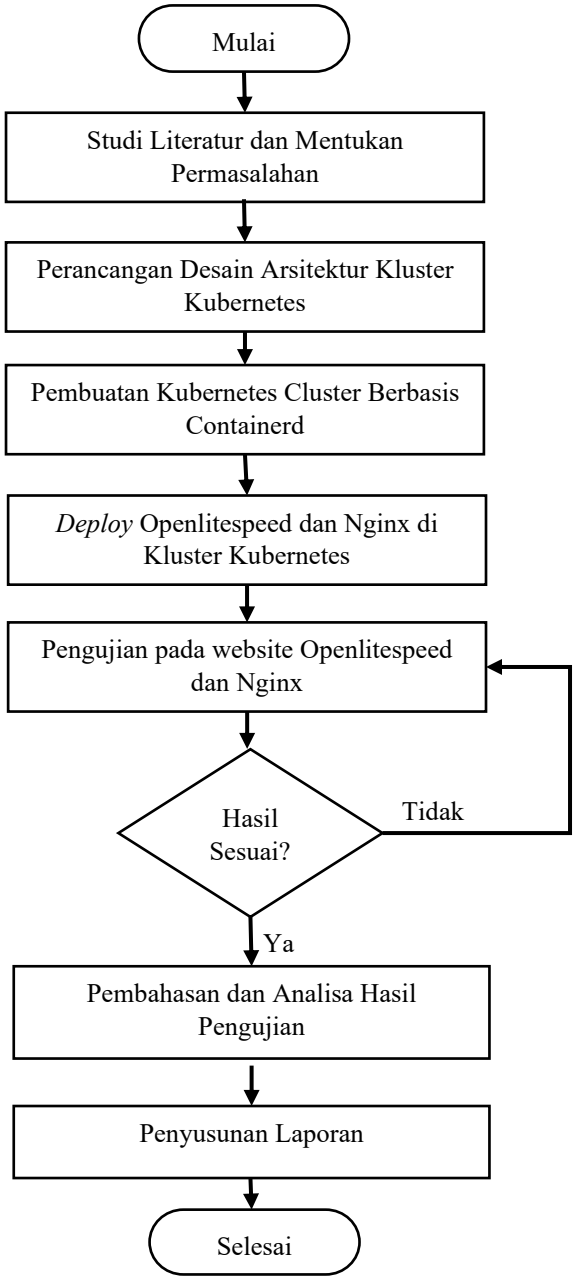
BAB 3
METODE PENELITIAN

3.1 ALUR PENELITIAN

Penelitian ini dilakukan dalam beberapa tahap antara lain tahap studi literature serta menentukan permasalah, tahap merancang kluster, tahap implementasi kluster, tahap *deploy container* diatas kluster, tahap pengujian, tahap pembahasan dan disertai analisis.



Gambar 3.1 Alur Metode Penelitian

Berikut penjelasan mengenai alur penelitian berdaasarkan Gambar 3.1 yang dilakukan dengan beberapa tahap. Mulai dari studi literature yang digunakan untuk mencari pedoman dan referensi penulisan yaitu kajian pustaka dan dasar teori penelitian. Menetapkan rumusan, batasan masalah, tujuan dan manfaat penelitian yang akan dilakukan. Proses tersebut berguna agar kegiatan yang dilakukan tetap sesuai dengan rencana penelitian. Pada perancangan kluster kubernetes menggunakan containerd, kemudian di implementasikan dengan kubernetes. Selanjutnya, tahap *deploy* Openlitespeed dan Nginx pada kluster kubernetes. Tahap pengujian dilakukan dengan membandingkan kedua webserver. Terakhir adalah penyusunan laporan yang sesuai dengan hasil penelitian.

3.2 ALAT DAN BAHAN

3.2.1 Perangkat Keras

Perangkat keras yang diperlukan dalam penelitian ini, terdiri atas 3 perangkat, yaitu *physical server*, virtual *machine* dan laptop. Server yang digunakan secara fisik sebanyak 1 unit, dengan spesifikasi seperti pada tabel 3.1.

