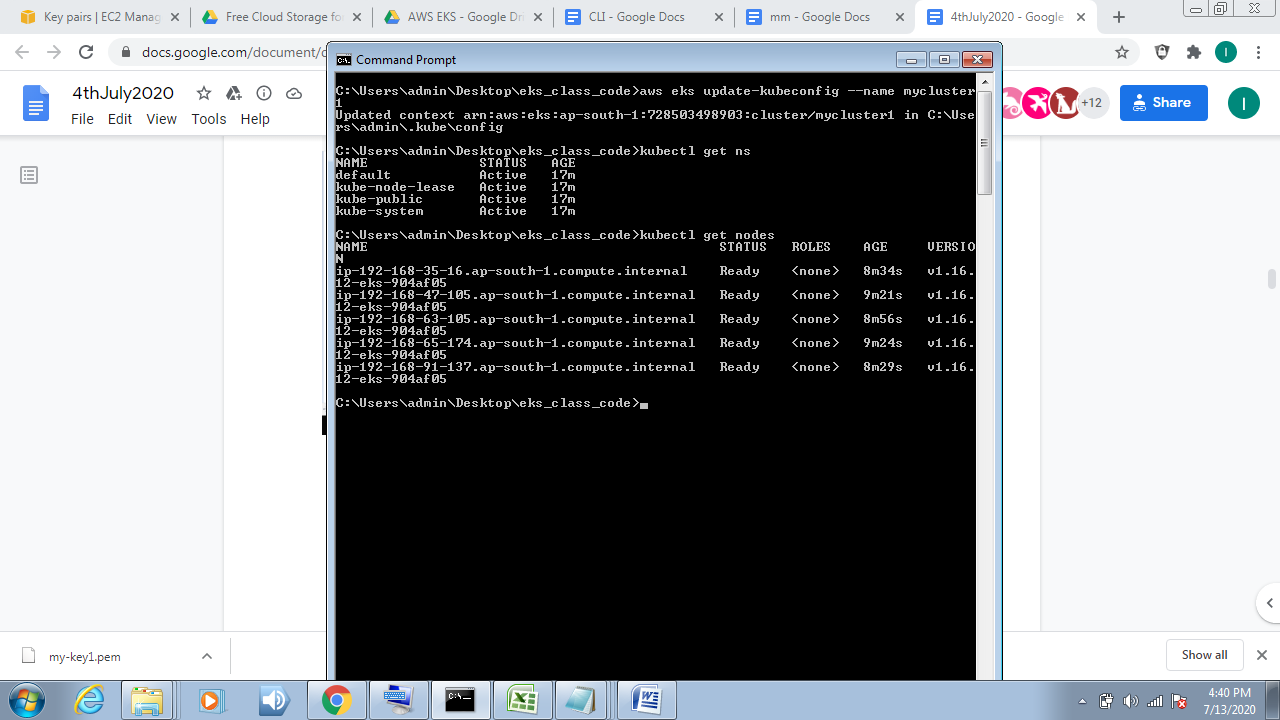
#Instance created.

