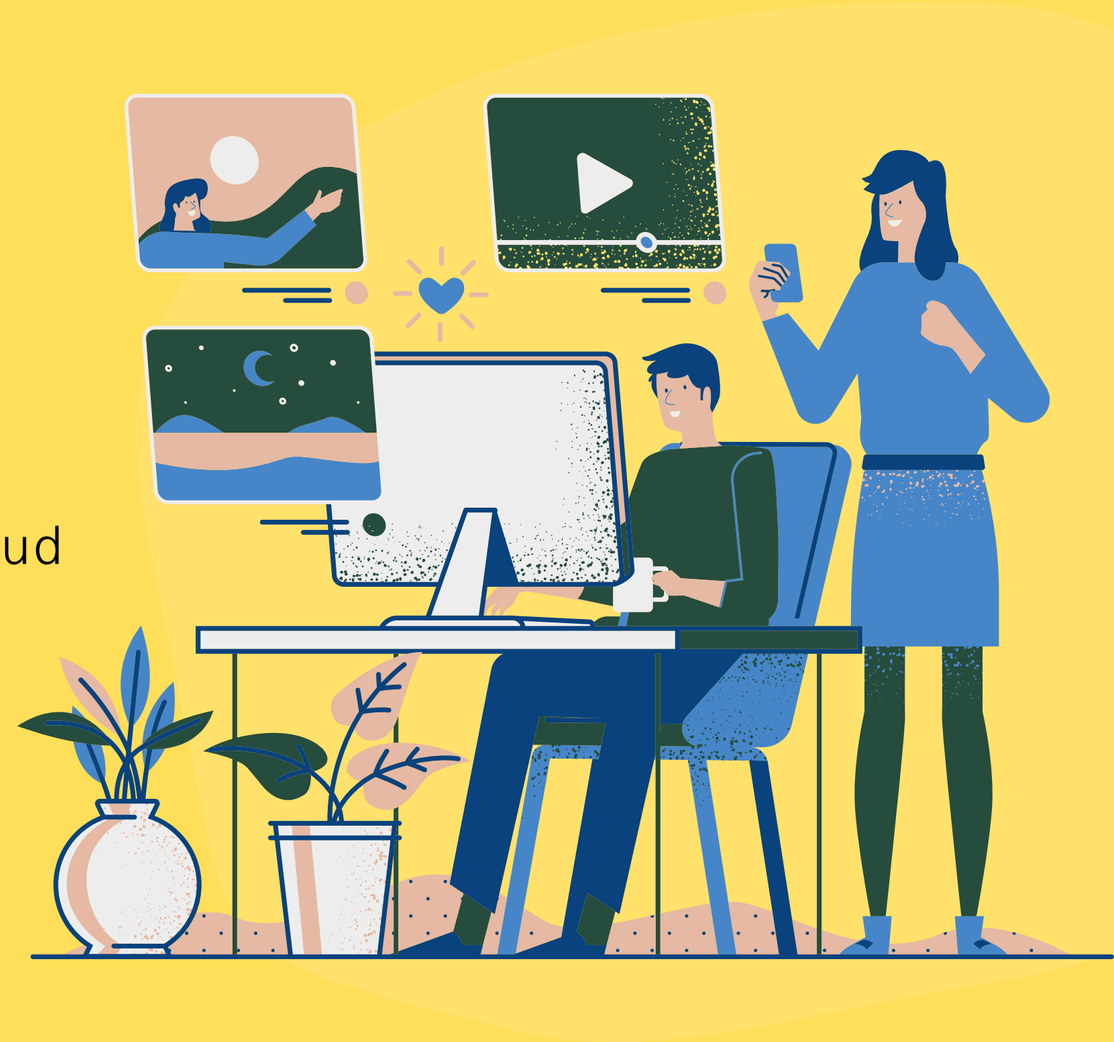
Big Project

Infrastructure Migration From On Premise to AWS Cloud



Case Perusahaan Cilsy

Saat ini, Cilsy sedang melakukan development produk baru yang bernama Cilist (CilsyList) Cilist ini merupakan aplikasi to-do list yang berbasis web. Aplikasi ini dibangun menggunakan MERN Stack (MongoDB, ExpressJS, ReactJS, dan NodeJS)

