# ELK SETUP ON UBUNTU

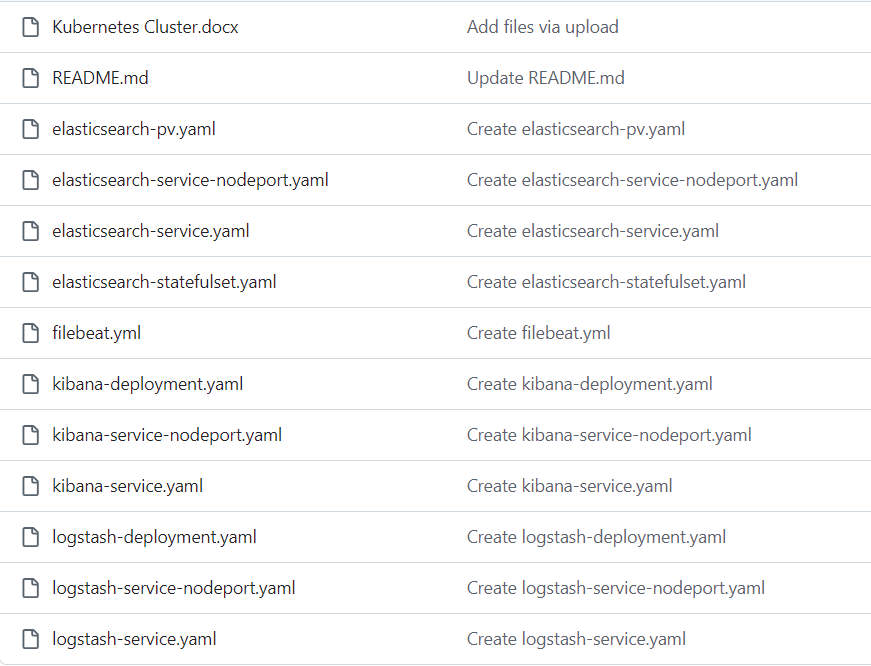
Pre-Requisites

* Ubuntu 22.04.4
* Kubernetes Version 1.30.4

# STEP 1

We have pushed YAML file to set-up ELK stack, First of all just clone the repo with the help of following command

git clone <https://github.com/ahmedbinmasud/elk-k8s-cluster>



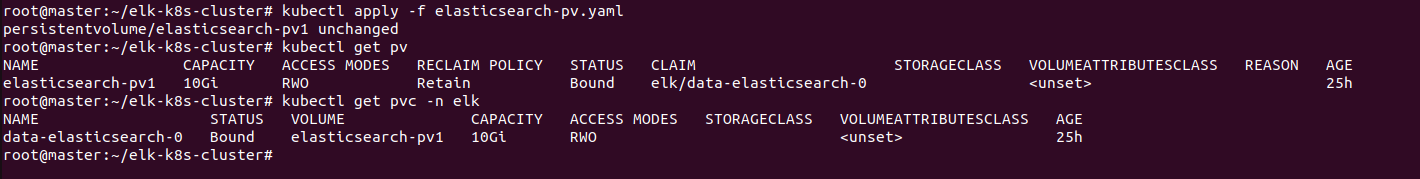
# STEP 2

After cloning the repo we have to jump into directory elk-k8s-cluster where we have all the above files.

## **Creating Persistent Volume**

**Kubectl apply –f elasticsearch-pv.yaml**

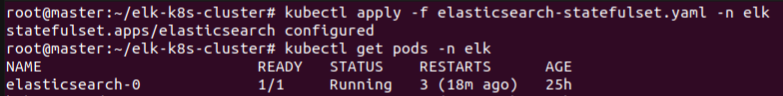
First of all we need to make Persistent Volume by applying the elasticsearch-pv.yaml file this will create a PersistentVolume named elasticsearch-pv1 with a storage capacity of 10Gi, allowing read-write access by a single node at a time (ReadWriteOnce). The volume uses the hostPath type, which directly mounts the node's filesystem directory /mnt/data/elasticsearch1 into the pod, ensuring data persistence on that specific node.



## **Creating Elasticsearch Pod**

**Kubectl apply –f elasticserch-statefulset.yaml**

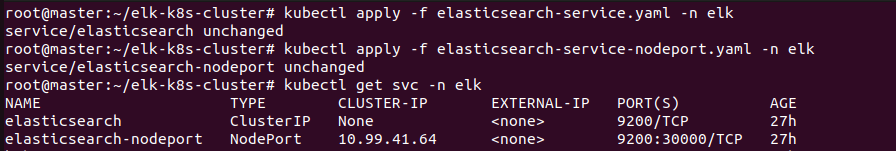
Applying this manifest will create a StatefulSet named elasticsearch with a single replica, deploying an Elasticsearch pod using the image docker.elastic.co/elasticsearch/elasticsearch:7.9.3. The pod will expose port 9200 for HTTP traffic and operate in single-node discovery mode, suitable for non-clustered setups. The pod's data will be stored in a persistent volume, mounted at /usr/share/elasticsearch/data, with a storage capacity of 10Gi requested through a PersistentVolumeClaim. The StatefulSet ensures stable network identities and persistent storage, ideal for stateful applications like Elasticsearch.

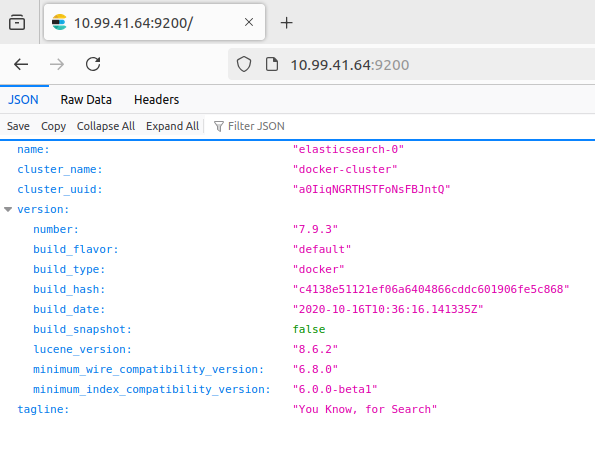


To access Elasticsearch we need create a service for that run the below services

**Kubectl apply –f elasticsearch-service.yaml**

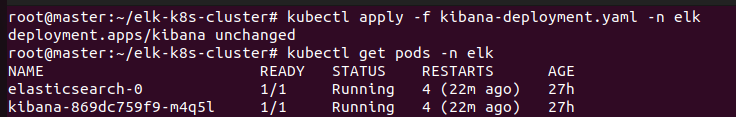
**Kubectl apply –f elasticsearch-service-nodeport.yaml**





