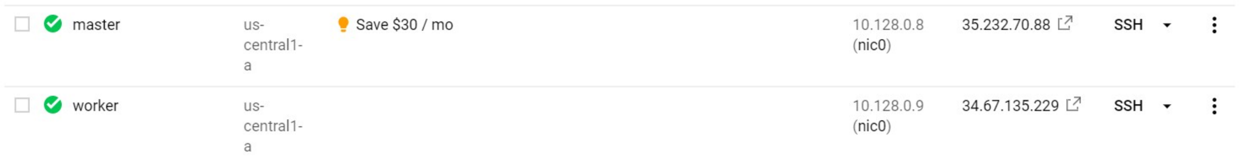
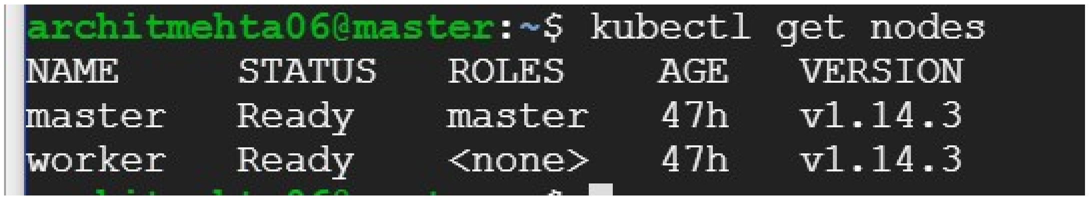
Tasks:

1. Create a Highly available Kubernetes cluster manually using Google Compute Engines (GCE). Do not create a Kubernetes hosted solution using Google Kubernetes Engine (GKE). Use Kubeadm(preferred)/kubespray. **Do not use kops**.
   - ➔ Used kubeadm for installation and flannel as overlay network. Created 2 node cluster 1-master, 1-slave
   - ➔ Created 2 VM on GCP



☐ ✓ master	us-central1-a	Save $30 / mo	10.128.0.8 (nic0)	35.232.70.88	SSH	⋮
☐ ✓ worker	us-central1-a		10.128.0.9 (nic0)	34.67.135.229	SSH	⋮

Created kubernetes cluster with 2 VMs



```
architmehta06@master:~$ kubectl get nodes
NAME     STATUS   ROLES    AGE   VERSION
master   Ready    master   47h   v1.14.3
worker   Ready    <none>   47h   v1.14.3
```

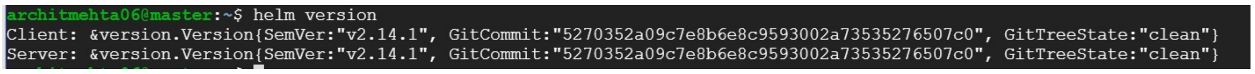
architmehta06@master:~$ kubectl cluster-info
Kubernetes master is running at https://10.128.0.8:6443
KubeDNS is running at https://10.128.0.8:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
architmehta06@master:~$ kubectl config current-context
kubernetes-admin@kubernetes

2. Create a CI/CD pipeline using Jenkins (or a CI tool of your choice) outside Kubernetes cluster (not as a pod inside Kubernetes cluster).
   → Skipped

3. Create a development namespace.
   - ➔ Skipped

4. Deploy guest-book application (or any other application which you think is more suitable to showcase your ability, kindly justify why you have chosen a different application) in the development namespace.
   - ➔ Skipped

5. Install and configure Helm in Kubernetes
   - ➔ Installed helm



```
architmehta06@master:~$ helm version
Client: &version.Version{SemVer:"v2.14.1", GitCommit:"5270352a09c7e8b6e8c9593002a73535276507c0", GitTreeState:"clean"}
Server: &version.Version{SemVer:"v2.14.1", GitCommit:"5270352a09c7e8b6e8c9593002a73535276507c0", GitTreeState:"clean"}
```