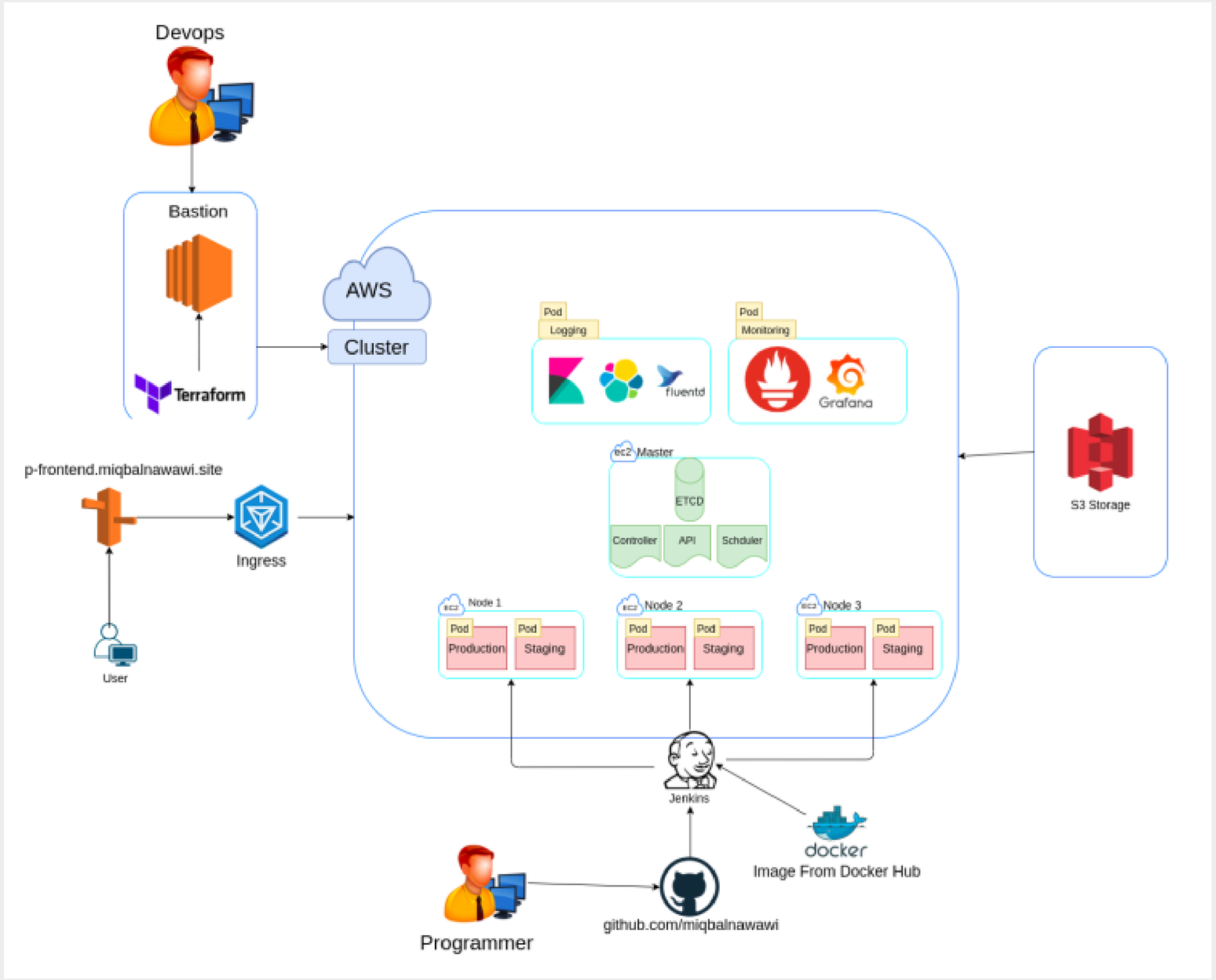
Problem

Saat ini frontend, backend, dan database baru disimpan di 1 vm yang sama sehingga apabila terjadi error di vm tersebut, maka aplikasi tidak bisa digunakan sama sekali.

Solution

- Membuat topologi yang cocok sesuai kebutuhan
- Membuat budgeting selama 6 bulan pemakaian
- Membuat Environtment Staging dan Production
- Mendeploy Image tersebut ke kubernetes cluster
- Mengunakan fitur CICD dengan Jenkins
- Membuat tools monitoring dan logging untuk memantau kinerja Apps
- Memisahkan service antara backend , frontend dan database

TOPOLOGI


Devops
Bastion
Terraform
AWS
Cluster
Pod
Logging
fluentd
Pod
Monitoring
Grafana
S3 Storage
p-frontend.miqbalnawawi.site
Ingress
User
Master
ETCD
Controller
API
Scheduler
Node 1
Node 2
Node 3
Pod
Production
Staging
Jenkins
docker
Image From Docker Hub
Programmer
github.com/miqbalnawawi

Estimate Budget

Service	Type	Count	Price
EC2	T2 Large	1	$40.25
EC2	T2 Medium	3	$45.87
EC1	T2 Micro	1	$4.60
S3	-	1	$0.03
Route 53	-	1	$0.50
LoadBalancer	Classic	4	$55.20
Total Monthly Payment			$146.45
6 Months Total Payment			$878.70