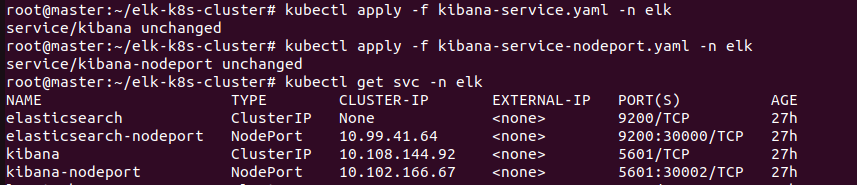
## **Deploying Kibana**

**Kubectl apply –f kibana-deployment.yaml**

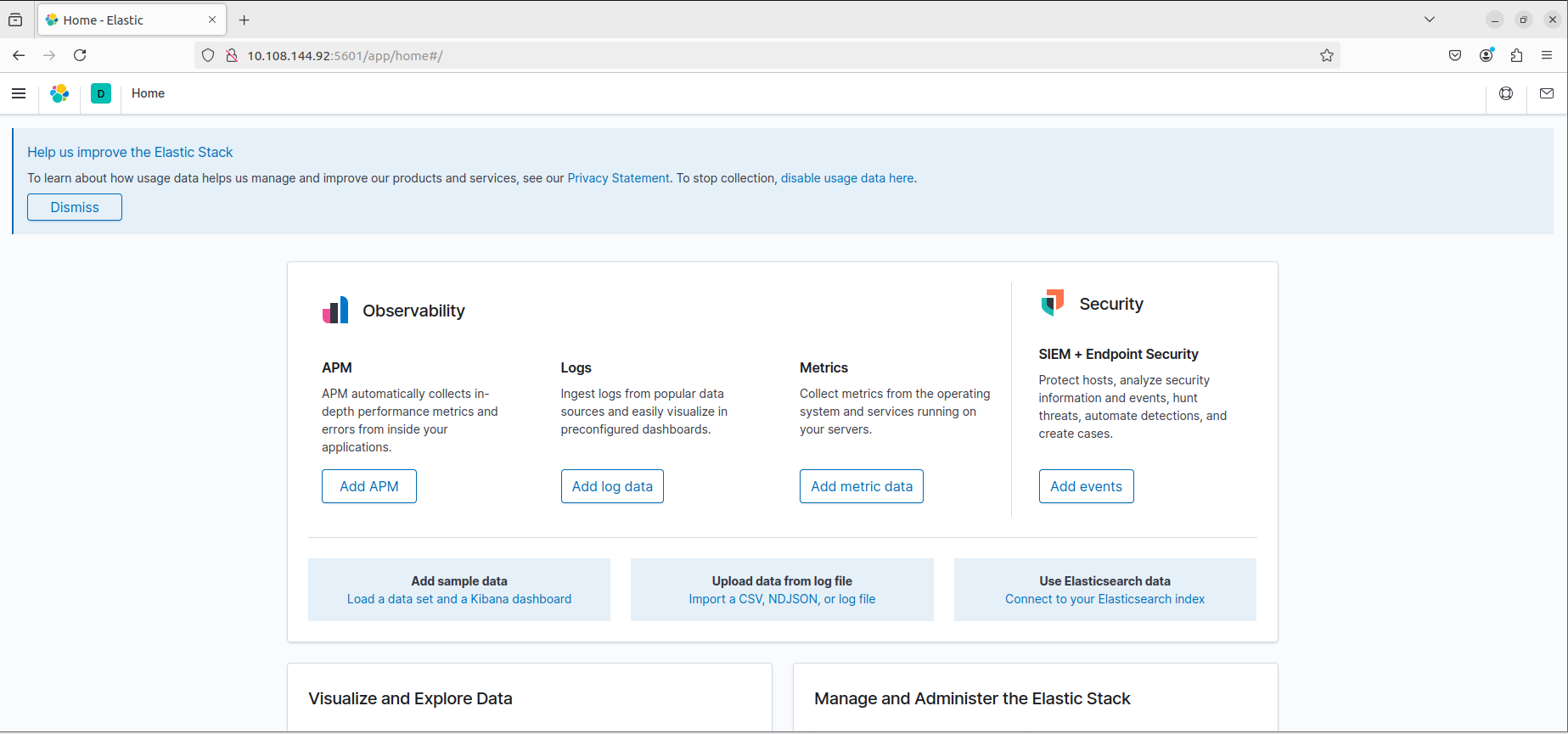


**Kubetcl apply –f kibana-service.yaml**

**Kubetcl apply –f kibana-service-nodeport.yaml**



This configuration deploys a Kibana instance in Kubernetes with a single replica, exposing it on port 5601. It includes a ClusterIP service for internal cluster access and a NodePort service, which makes Kibana accessible externally on port 30002, allowing both internal and external users to interact with the Kibana interface for visualizing Elasticsearch data.



## **Deploying Logstash Pod**

**Kubectl apply –f logstash-deployment.yaml**

**Kubetcl apply –f logstash-service.yaml**

**Kubectl apply –f logstash-service-nodeport.yaml**

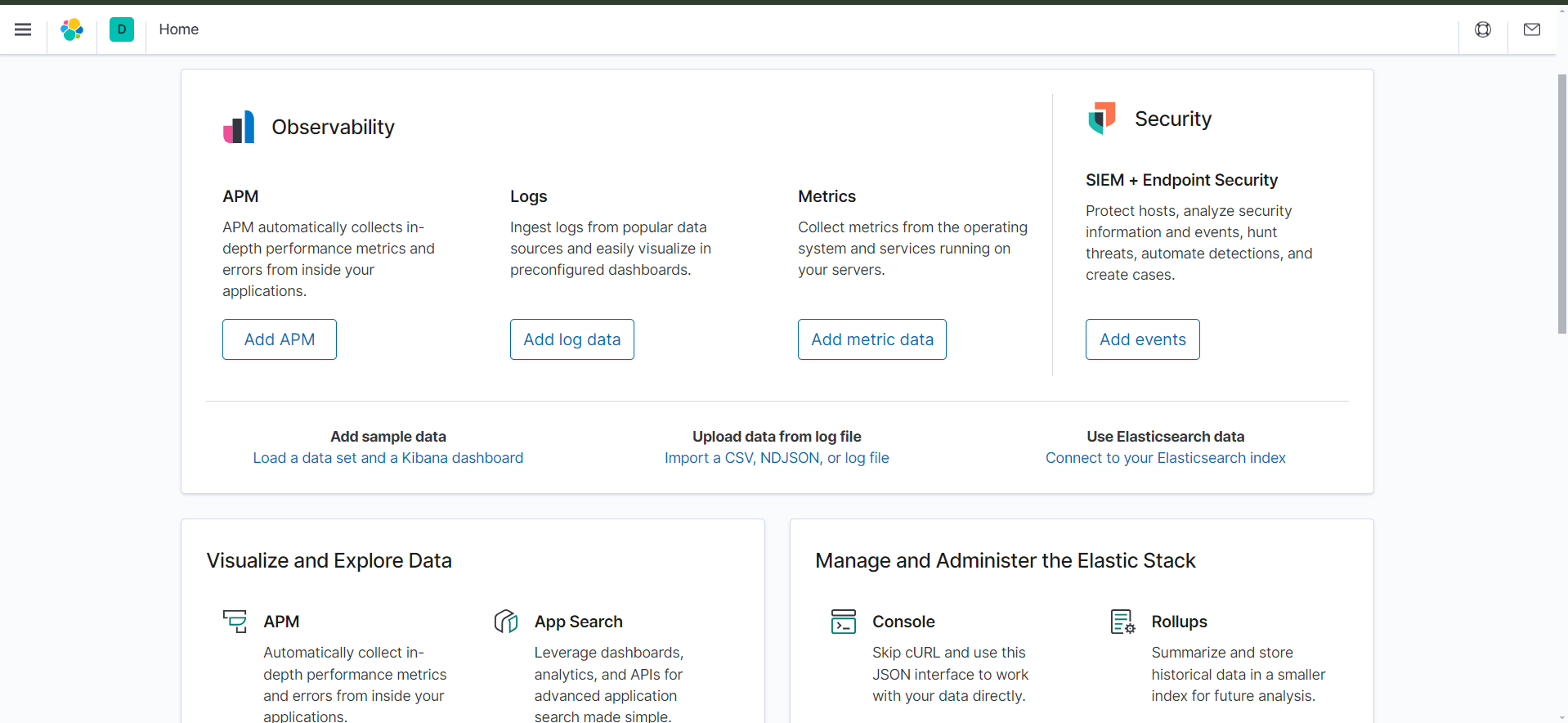
This configuration deploys a Logstash instance in Kubernetes using a Deployment with one replica, exposing the container on port 5044. Two services are created: a ClusterIP service named logstash for internal cluster access, and a NodePort service named logstash-nodeport, which exposes Logstash externally on port 30001. This setup allows Logstash to receive logs from both within the Kubernetes cluster and from external sources.