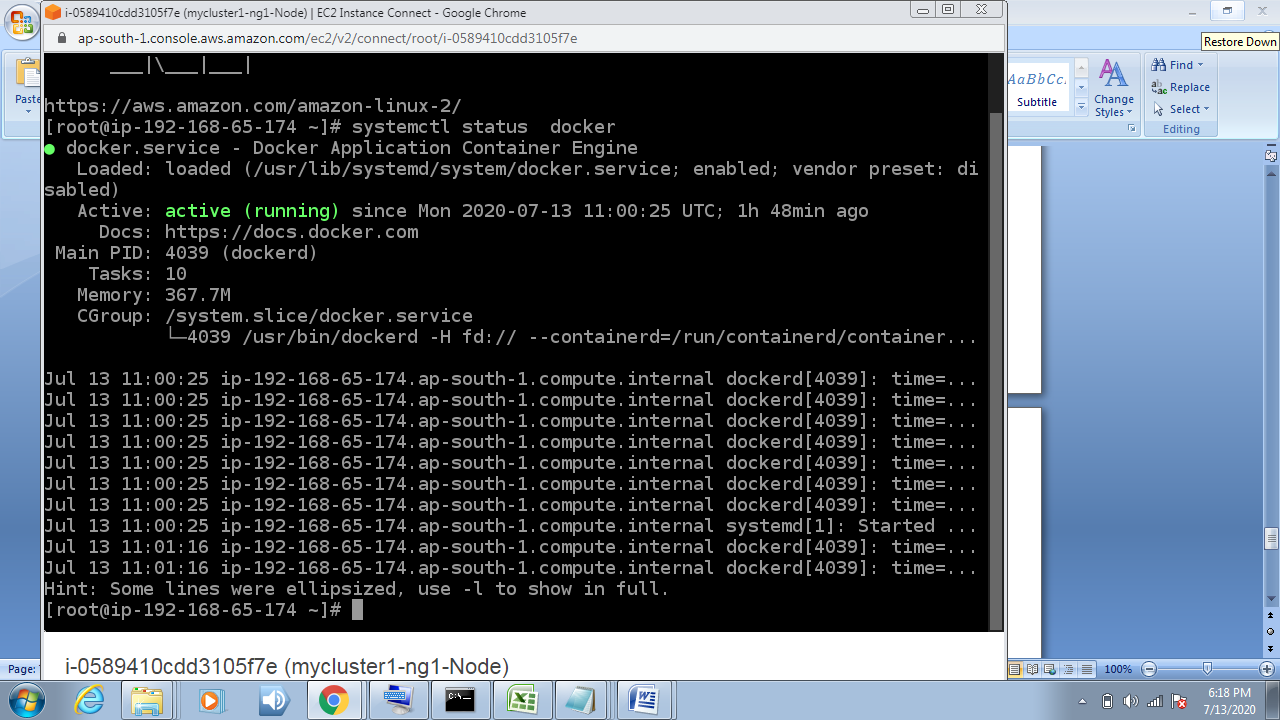


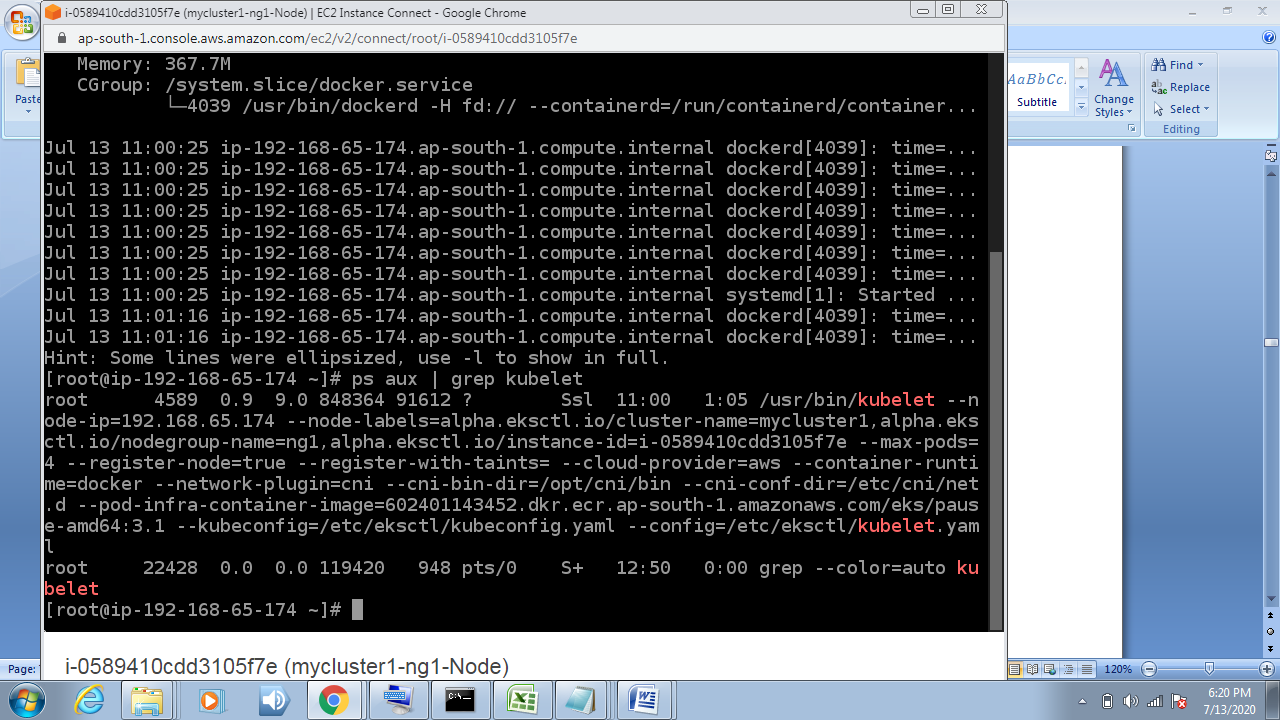
# Security group:



To get nodes command kubectl get nodes:





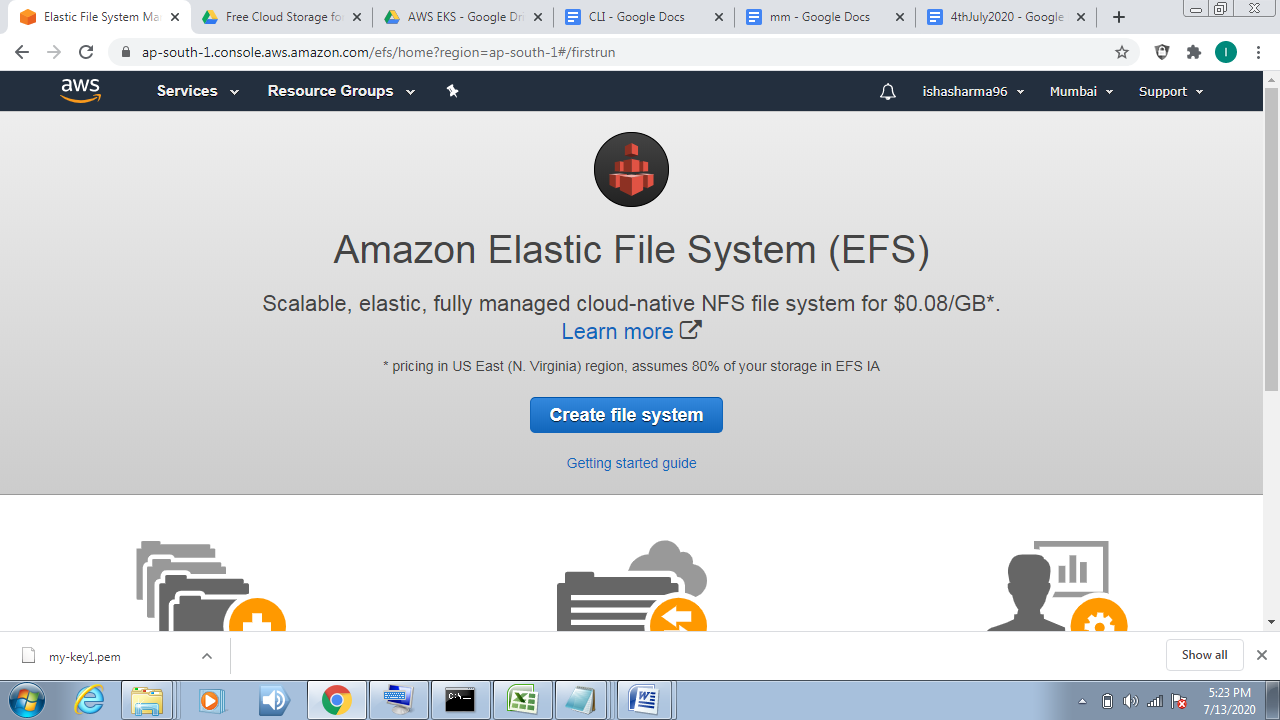


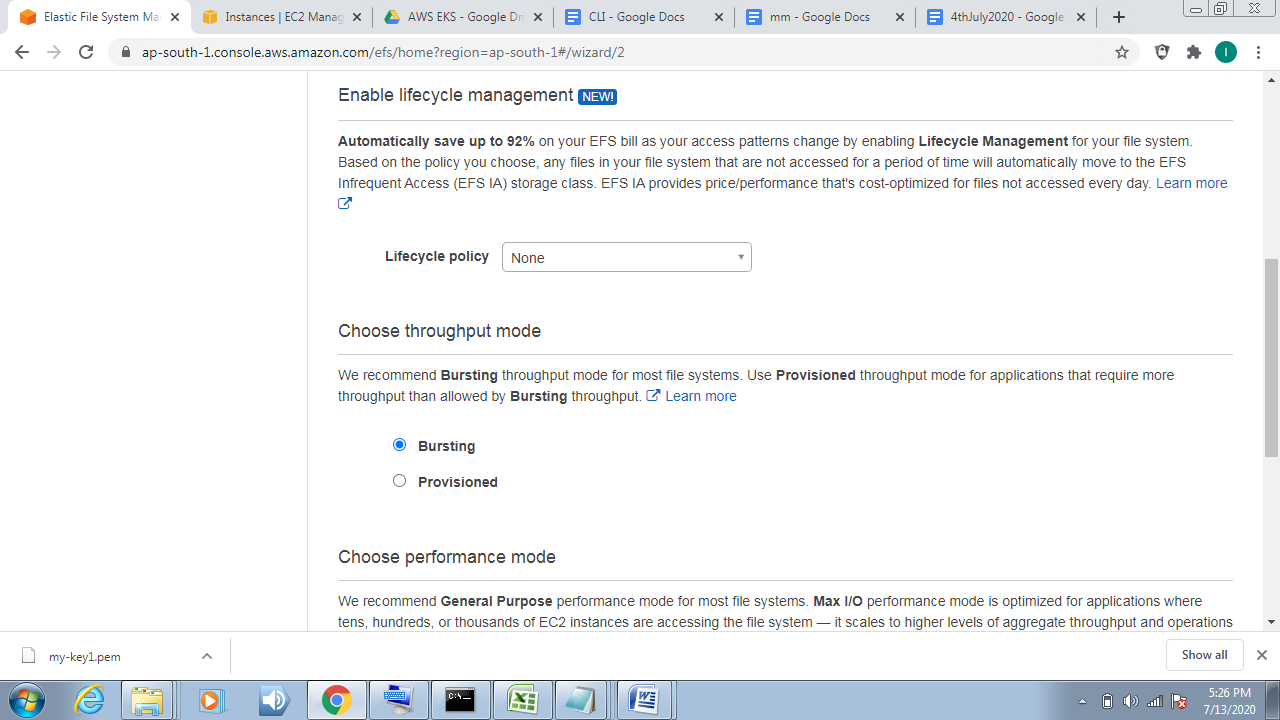
**Step7. Going to create EFS .**

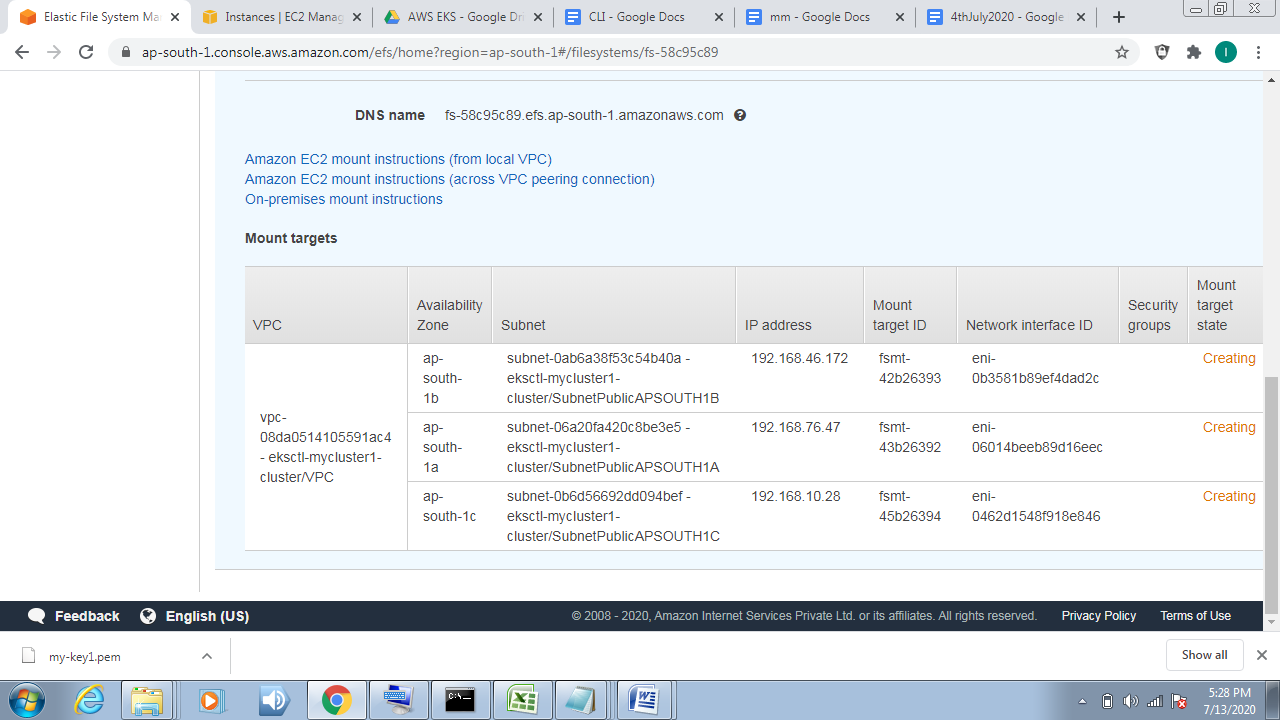
**“Amazon Elastic File System (Amazon EFS) provides a simple, scalable,fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.”**

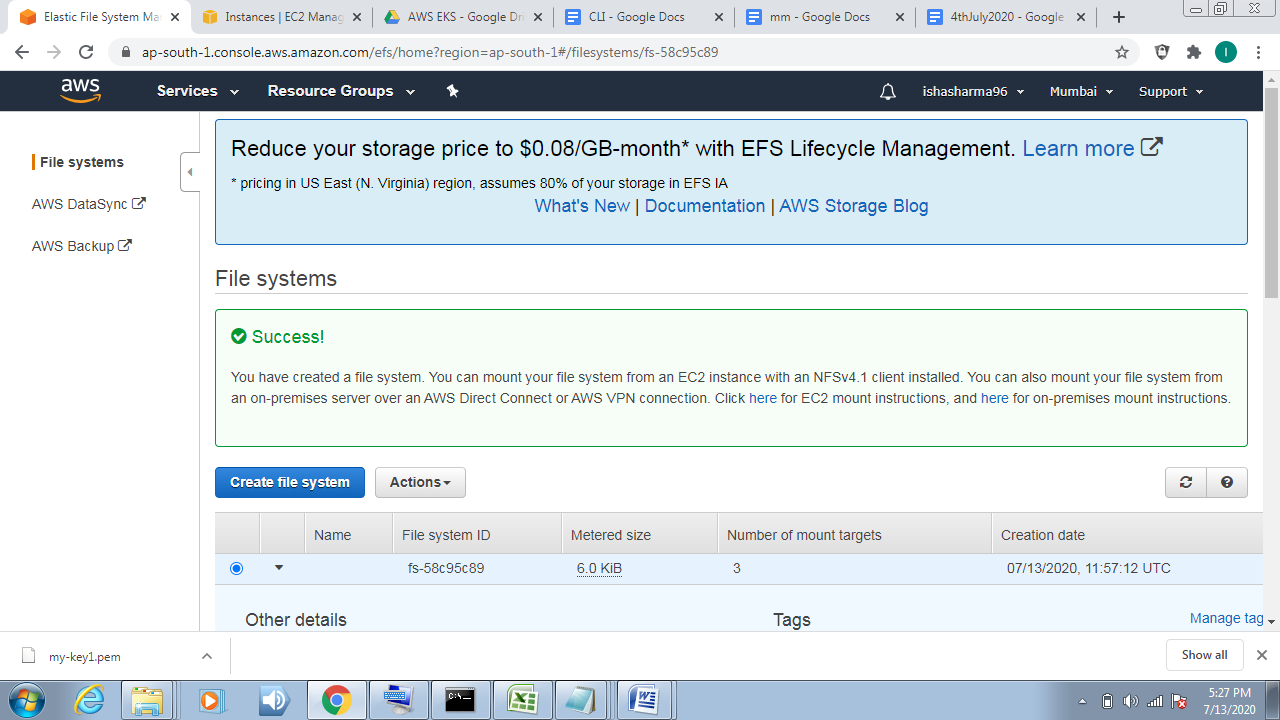
aws console -> Services -> EFS -> Create EFS

In this you have to config security group very carefully ,first choose the VPC of your cluster then choose the security group accordingly. Do not go for default VPC.