Tabel 3.1 Spesifikasi *Physical Server*

Merk	Tipe Model	CPU	*Memory*	*Disk*
Asus	Z10PA-U8	Intel(R) Xeon(R) CPU E5-1650 v3 @ 3.50GHz	256 GB	1 TB

Perangkat yang kedua adalah VM, penggunaannya sebagai kluster kubernetes sebanyak 3 VM dan 1 VM Apache Benchmark digabung tcpdump, dengan spesifikasi seperti pada tabel 3.2.

Tabel 3.2 Spesifikasi Virtual *Machine*

Hostname	Role	vCPU	*Memory*	*Disk*
bib-master	Master, Control Plane	2	2	100GB
bib-worker1	Worker	2	4	100GB
bib-worker2	Worker	2	4	100GB
bib-wonderful	*Tool* Pengujian	1	2	50GB

Perangkat yang ketiga adalah laptop, laptop yang digunakan sebanyak 1 unit. Laptop digunakan untuk melakukan akses dan konfigurasi server, untuk spesifikasi laptop yang digunakan ditampilkan pada Tabel 3.3.

Tabel 3.3 Spesifikasi Laptop

Merk	Tipe Model	CPU	*Memory*	*Disk*
Lenovo	Thinkbook 14-IIL	Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz	8 GB	256 GB

3.2.2 Perangkat Lunak

Perangkat lunak yang digunakan sesuai pada tabel 3.4 yang terdiri dari:

1. **Ubuntu**

   Sebagai sistem operasi yang digunakan pada *virtual machine* untuk kluster kubernetes.

2. **Windows**

   Sebagai sistem operasi yang digunakan pada laptop untuk akses ke *virtual machine*.

3. **Kubernetes**

   Sebagai *container* orkrestasi pada lingkungan kluster kubernetes. Kubernetes yang digunakan adalah versi.

4. **Containerd**

   Sebagai *container runtime* yang digunakan pada lingkungan kluster kubernetes.

5. ***Virtual Machine***

   Sebagai tool untuk membuat virtualisasi diatas server fisik, yang hasil outputnya berupa *virtual machine* yang digunakan kluster kubernetes. Pada penelitian ini, *virtual machine* didukung dengan KVM sebagai *hypervisor,* libvirtd sebagai *tool* untuk mengelola KVM, dan terraform sebagai tool untuk *automation* dalam membuat *virtual machine* dengan mengintegrasikan KVM dan libvirtd.

6. **Windows Terminal**

   Sebagai antarmuka untuk menggunakan perintah unix dan mengakses atau me-*remote virtual machine* pada kluster kubernetes.

7. **Apache Benchmark**

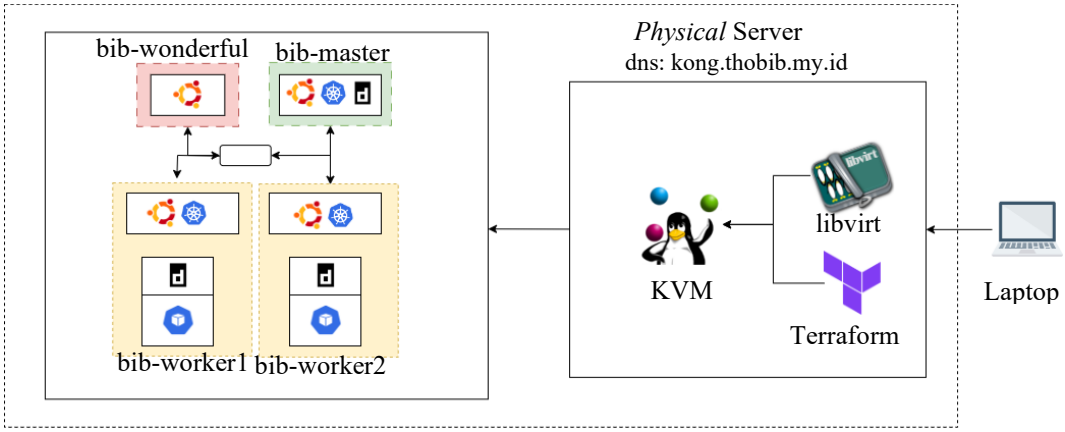
   Sebagai *tool* untuk melakukan pengujian terhadap web server Openlitespeed dan Nginx.