## **Installing Filebeat**

Pre-Requisites

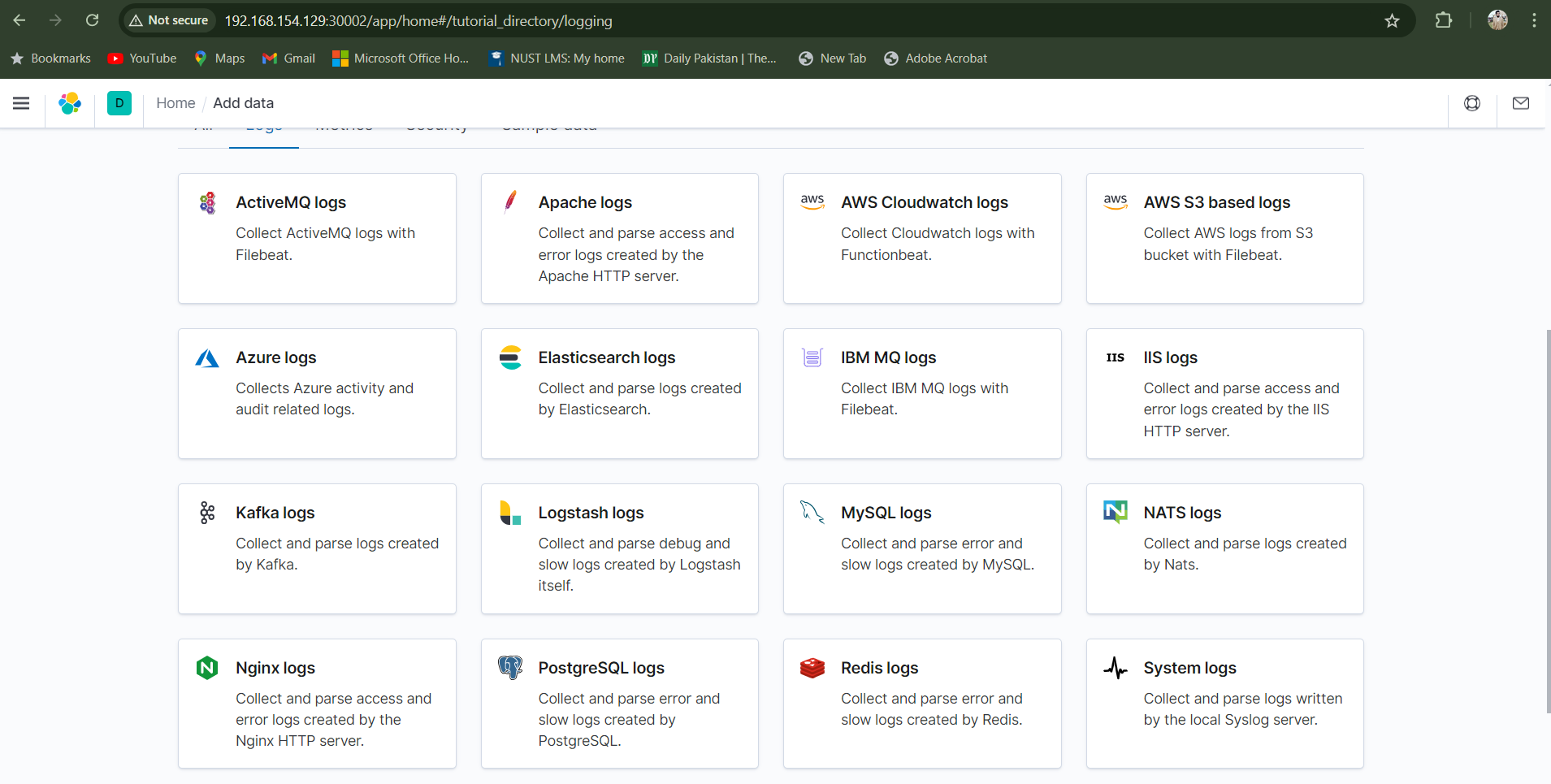
* ELK and Filebeat verison 7.9.3
* Kubernetes Version 1.30.4

# Step 1



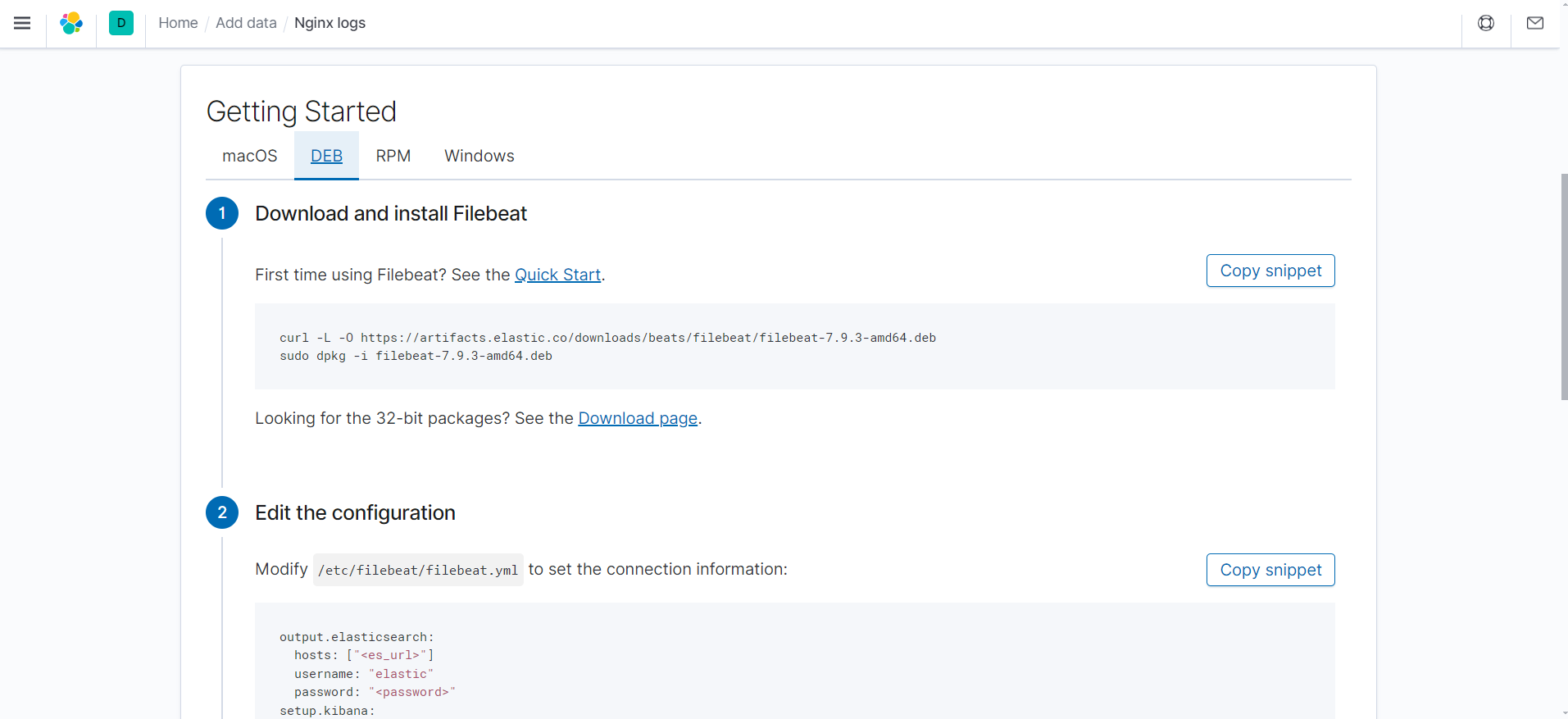
After successfully loading the kibana dashboard using the nodeport ip and port you need to visit the Logs and click on Add log data.  
  