CREATE BASTION WITH TERRAFORM

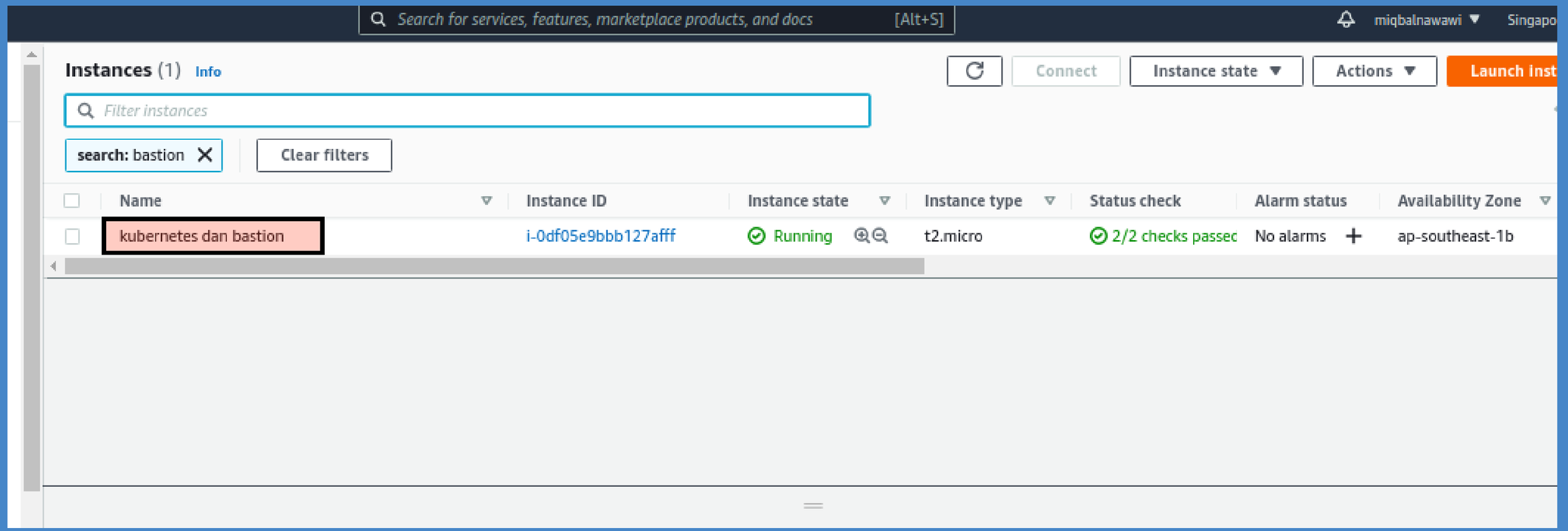
Create Env :

```
variable "aws_access_key" {
   default = "your_aws_access_key"
}

variable "aws_secret_key" {
   default = "your_aws_secret_key"
}

variable "region" {
  default = "ap-southeast-1"
}

variable "availability_zone" {
   default = "ap-southeast-1a"
}

variable "ami" {
   default = "ami-05b891753d41ff88f"
}

variable "instance_type" {
   default = "t2.micro"
}

variable "root_volume_size" {
   default = 22
```

Main.tf :

```
provider "aws" {
  access_key = "${var.aws_access_key}"
  secret_key = "${var.aws_secret_key}"
  region     = "${var.region}"
}

resource "aws_instance" "instance" {
  ami = "${var.ami}"
  instance_type = "${var.instance_type}"
  key_name = "${var.key_name}"
  vpc_security_group_ids = ["${var.vpc_security_group_ids}"]
  subnet_id = "${var.subnet_id}"
  associate_public_ip_address = "${var.associate_public_ip_address}"
  count = "${var.instance_count}"
  root_block_device {
    volume_type           = "${var.volume_type}"
    volume_size           = "${var.root_volume_size}"
    delete_on_termination = "${var.delete_on_termination}"
```

