**NOTES:**

Task: **Level#2**

Azure: Infrastructure used

Terraform: Used to provision the infrastructure

Kubeadm : Used to create the kubernetes cluster ( Since I have limited credit limits available in Azure , I created a 3 node cluster – But HA cluster can be created by just following the link - <https://medium.com/velotio-perspectives/demystifying-high-availability-in-kubernetes-using-kubeadm-3d83ed8c458b> )

NodePort : Used to expose the services.

OS : Ubuntu 18.04

**Applications are accessible on the below URL’s.**

* http://157.55.86.132:32319 --> grafana
* http://157.55.86.132:30170 --> prometheus
* http://157.55.86.132:31416 --> guestbook
* http://157.55.86.132:30194/app/kibana --> kibana
* http://157.55.86.132:32721 --> Elasticsearch
* http://23.102.128.125:30194/job/helm\_install\_tomcat/12/console --> application installation from Jenkins though helm charts.
* http://157.55.86.132:32555/ --> blue-green deployments
* while sleep 0.1; do curl 157.55.86.132:32097 -H "Host: my-app.com"; done --> run this command on the server to see the canary deployment results

1. **Provision the infrastructure.**
2. Requirements: Azure-cli, terraform
3. Go to terraform folder
4. Az login and get authenticated with azure
5. Run the commands – terraform init;terraform plan; terraform apply
6. Infrastructure is ready now – ( This will create four servers including master ,node1,node2 and Jenkins )
7. **Install kubernetes cluster with kubeadm**
8. Run the script **install\_master.sh** on the master node , this will install all the required software’s and make master node ready
9. Run the script **install\_nodes.sh** on the slave nodes , this install required software’s and join the node to the cluster