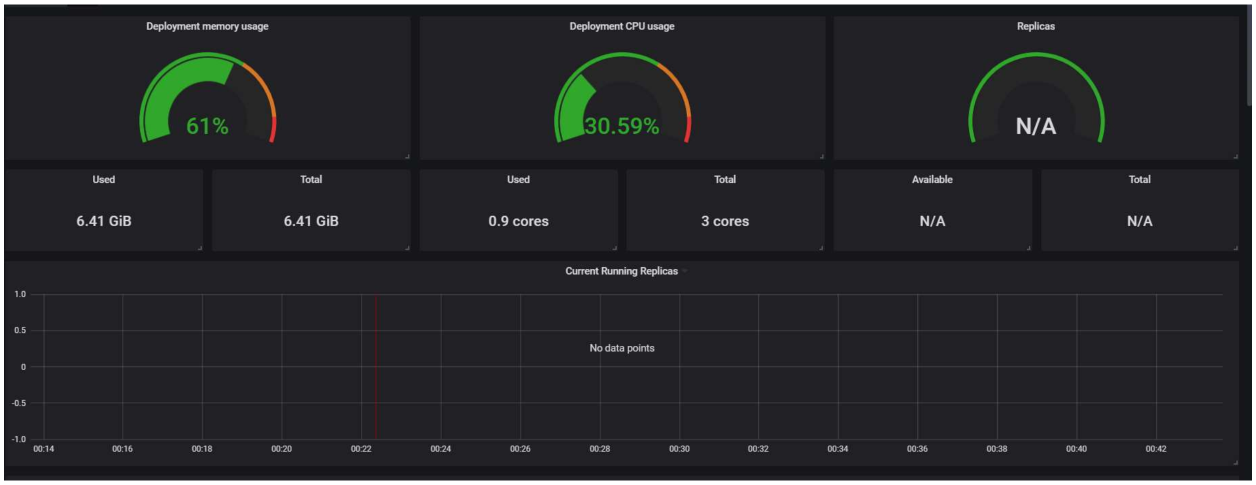
6. Use Helm to deploy the application on Kubernetes Cluster from CI server.
   ➔ Skipped

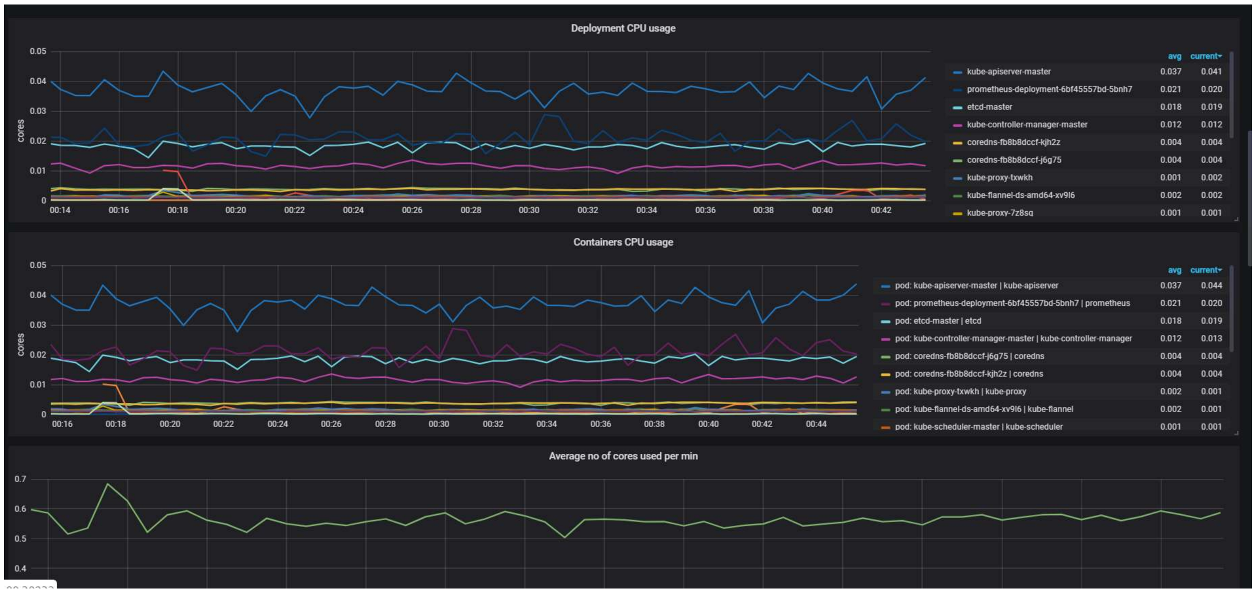
7. Create a monitoring namespace in the cluster.
   → kubectl create namespace monitoring