BASTION CREATED



CREATE JENKINS WITH TERRAFORM

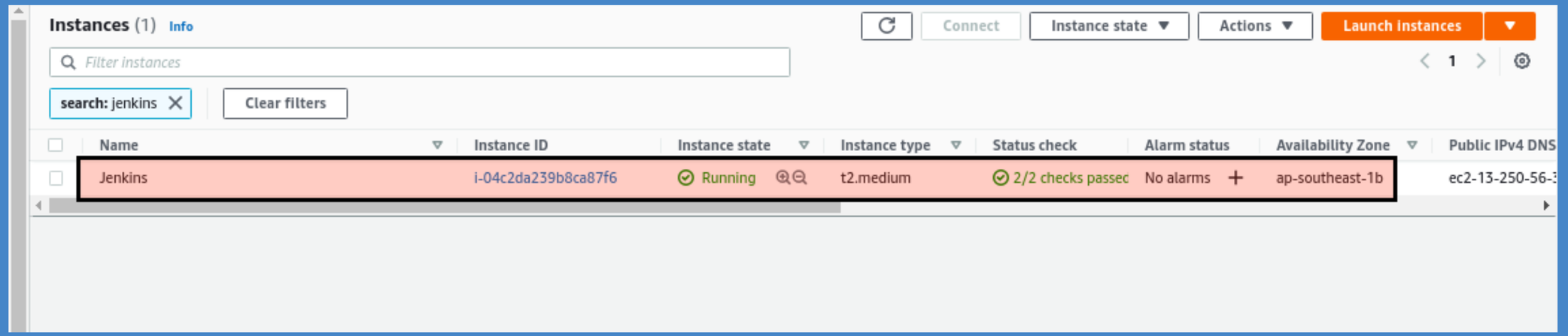
Create Env :

```
variable "aws_access_key" {
   default = "your_aws_access_key"
}

variable "aws_secret_key" {
   default = "your_aws_secret_key"
}

variable "region" {
  default = "ap-southeast-1"
}

variable "availability_zone" {
   default = "ap-southeast-1a"
}

variable "ami" {
   default = "ami-05b891753d41ff88f"
}

variable "instance_type" {
   default = "t2.micro"
}

variable "root_volume_size" {
   default = 22
```

Main.tf :

```
provider "aws" {
  access_key = "${var.aws_access_key}"
  secret_key = "${var.aws_secret_key}"
  region     = "${var.region}"
}

resource "aws_instance" "instance" {
  ami = "${var.ami}"
  instance_type = "${var.instance_type}"
  key_name = "${var.key_name}"
  vpc_security_group_ids = ["${var.vpc_security_group_ids}"]
  subnet_id = "${var.subnet_id}"
  associate_public_ip_address = "${var.associate_public_ip_address}"
  count = "${var.instance_count}"
  root_block_device {
    volume_type           = "${var.volume_type}"
    volume_size           = "${var.root_volume_size}"
    delete_on_termination = "${var.delete_on_termination}"
```

JENKINS CREATED



CREATE CLUSTER WITH KOPS

Create Env :

```
export bucket_name=belajar-kubernetsexport
KOPS_CLUSTER_NAME=belajarkubernetes.comexport
KOPS_STATE_STORE=s3://${bucket_name}
```

Create Cluster Kubernetes :

```
kops create cluster --zones=ap-southeast-1 --node-count=3 --master-count=1 --node-size=t2.medium --master-size=t2.large --name=${KOPS_CLUSTER_NAME} --ssh-public-key=~/.ssh/id_rsa.pub
```

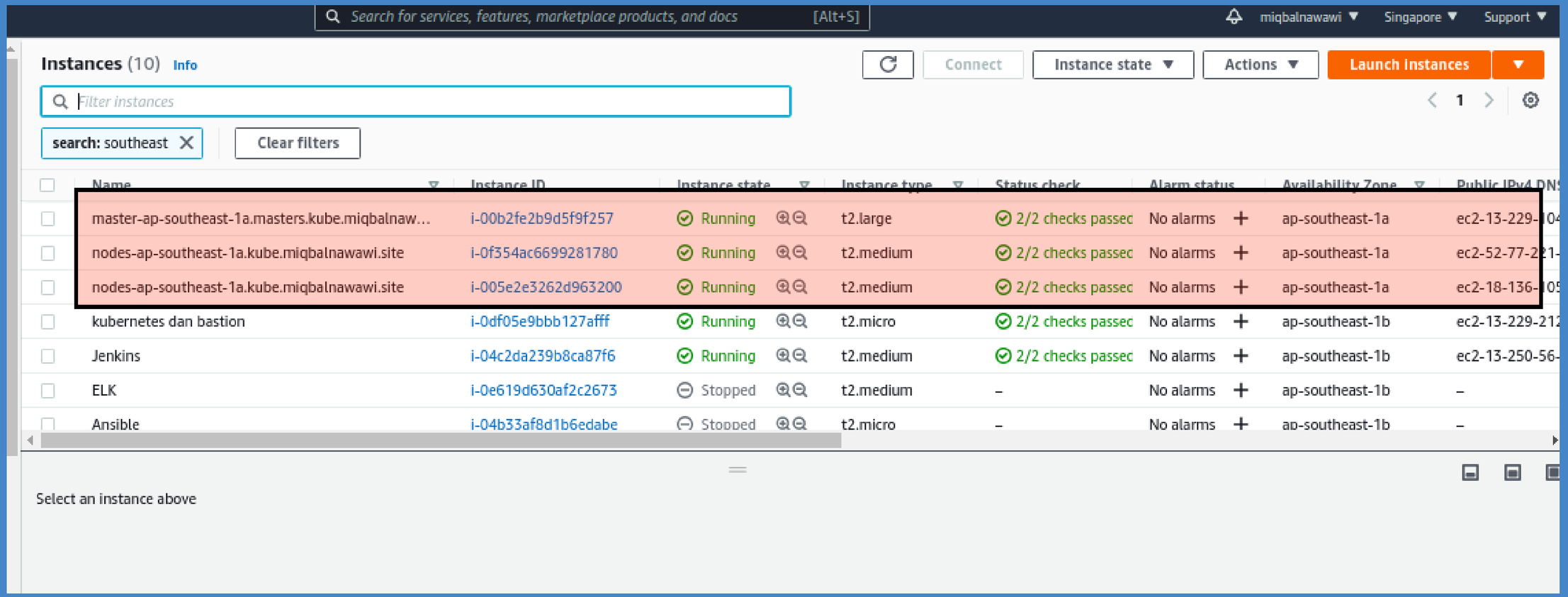
Update Cluster :

```
kops update cluster --name ${KOPS_CLUSTER_NAME} --yes --admin
```