 Step 2

You will get some options you can select any depending on the which service logs do you want to see in kibana . I my case I am going with Nginx Logs

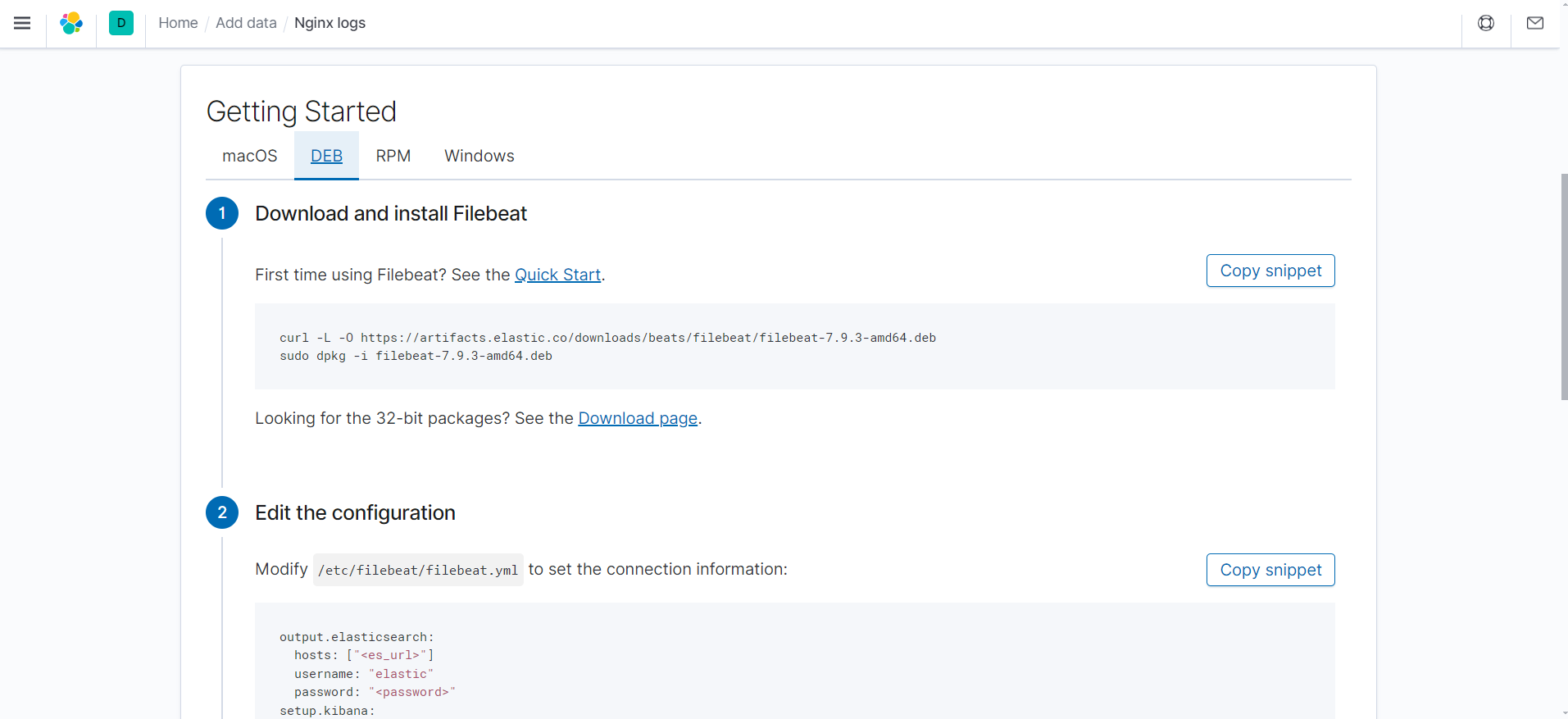


# Step 3:

# Choose Your Operating System

  
  
Then after selecting DEB for Ubuntu or RPM for Redhat kibana will start showing you steps to Install and setup filebeat to start fetching logs from nginx and delivering them to elk stack .

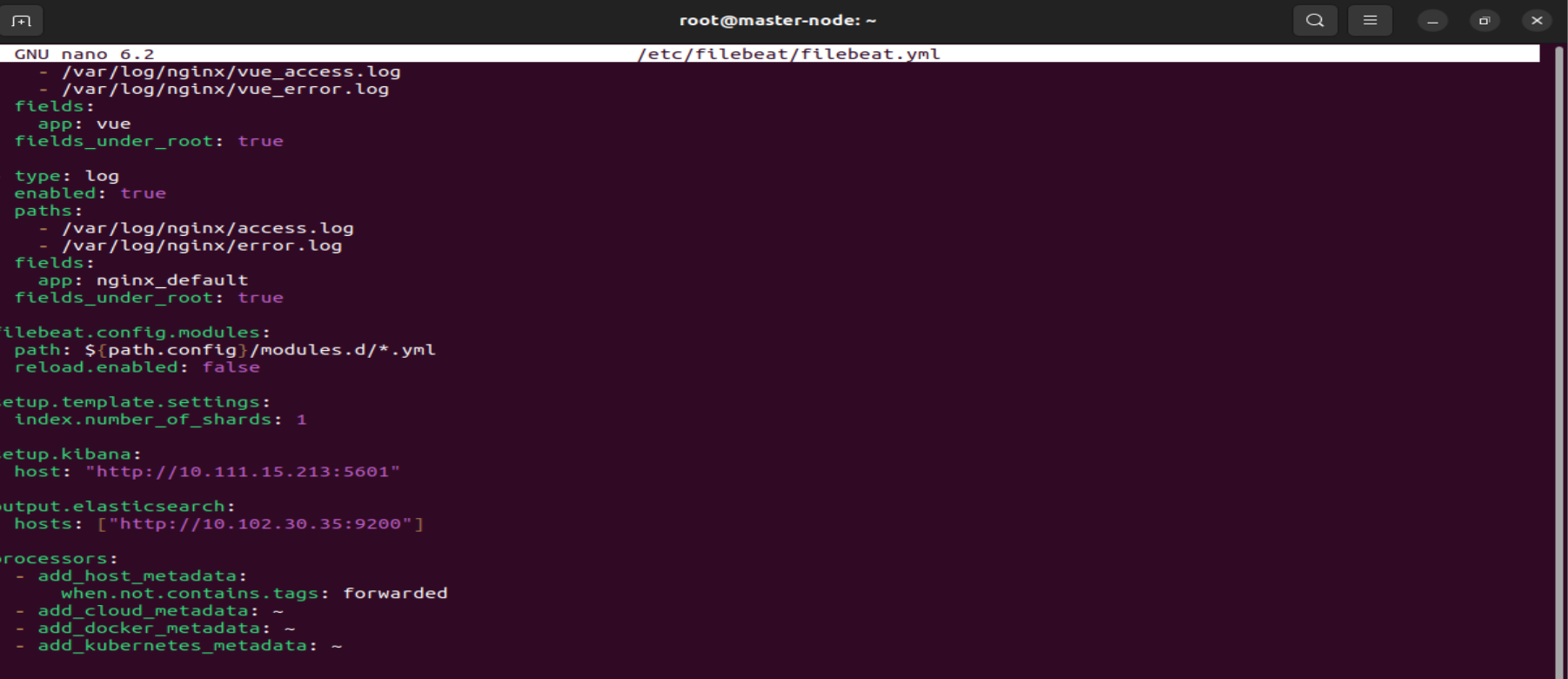
# Step 4 Download and Install filebeat



After running the first step of Download and Install Filebeat and after depackaging it all the filebeat directories will be set up in your local machine/server then go on to the next step and edit the configuration in /etc/filebeat/filebeat.yml