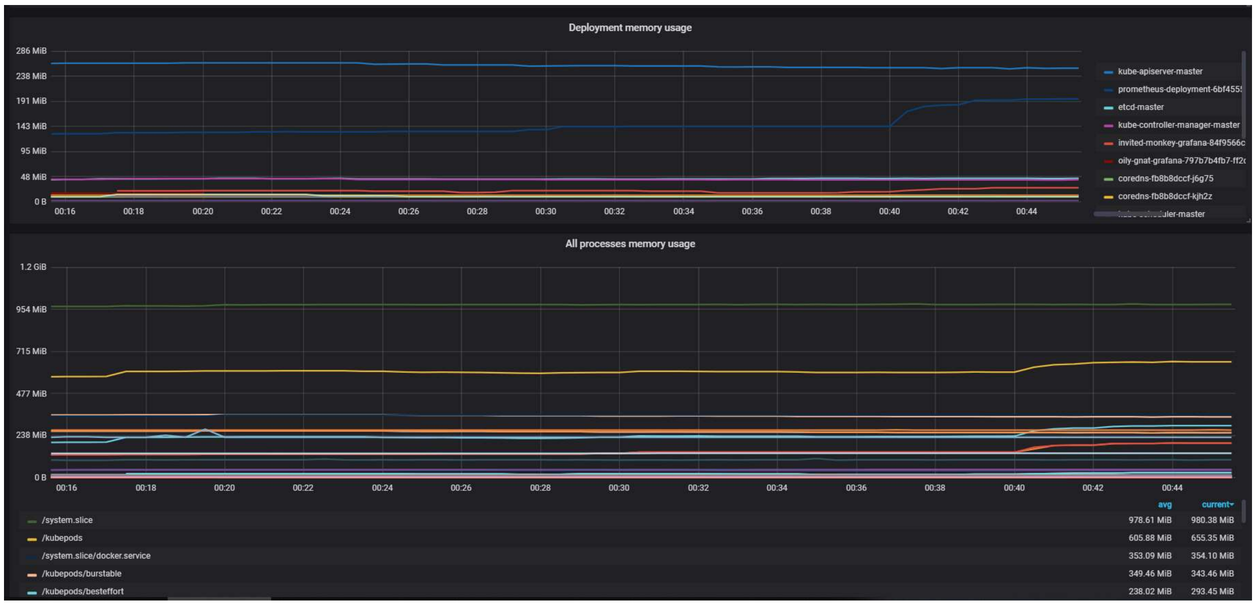
8. Setup Prometheus (in monitoring namespace) for gathering host/container metrics along with health check status of the application.
   → http://35.232.70.88:30000

9. Create a dashboard using Grafana to help visualize the Node/Container/API Server etc. metrices from Prometheus server. Optionally create a custom dashboard on Grafana
   →http://35.232.70.88:30233/

   Deployment Matrics : http://35.232.70.88:30233/d/XOE4JCfmz/kubernetes-deployment-metrics?orgId=1







10. Setup log analysis using Elasticsearch, Fluentd (or Filebeat), Kibana.
    ➔ Skipped

11. Demonstrate Blue/Green and Canary deployment for the application (For e.g. Change the background color or font in the new version etc.,)
    → Skipped

12. Write a wrapper script (or automation mechanism of your choice) which does all the steps above.
    → Skipped
13. Document the whole process in a README file at the root of your repo. Mention any pre-requisites in the README.