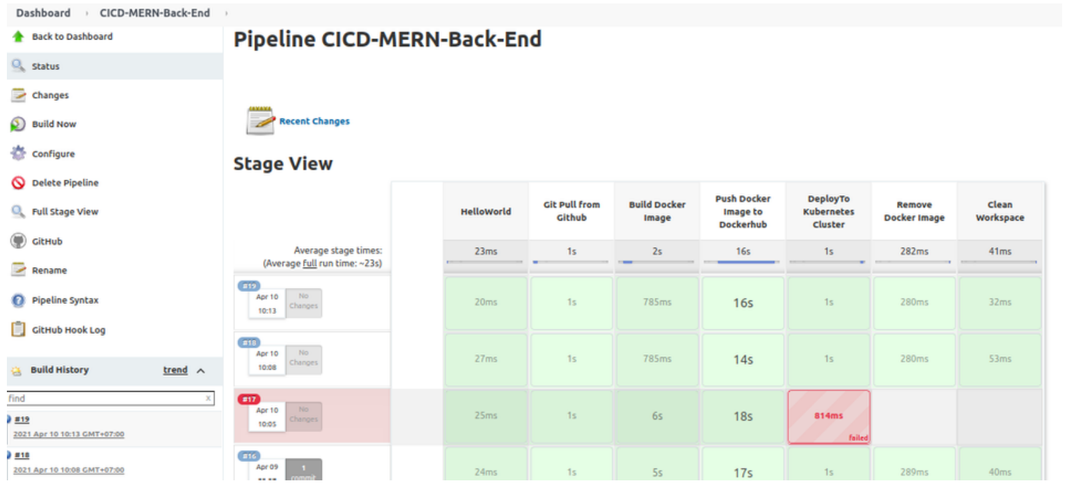
Validate CLuster :