root@master:~# kubectl get nodes

NAME STATUS ROLES AGE VERSION

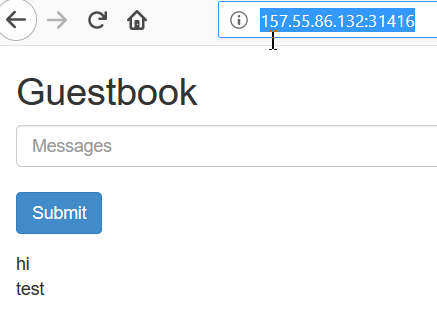
master Ready master 2d19h v1.14.3

node1 Ready <none> 2d19h v1.14.3

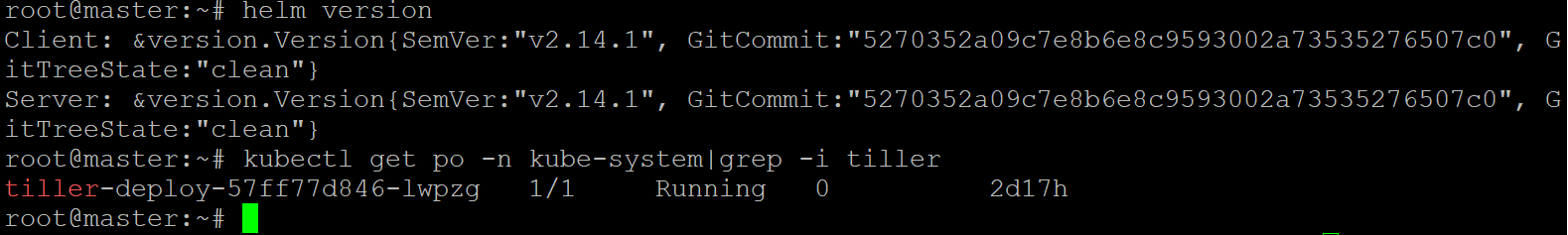
node2 Ready <none> 2d18h v1.14.3

root@master:~#

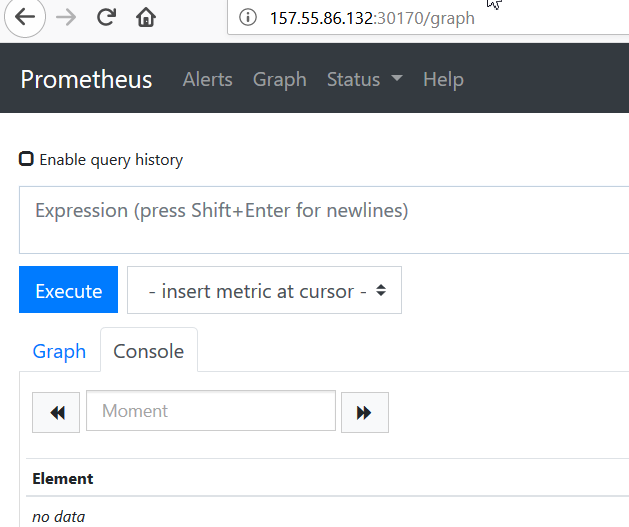
1. **Guest-book installation**
2. Run the script **install\_gues-book.sh** – This will create the development name space and install guest-book application on the development namespace.
3. Application will be accessible through the URL <http://157.55.86.132:31416/>

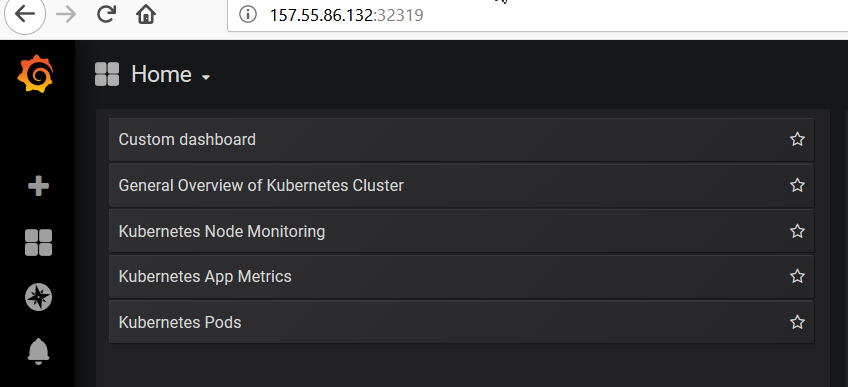


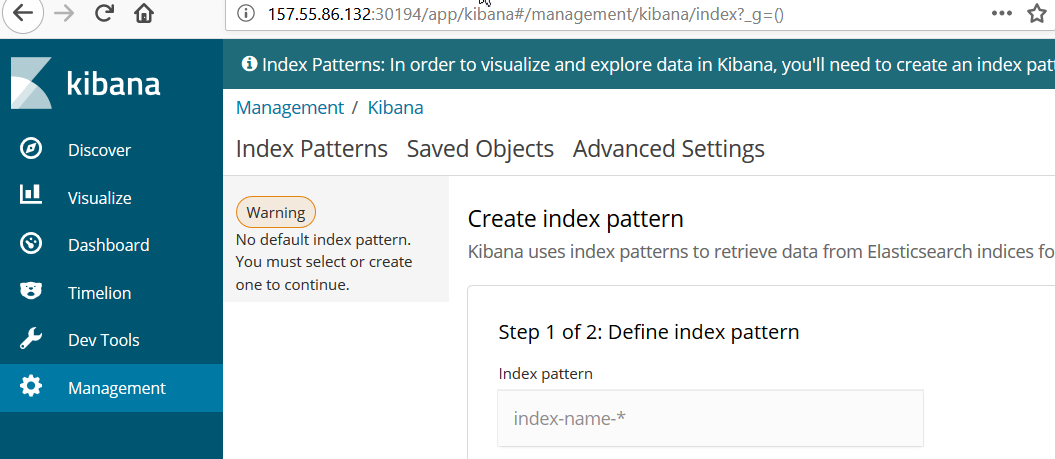
1. **Install and configure Helm in Kubernetes**
2. Run the script **install\_helm.sh -**  This will install helm and tiller on the kube-system namespce