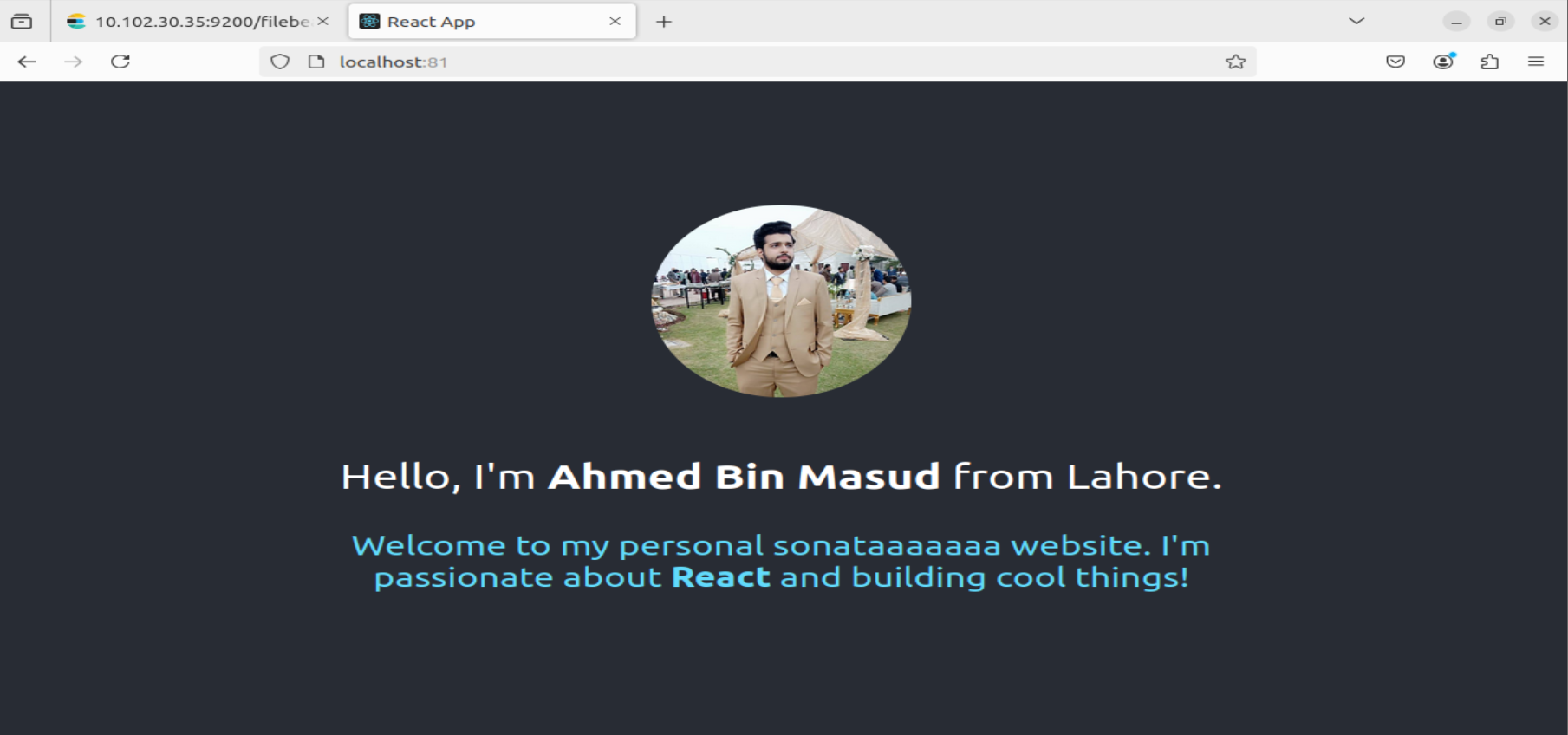
# Step 5 Make Changes in etc filebeat.yml file

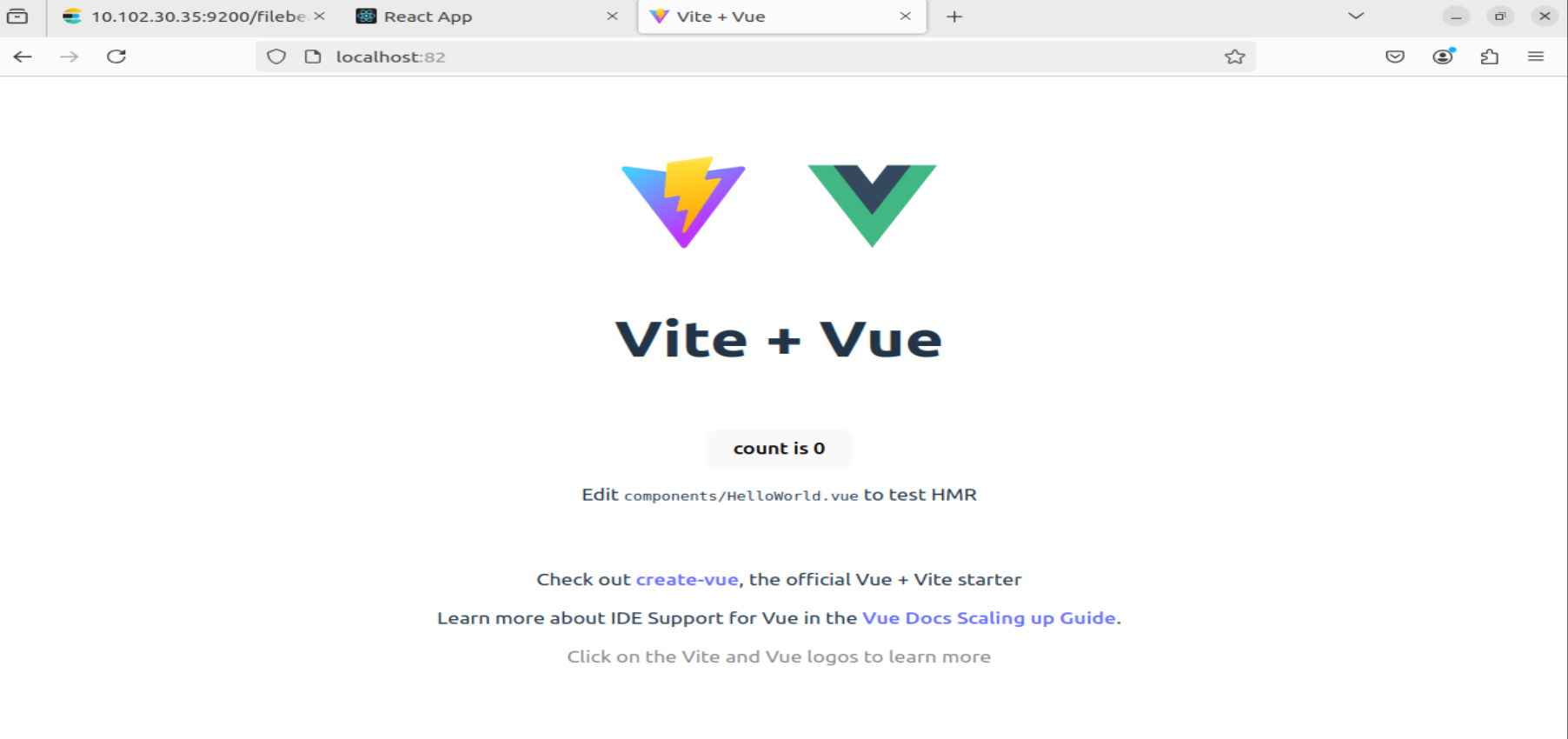
In /etc/filebeat/filebeat.yml you need to provide the nodeport ip and port of you elastic search and kibana pods just uncomment and change the dummy ip and port already provided in the .yml file

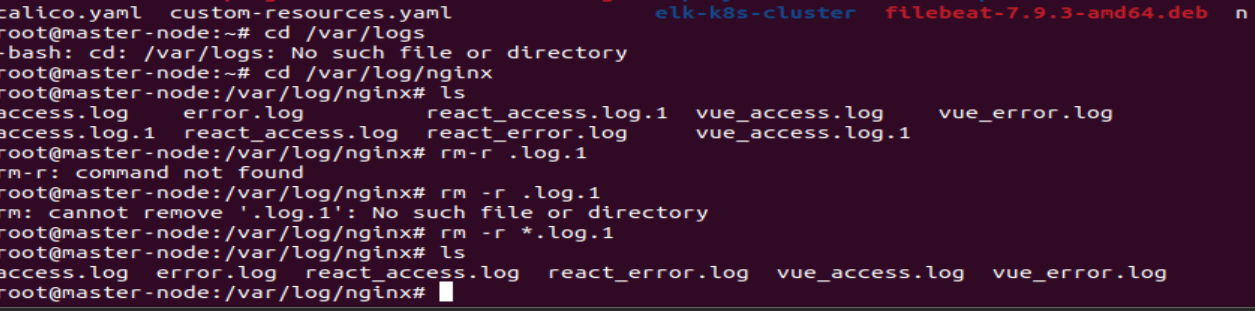


# Step 5 NGINX Deployments

These are the two apps I deployed on nginx one is react app and other one is vue app . React app in running on port 81 and vue app is running on port 82