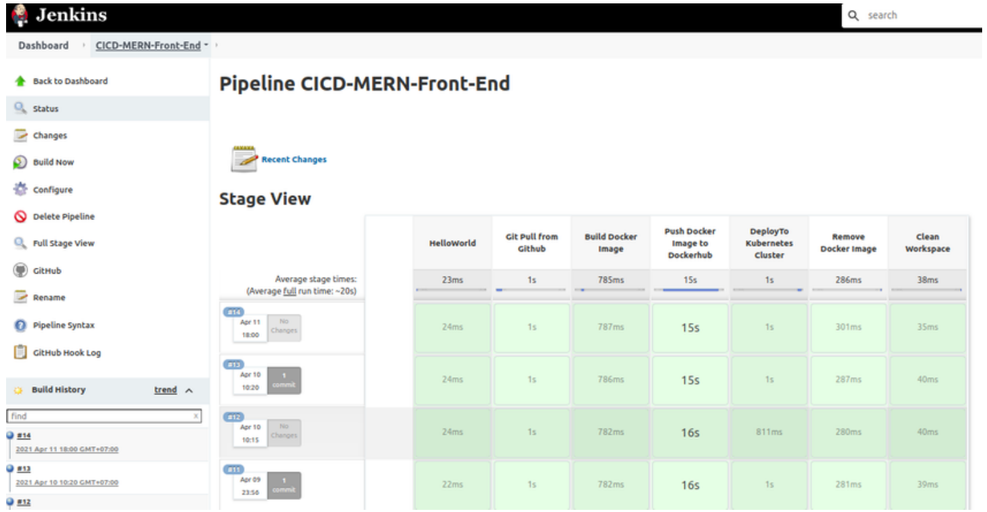
```
kops validate cluster
```

Cluster Created



Build Deployment Backend - Frontend on Jenkins





Pods Production Created

```
root@ip-10-0-1-154:~# kubectl get pods -n production
NAME                             READY   STATUS    RESTARTS   AGE
backend-prod-5d6db44455-tn7h7    1/1     Running   0          147m
backend-prod-5d6db44455-zq9n2    1/1     Running   0          147m
frontend-prod-ddf8b476f-7qt4v    1/1     Running   0          78m
frontend-prod-ddf8b476f-p757j    1/1     Running   0          78m
mongodb                          1/1     Running   0          147m
root@ip-10-0-1-154:~#
```

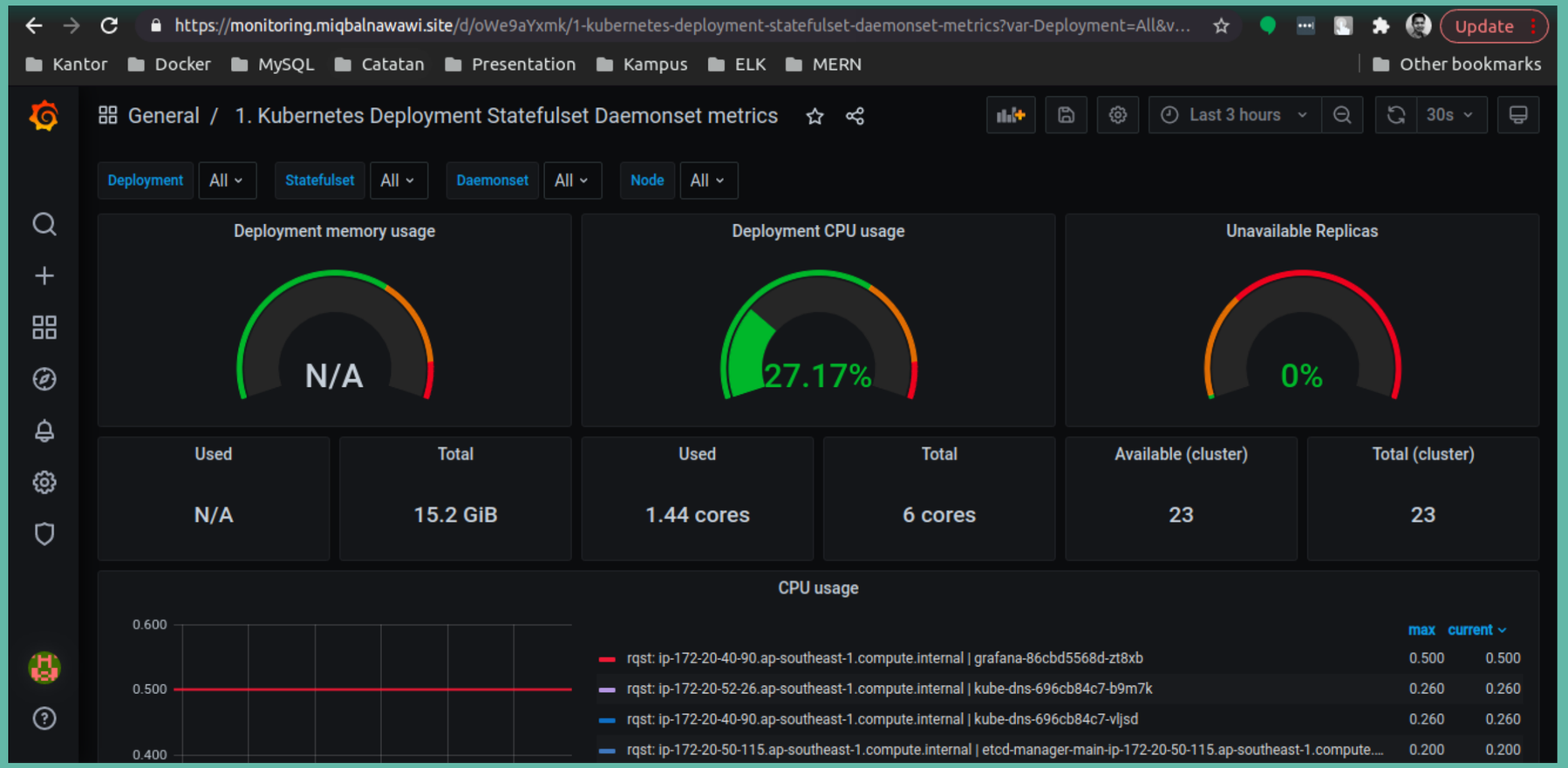
Build Pods Monitoring Grafana – Prometheus

```
root@ip-10-0-1-154:~# kubectl get pods -n monitoring
NAME                          READY   STATUS    RESTARTS   AGE
grafana-86cbd5568d-zt8xb      1/1     Running   0          2d8h
prometheus-9974bdd74-jrm6m    1/1     Running   0          2d8h
root@ip-10-0-1-154:~#
```