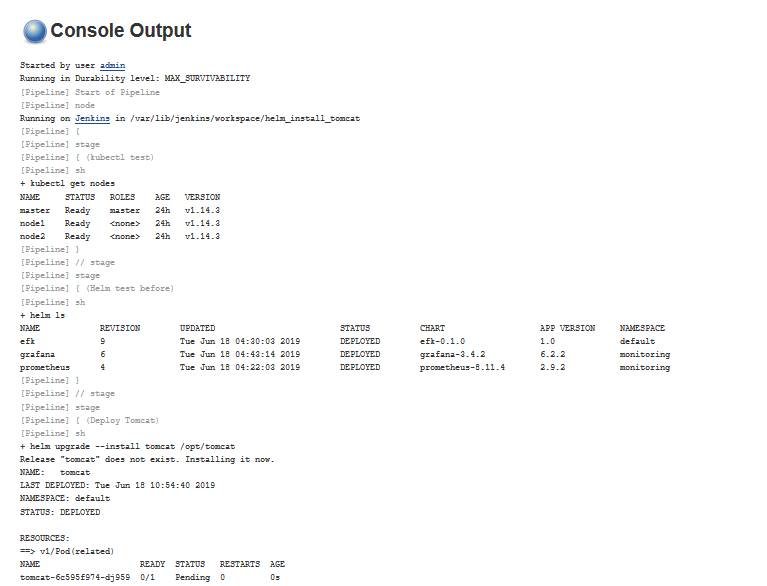
1. **Installation of Prometheus , grafana and efk using helm charts .**
2. Run the script **install\_grafana\_prometheus\_efk.sh -** This will install Prometheus and Grafana on the monitoring namespace ( grafana datasource is also updated through the helm charts) and efk on the efk namespace .
3. Required dashboards created on Grafana

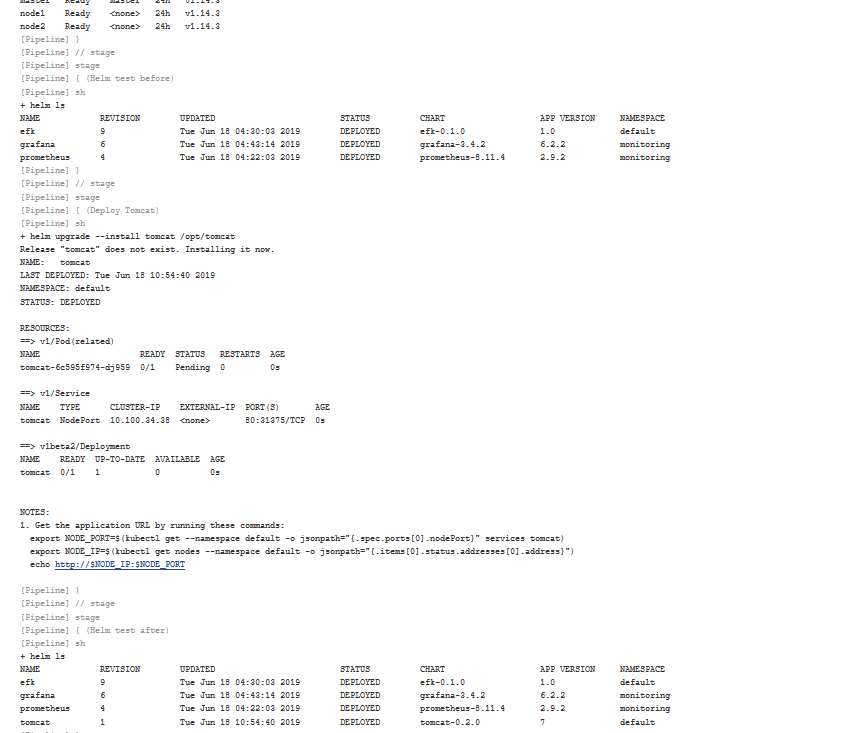






1. **Create a CI/CD pipeline using Jenkins**
2. Run the script **install\_jenkins.sh** on the Jenkins server **-** This will install required software’s on the server and setup the Jenkins server.
3. Install required plugins
4. Create a pipeline job to install application on the kubernetes cluster using helm
5. Tomcat application used for the installation