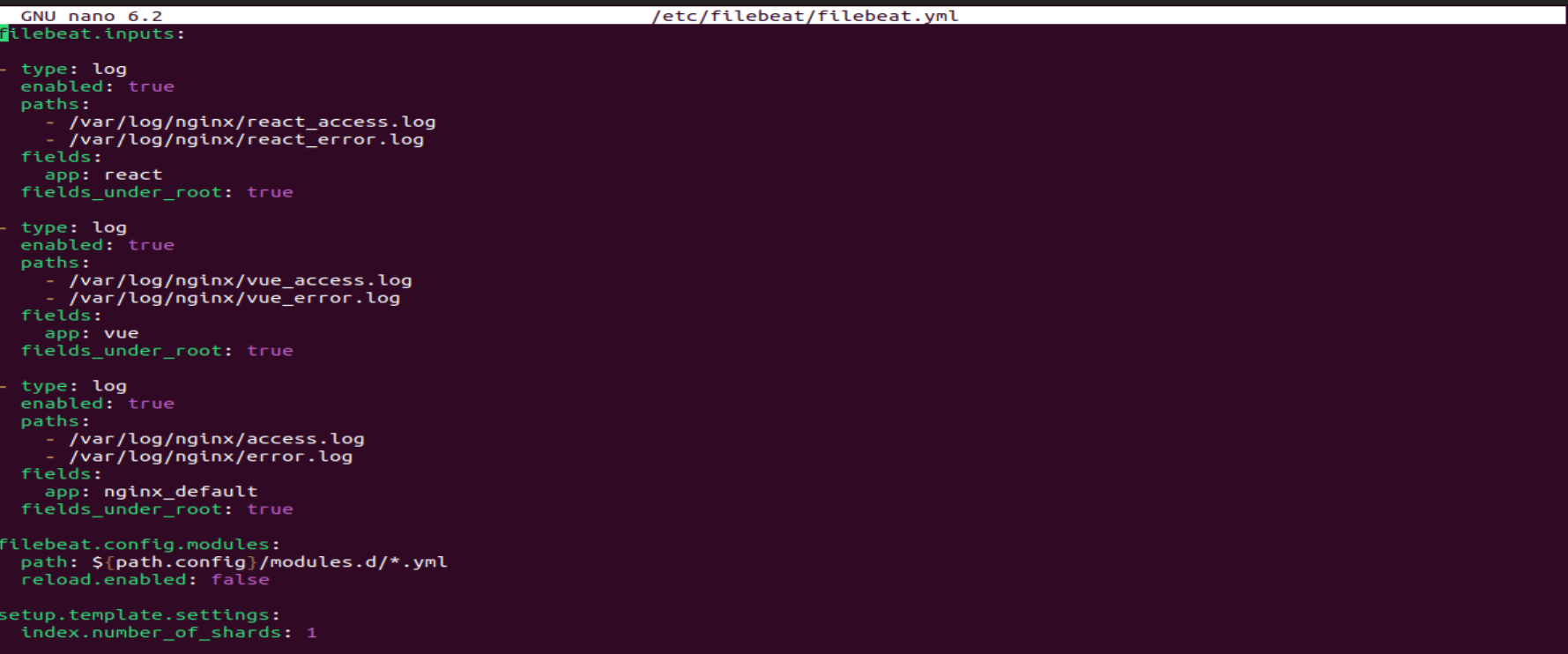


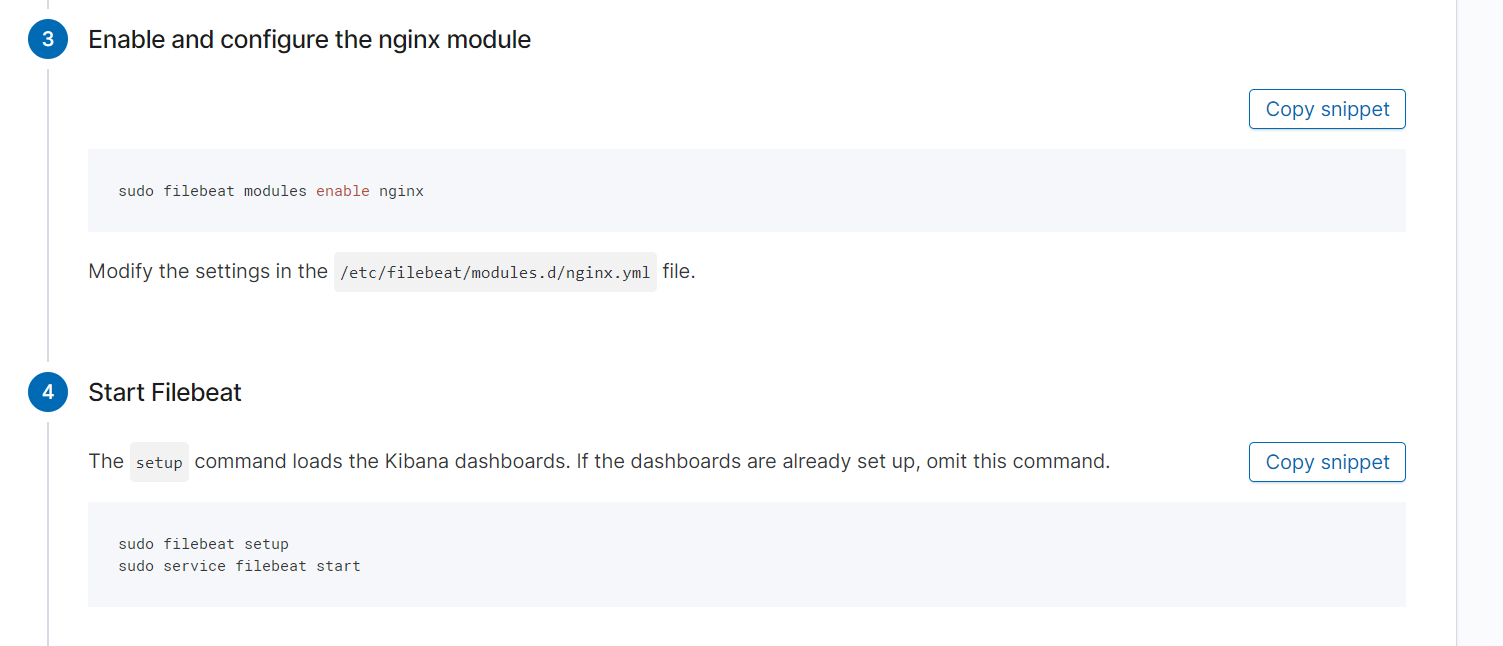


There is where I am receiving the logs of each app separately   


# Step 6 Specify the log path in etc filebeat.yml file

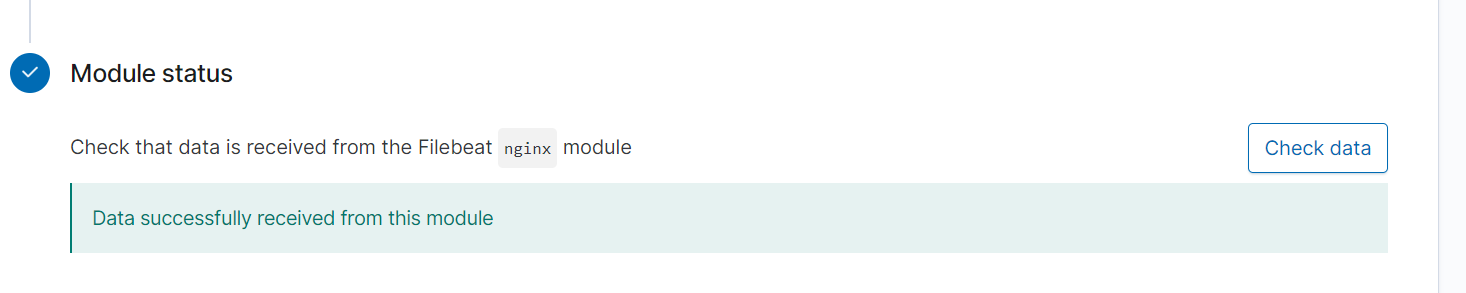
Also specify the full log file path of the website you want to see logs in kibana dashboard   
this is the same filebeat.yml file in which we added nodeport ip and ports