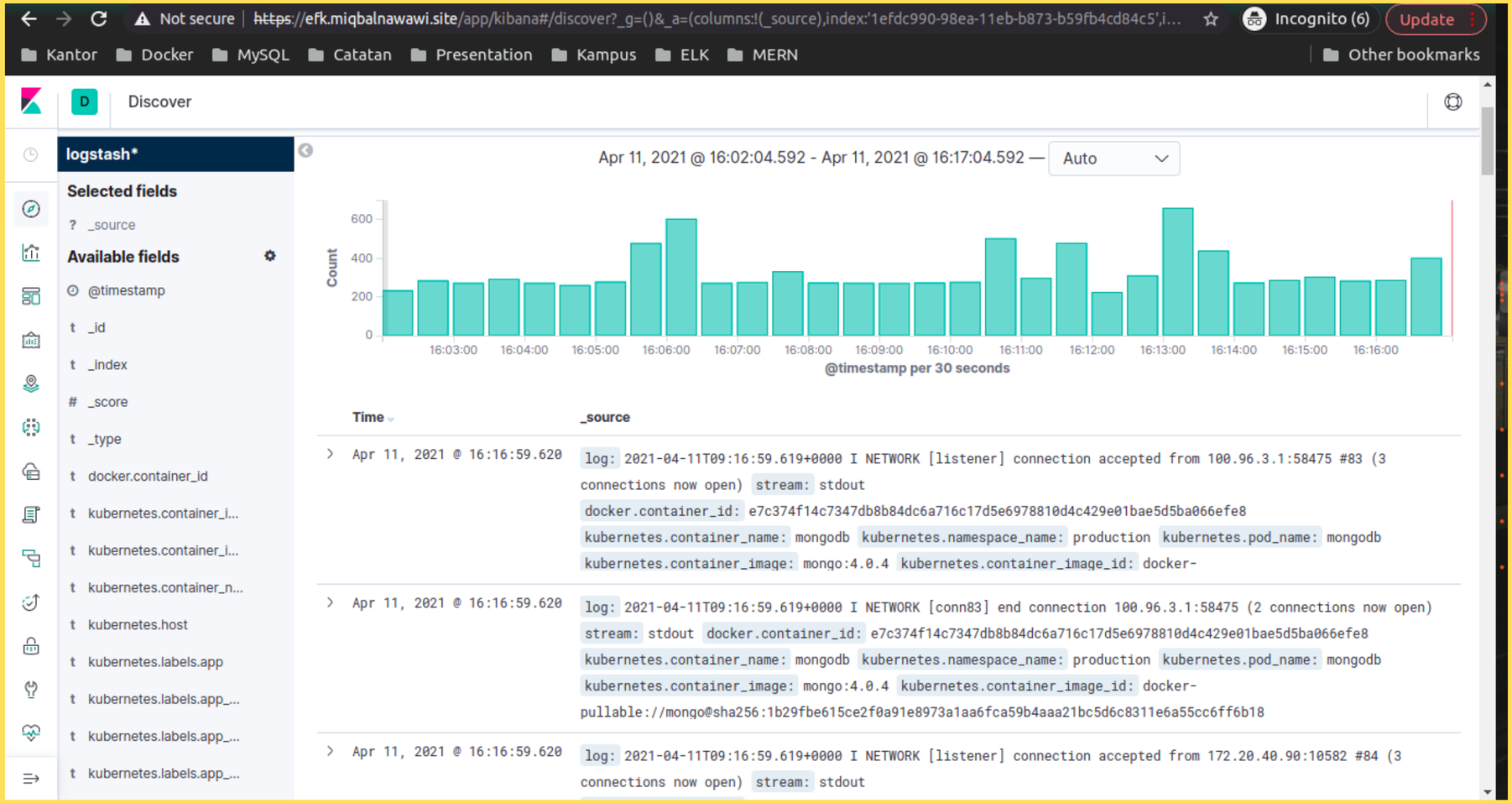
Build Pods Logging Kibana

```
root@ip-10-0-1-154:~# kubectl get pods -n kube-logging
NAME                    READY   STATUS    RESTARTS   AGE
es-cluster-0            1/1     Running   0          42h
es-cluster-1            1/1     Running   0          42h
es-cluster-2            1/1     Running   0          42h
fluentd-5jwrm           1/1     Running   0          2d7h
fluentd-894q9           1/1     Running   0          42h
fluentd-z5bbb           1/1     Running   0          2d7h
kibana-84cf7f59c-5httf  1/1     Running   0          42h
root@ip-10-0-1-154:~#
```