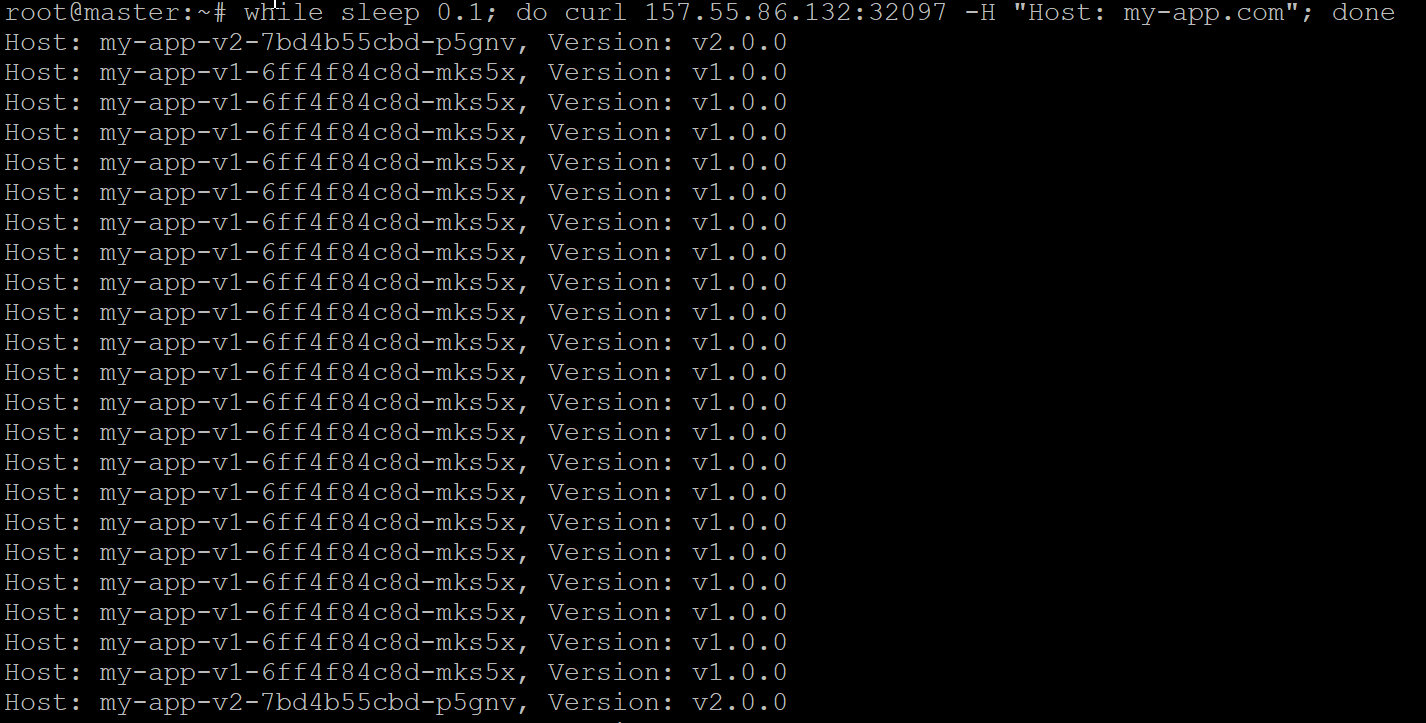
1. **Demonstrate Blue/Green deployment**
2. Run the script **install\_blue\_green.sh** - This will install the deployments.
3. Used custom image and put this image manually on the nodes.





1. **Canary deployment**
2. Run the script **install\_canary.deployment.sh -** script will do the below steps.
   1. version 1 is serving traffic
   2. deploy version 2
   3. create a new "canary" ingress with traffic splitting enabled
   4. wait enought time to confirm that version 2 is stable and not throwing unexpected errors
   5. delete the canary ingress
   6. point the main application ingress to send traffic to version 2
   7. shutdown version 1

split traffic: 90% to v1, 10% to v2



1. **wrapper scripts -**  All scripts placed in the wrapper\_scripts directory.