**Step 8. Create mysql pod (**for this we create mysql-deployment.yaml file)

**CODE:-**

apiVersion: v1

kind: Service

metadata:

name: wordpress-mysql

labels:

app: wordpress

spec:

ports:

- port: 3306

selector:

app: wordpress

tier: mysql

clusterIP: None

---

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: mysql-pv-claim

labels:

app: wordpress

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 10Gi

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: wordpress-mysql

labels:

app: wordpress

spec:

selector:

matchLabels:

app: wordpress

tier: mysql

strategy:

type: Recreate

template:

metadata:

labels:

app: wordpress

tier: mysql

spec:

containers:

- image: mysql:5.6

name: mysql

env:

- name: MYSQL\_ROOT\_PASSWORD

valueFrom:

secretKeyRef:

name: mysql-pass

key: password

ports:

- containerPort: 3306

name: mysql

volumeMounts:

- name: mysql-persistent-storage

mountPath: /var/lib/mysql

volumes:

- name: mysql-persistent-storage

persistentVolumeClaim:

claimName: mysql-pv-claim

**Step 9. Create WordPress pod (**for this we create wordpress-deployment.yaml**)**

**CODE:-**

apiVersion: v1

kind: Service

metadata:

name: wordpress

labels:

app: wordpress

spec:

ports:

- port: 80

selector:

app: wordpress

tier: frontend

type: LoadBalancer

---

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: wp-pv-claim

labels:

app: wordpress

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 5Gi

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: wordpress

labels:

app: wordpress

spec:

selector:

matchLabels:

app: wordpress

tier: frontend

strategy:

type: Recreate

template:

metadata:

labels:

app: wordpress

tier: frontend

spec:

containers:

- image: wordpress:4.8-apache

name: wordpress

env:

- name: WORDPRESS\_DB\_HOST

value: wordpress-mysql

- name: WORDPRESS\_DB\_PASSWORD

valueFrom:

secretKeyRef:

name: mysql-pass

key: password

ports:

- containerPort: 80

name: wordpress

volumeMounts:

- name: wordpress-persistent-storage

mountPath: /var/www/html

volumes:

- name: wordpress-persistent-storage

persistentVolumeClaim:

claimName: wp-pv-claim

**Step 10. Now both wordpress\_deployment and mysql-deployment has been created . Now we shall create kustomization.yml file as it let us deploy whole setup with just one command and few other functionalities.**

**Kustomization.yml**

apiVersion: kustomize.config.k8s.io/v1beta1

kind: Kustomization

secretGenerator:

- name: mysql-pass

literals:

- password=redhat

resources:

- mysql-deployment.yaml

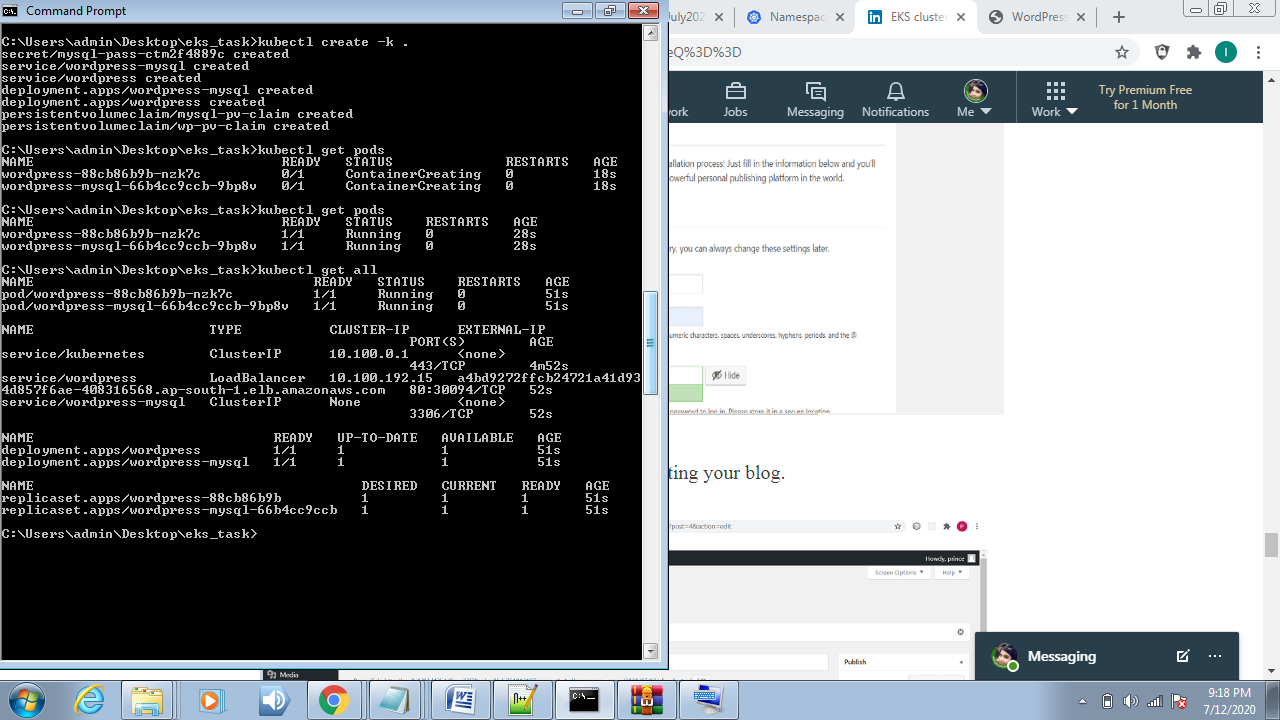
- wordpress-deployment.yaml

**Step 11. Deploy the whole setup and run these commands:-**

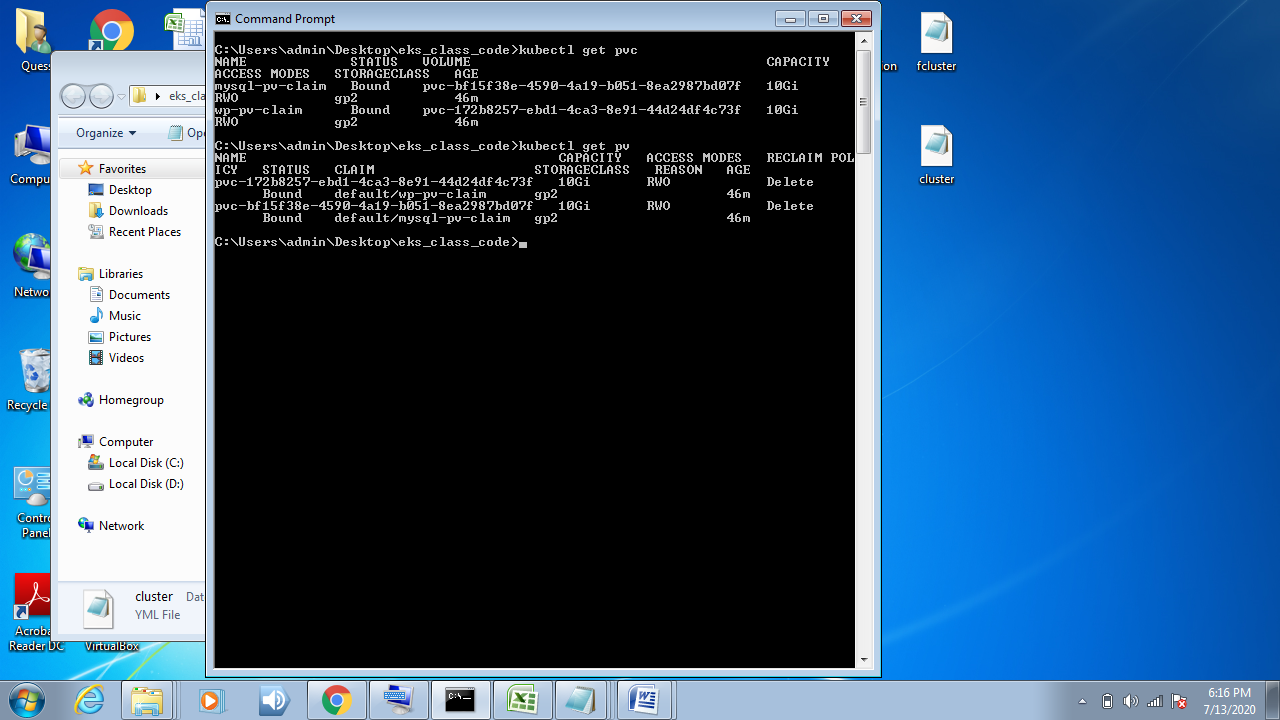
Ø **kubectl create –k .**

Ø **kubectl get pods //to show all pods**

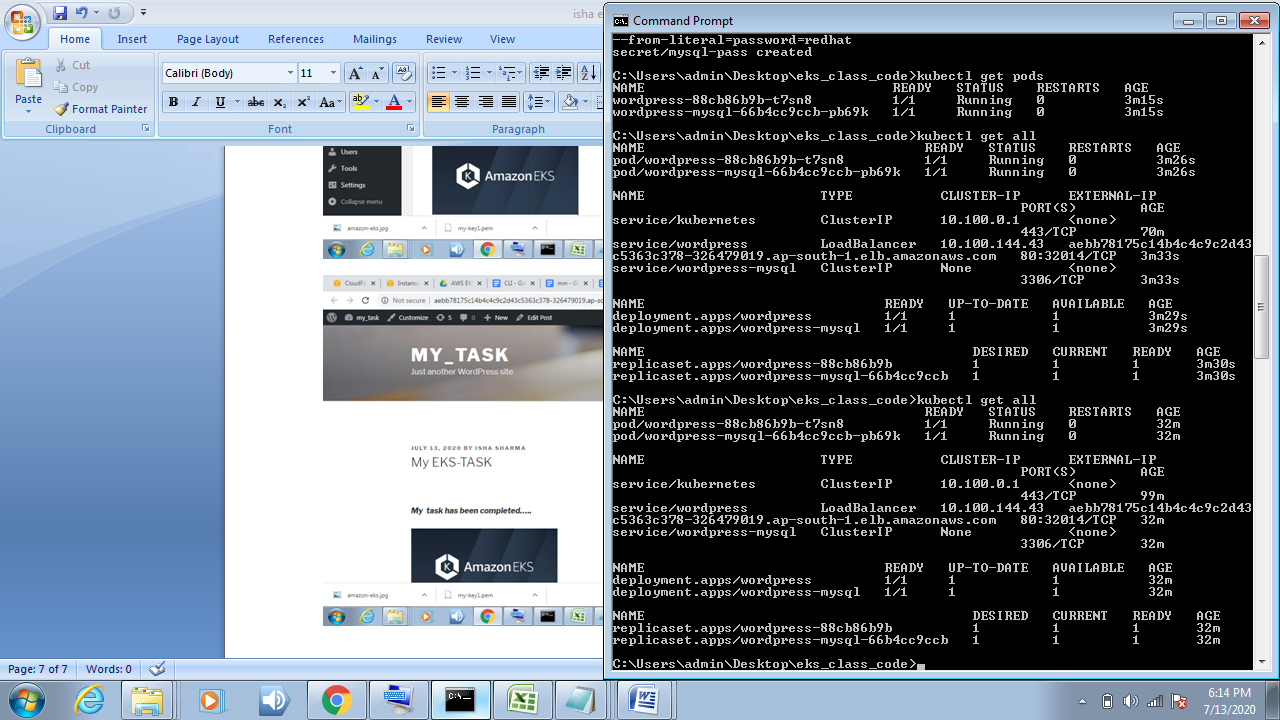
Ø **kubectl get all //provide all details**