Step 7 Enable and start filebeat   
 After that run the commads mentioned in step 3 and step 4 which will enable the nginx module of filebeat and then start filebeat service

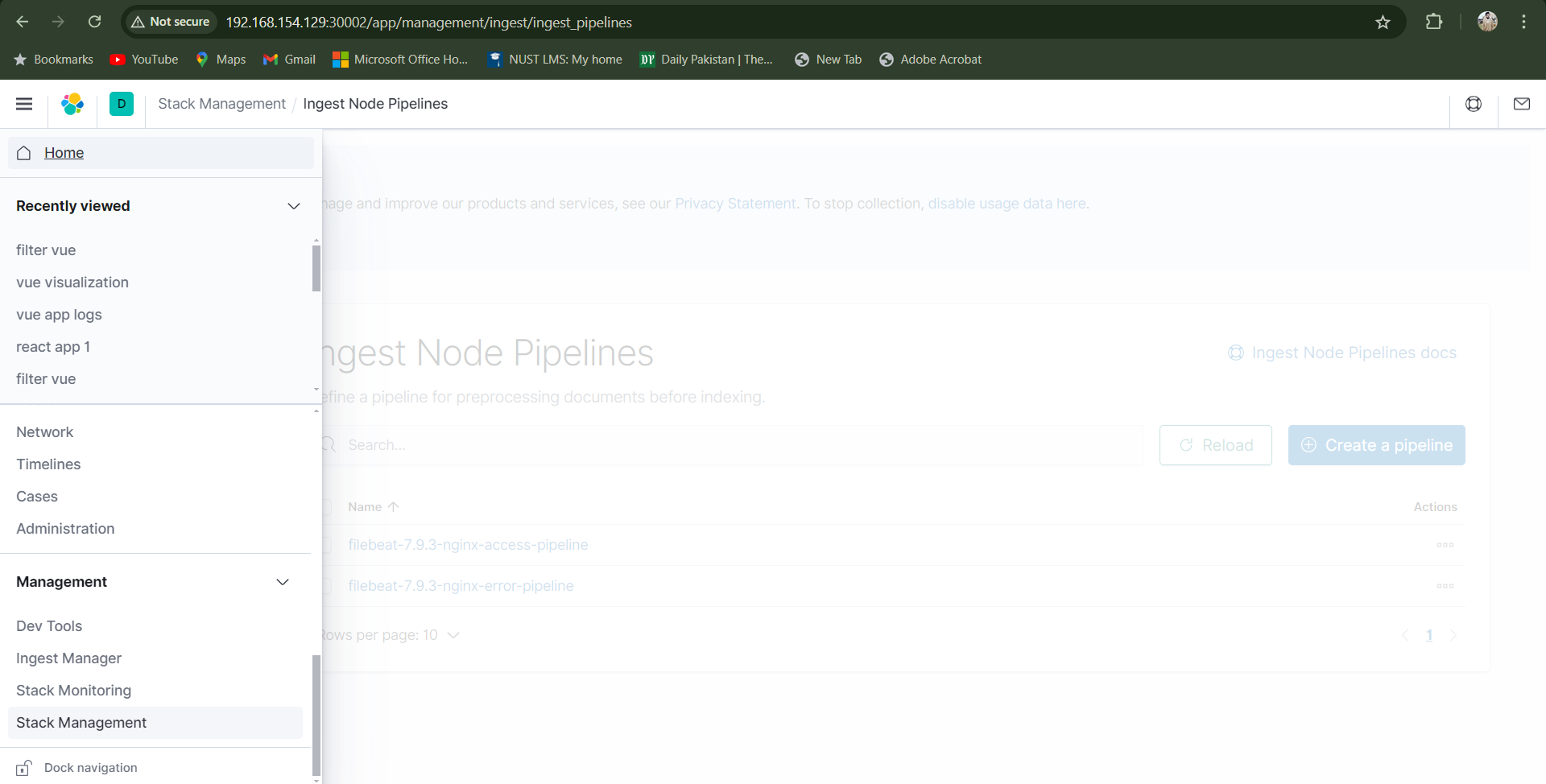
Step 8 Check if data is being recieved

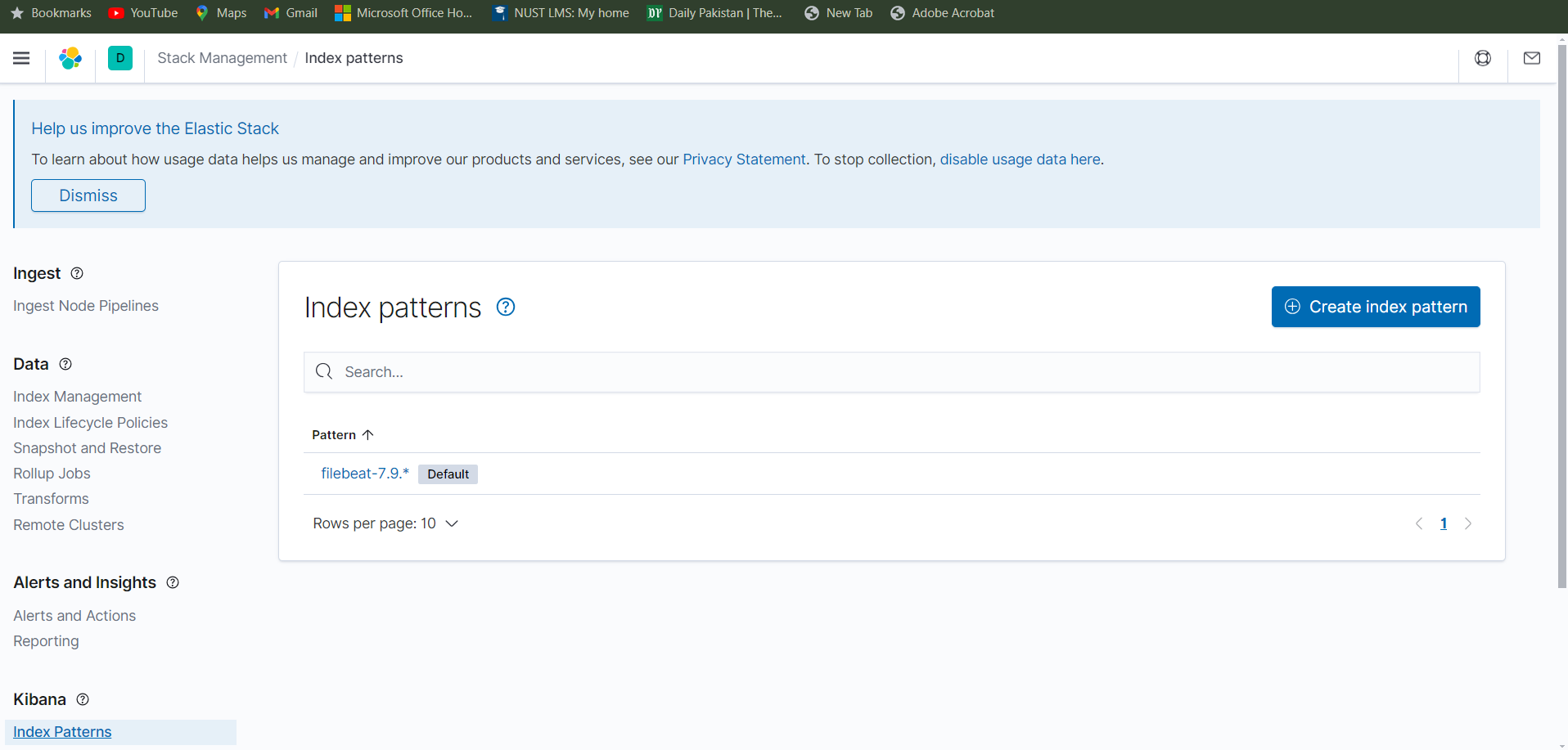
You can check if data is being received in elk by filebeat or not by clicking check data in my case I am getting Data successfully received from this module means all the things are setup from my side .If you get any error after clicking on check data reload this page and check again .