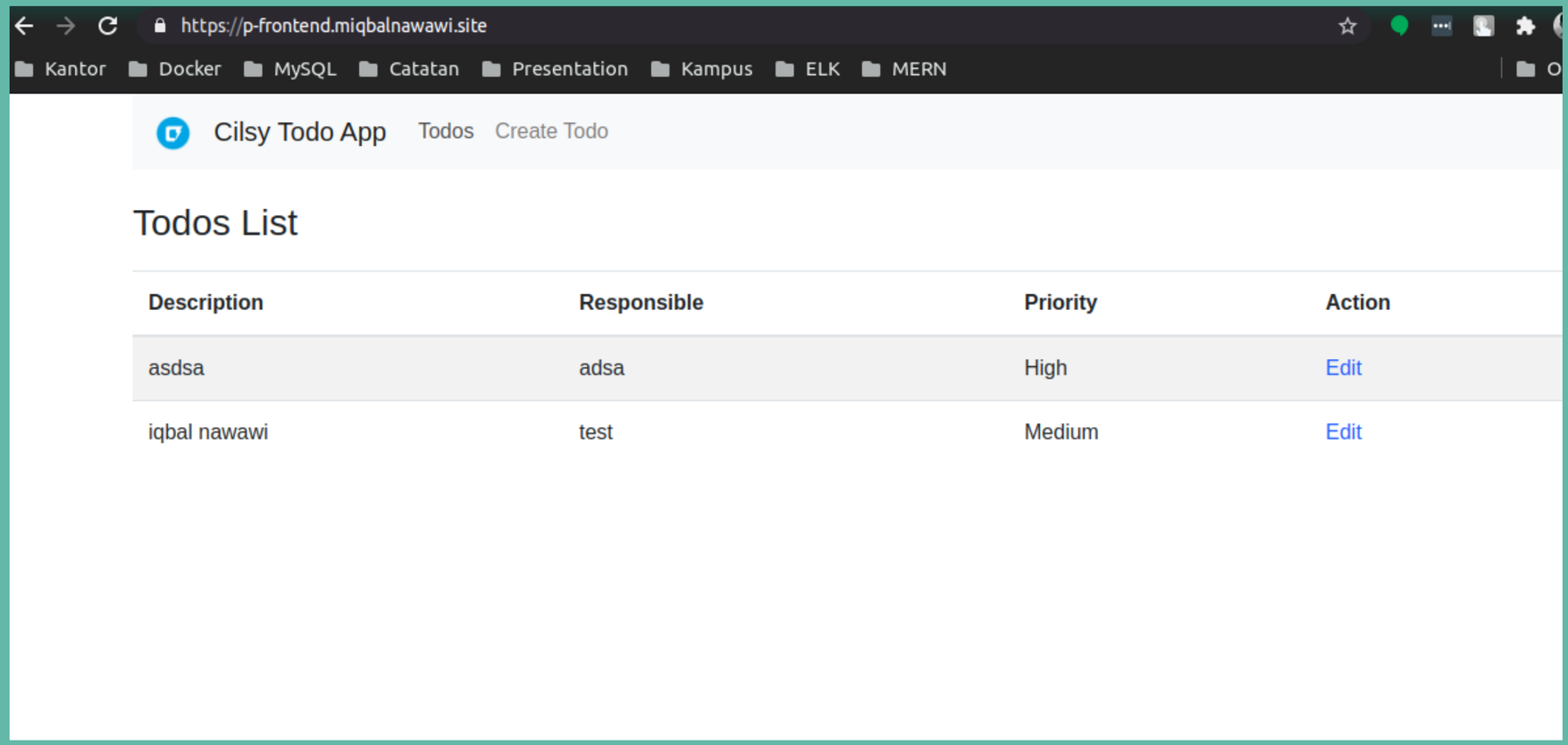
Monitoring With Prometheus



Monitoring With Logging EFK



Page View Frontend



Thank You

Contact Me :

Github : github.com/miqbalnawawi

LinkedIn : linkedin.com/in/muhammad-iqbal-nawawi-73a993165/

Telegram : @miqbalnawawi