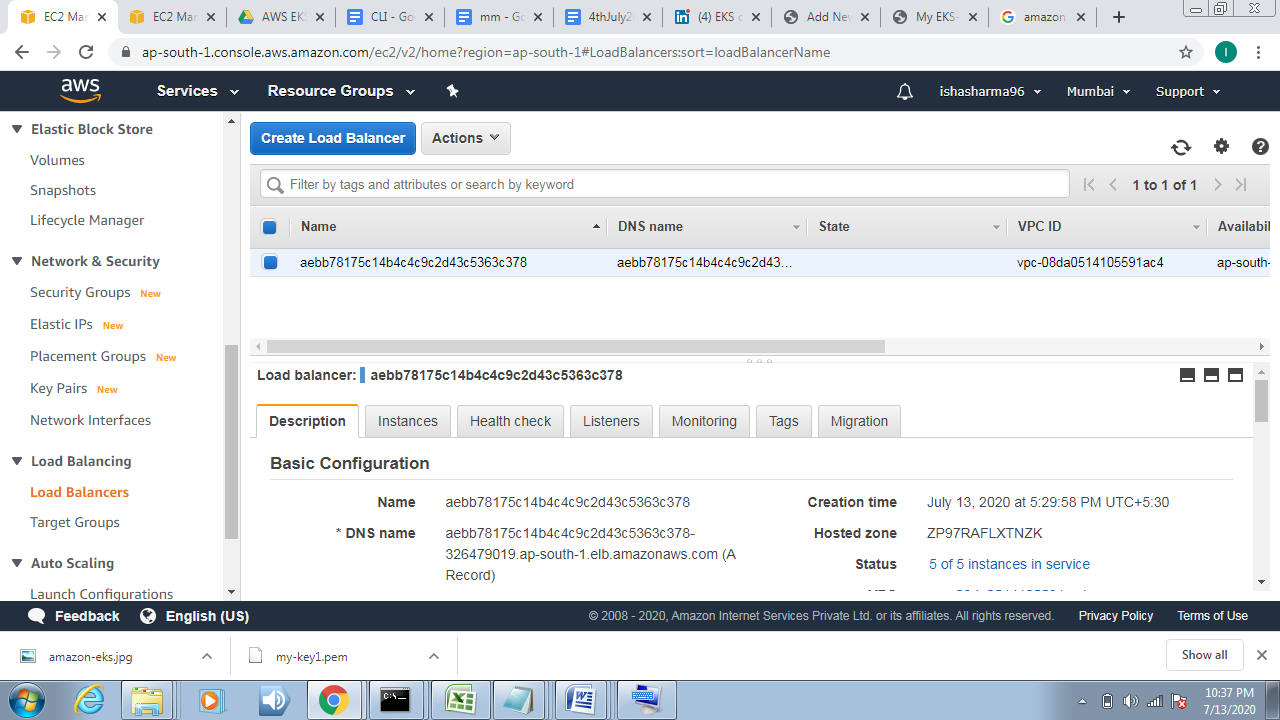


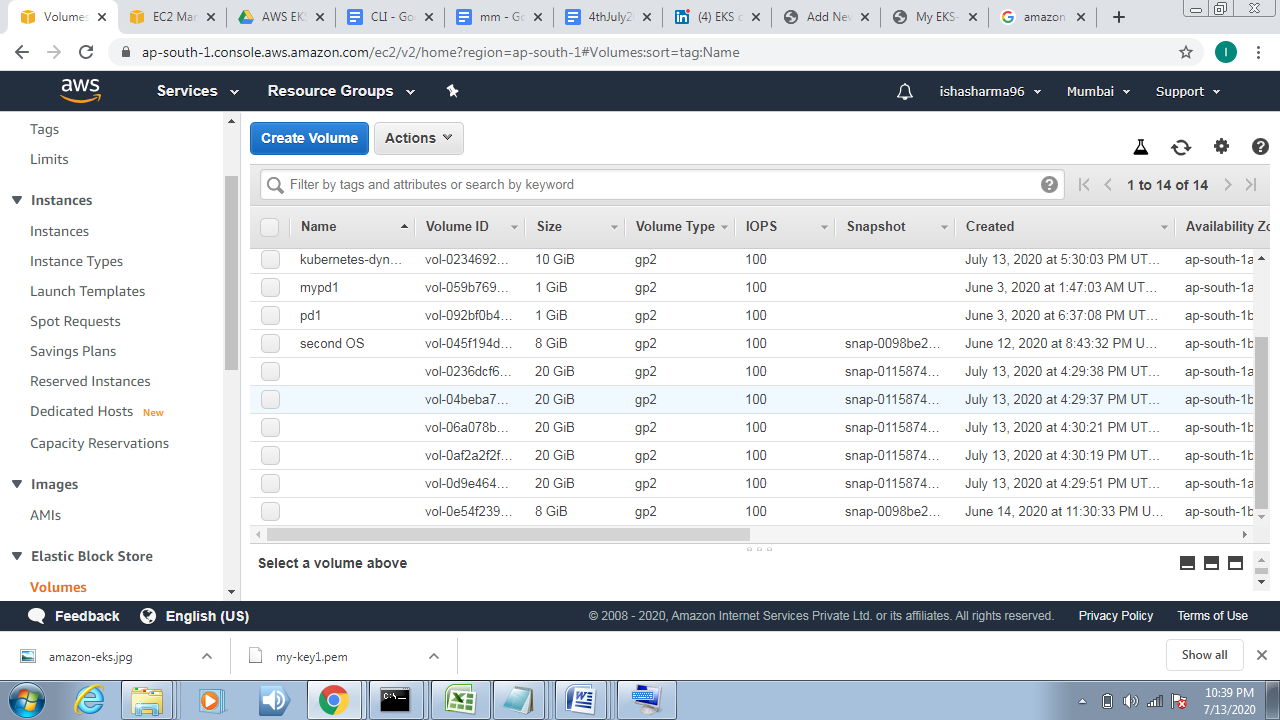
**To check pvc and pv by using CLI Command →**kubectl get pvc ,and kubectl get pv



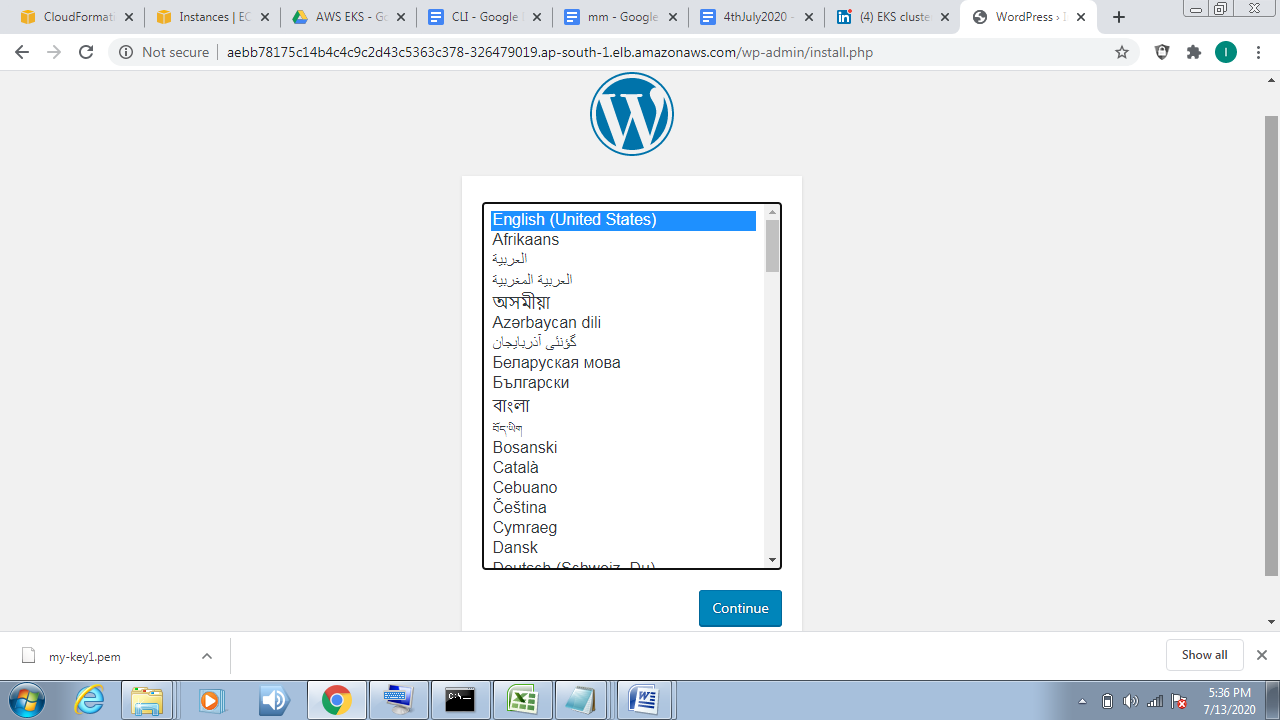
Load balancer:

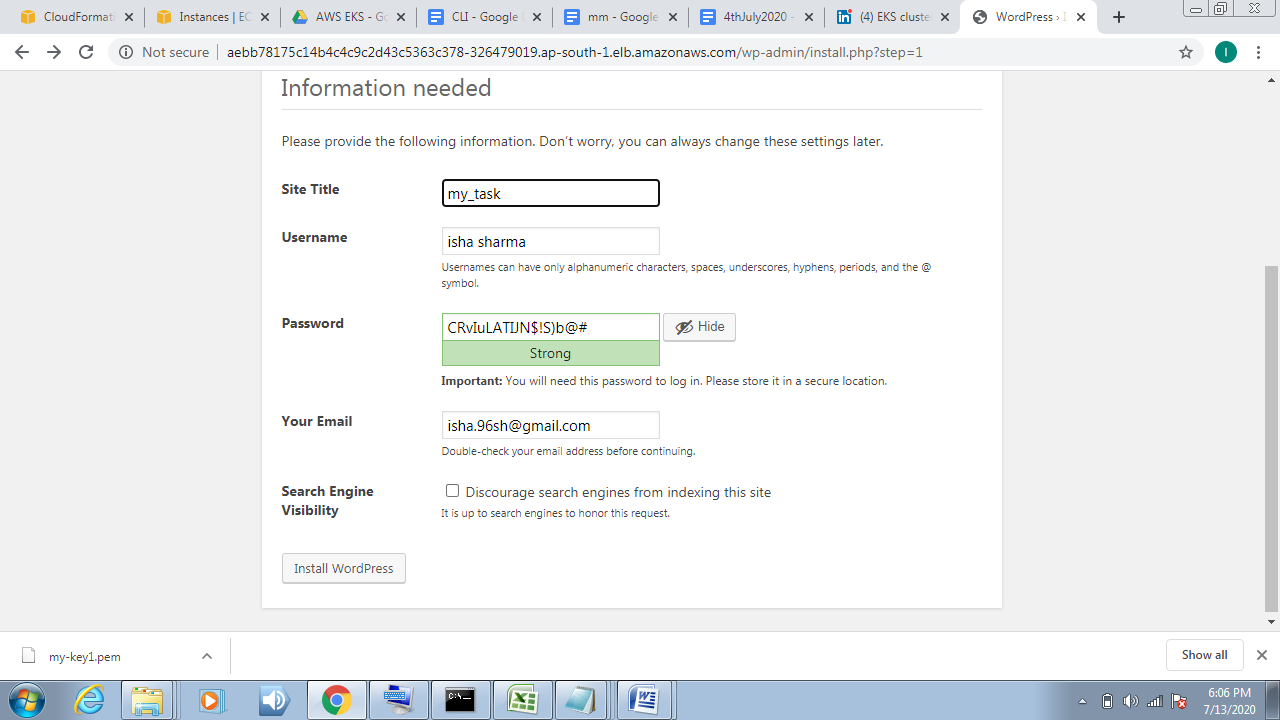


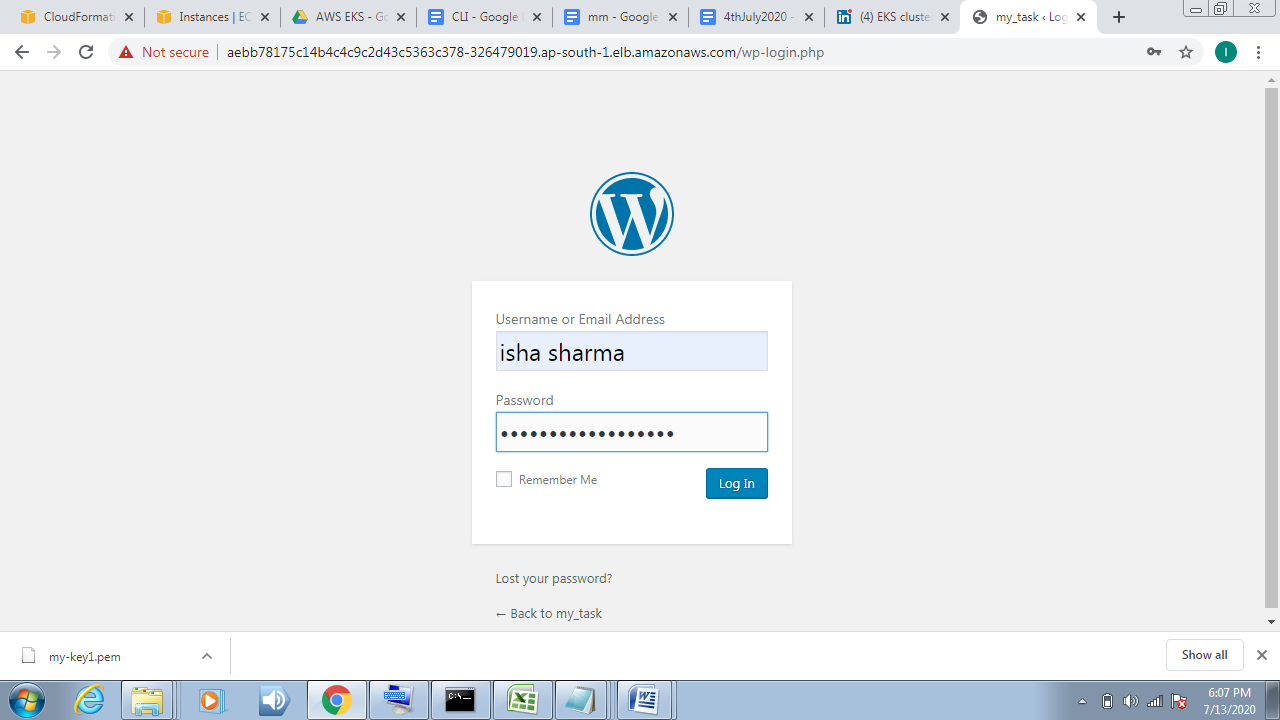
  
Volume:

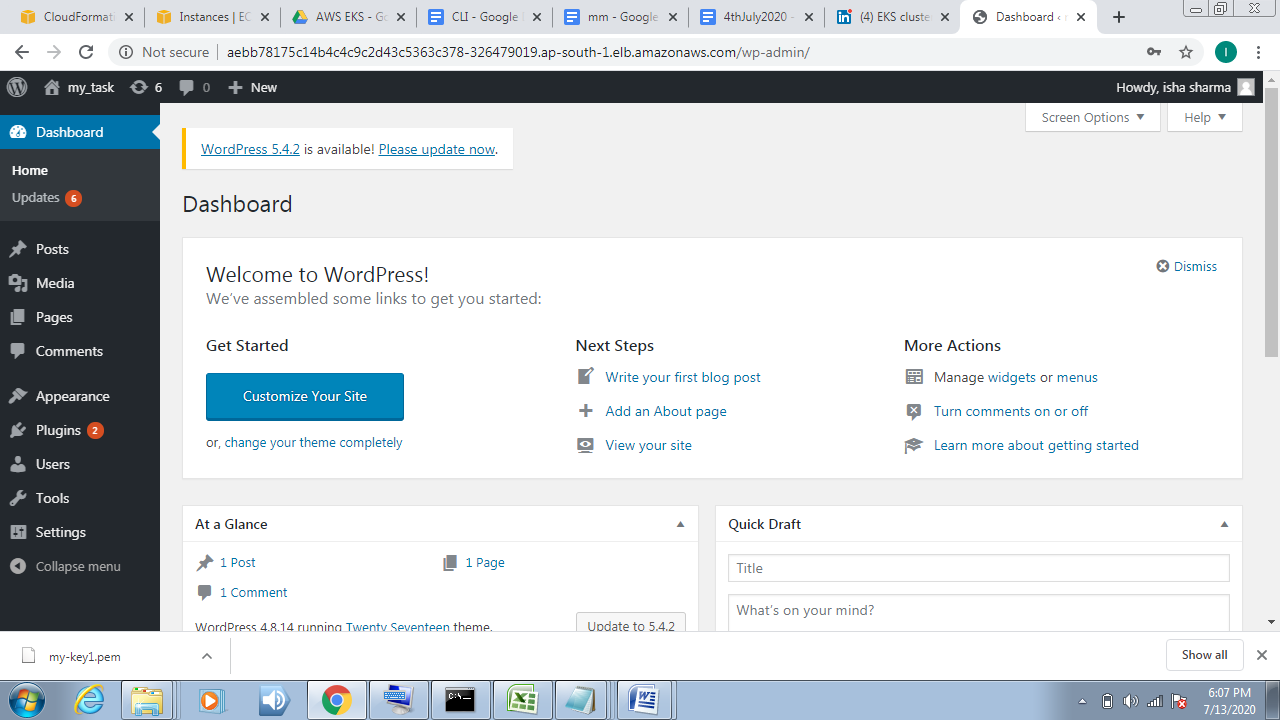


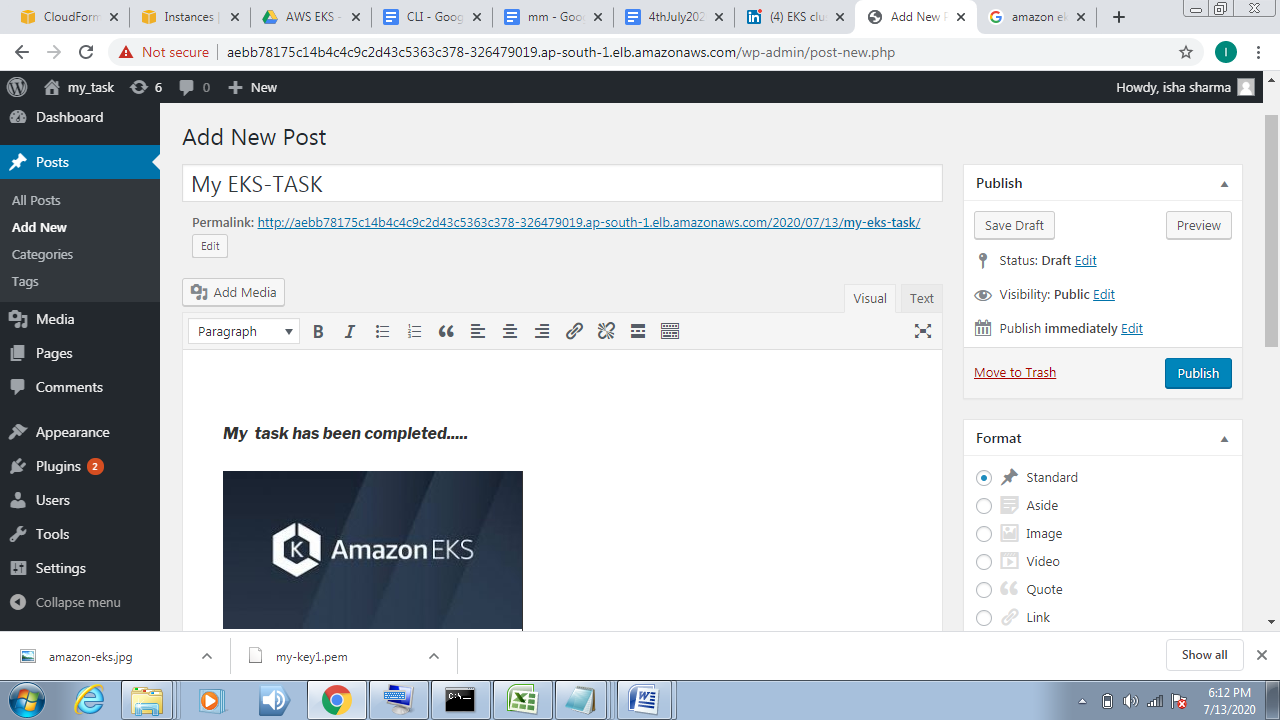
**Step 12 .Now copy the external ip (**get it from command kubectl get all**) and paste it into your browser.**

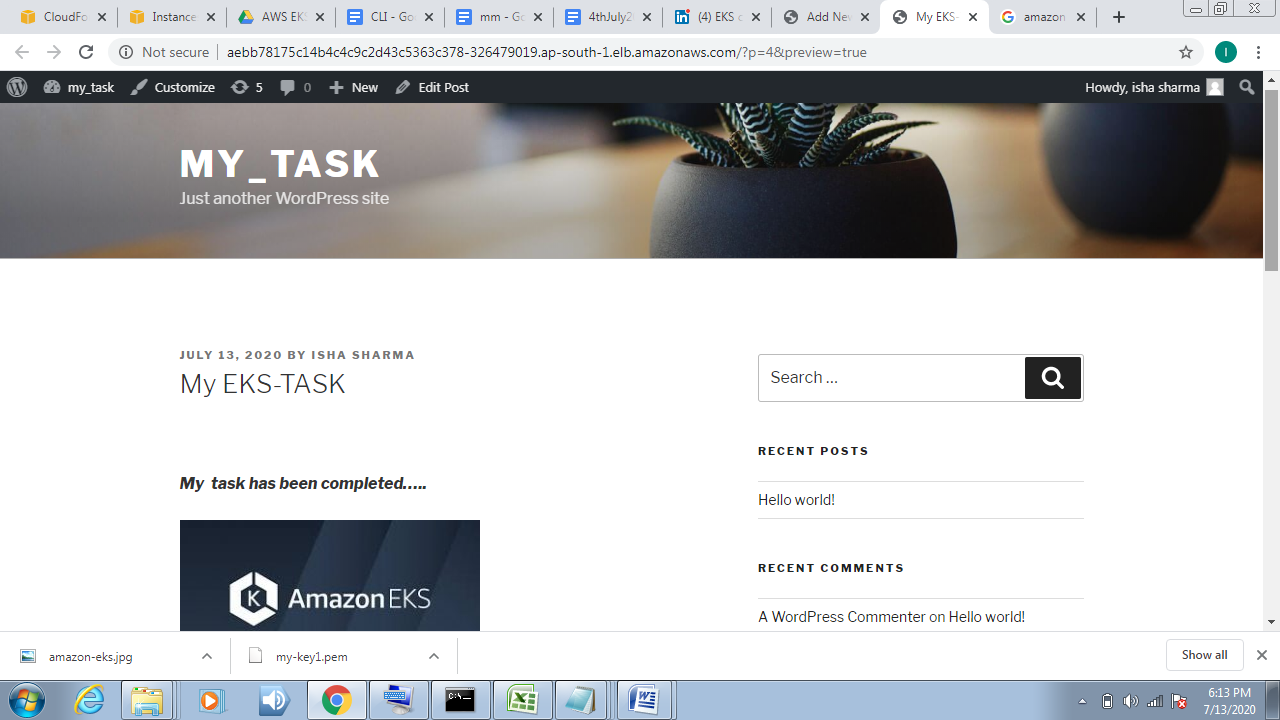












**In last check your bill and delete your cluster by CLI using command -> “**eksctl delete cluster –f cluster.yml**”**

**task has been completed. thanks to vimal daga sir for guidances us.**