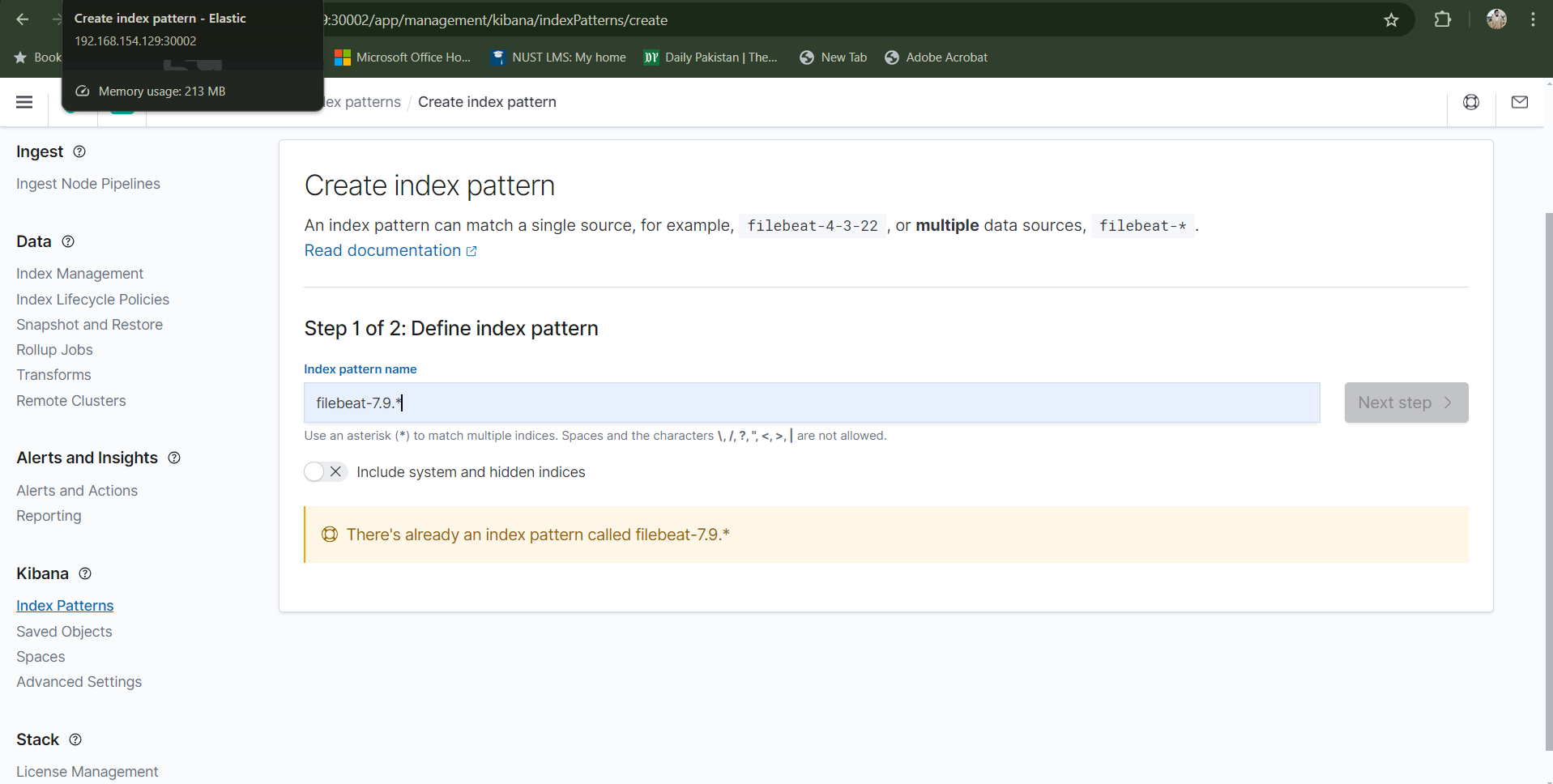


# Step 9

Then go to the Stack Management and create new filebeat index.





In the create Index pattern write your specific version of filebeat like I have written filebeat-7.9.\* adding streic will add all the entreies with this name and after that select timestamp  
  


# Step 10 Start getting logs

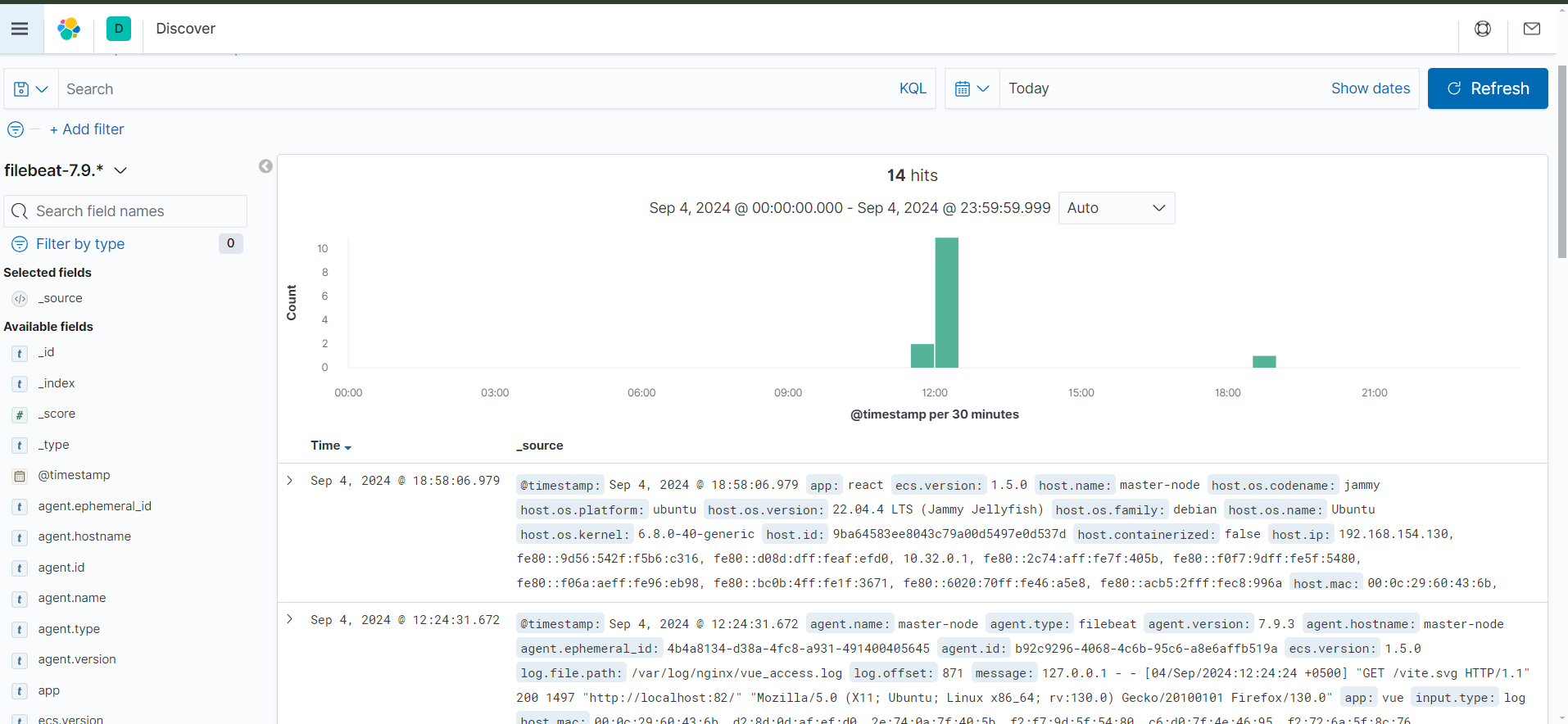
Then go to discover portion of kibana where you will be able to see the logs coming directly from nginx /var/logs/nginx

# 

# Step 11:

In this specific portion select filebeat 7.9.\* the same you just added in index patterns

You will now be able to see the logs



You can separate the logs based on the Queries which you will require to put only once