Tabel 3.4 Spesifikasi Perangkat Lunak

No	*Software*	Versi
1.	Ubuntu	20.04
2.	Windows	10 build 22621.1702
3.	Kubernetes	1.17
4.	Containerd	1.6.12
5.	KVM	4.2.1
6.	libvirtd	6.0.0
7.	Terraform	1.3.6
8.	Windows Terminal	1.16.10262.0
9.	Apache Benchmark	2.4.41
10.	Sar	12.2.0-2ubuntu0.1
9.	tcpdump	4.9.3

3.3 PERANCANGAN DESAIN KLUSTER ARSITEKTUR KUBERNETES

Arsitektur yang digunakan pada penelitian ini adalah virtualisasi yang dijalankan pada server fisik sebagai *hypervisor* untuk *virtual machine*. Pada server fisik dijalankan KVM dan libvirtd. *Virtual machine* yang ada pada klutser kubernetes berjalan diatas KVM, dan *virtual machine* tersebut diinstal kubernetes, dengan bib-master sebagai *control plane,* dan bib-worker1 dan bib-worker2 sebagai tempat pod container berjalan. Untuk bib-wonderful sebagai *tool* pengujian.



Gambar 3.2 Arsitektur Yang Digunakan Pada Penelitian

Tabel 3.5 IP *Address* Yang Digunakan Pada Penelitian

Hostname	Interface	IP *Address*
bib-master	ens3	9.8.8.11/24
bib-worker1	ens3	9.8.8.21/24
bib-worker2	ens3	9.8.8.22/24
bib-wonderful	ens3	9.8.8.254/24

3.4 PEMBUATAN KUBERNETES CLUSTER BERBASIS CONTAINERD

Berikut akan dijelaskan proses pembuatan *virtual machine* pada penelitian ini. Pembuatan *virtual machine* menggunakan hypervisor KVM yang ada di server fisik dengan spesifikasi sesuai dengan tabel 3.2. Pada tahap ini, *source code* ditulis untuk membuat VM pada *Physical Server* sesuai dengan *source code* terlampir dibawah. Nantinya proses pembuatan VM ini dilakukan dengan melinbatkan Terraform, Libvirtd dan KVM *Hypervisor*.

```
PUBKEY1:ssh-rsa AAyc2EAAAADAlttc……HheNFeTi6nXSqbgSf root@sokong

[VM1]
NAME: bib-master
OS: focal-server-cloudimg-amd64.img
NESTED: y
VCPUS: 2
MEMORY: 2G
DISK1: 100G
IFACE_NETWORK1: 9.8.8.0
IFACE_IP1: 9.8.8.11
CONSOLE: vnc

[VM2]
NAME: bib-worker1
OS: focal-server-cloudimg-amd64.img
NESTED: y
VCPUS: 2
MEMORY: 4G
DISK1: 100G
IFACE_NETWORK1: 9.8.8.0
IFACE_IP1: 9.8.8.21
CONSOLE: vnc

[VM3]
NAME: bib-worker2
OS: focal-server-cloudimg-amd64.img
NESTED: y
VCPUS: 2
MEMORY: 4G
```

```
DISK1: 100G
IFACE_NETWORK1: 9.8.8.0
IFACE_IP1: 9.8.8.22
CONSOLE: vnc
```

Selanjutnya, *source code* tersebut di jalankan terraform untuk memproses pembuatan *virtual machine*, dan nanti hasil outputnya seperti dibawah ini.

```
root@sokong ~ # virsh list
 Id    Name                 State
------------------------------------
 427   bib-master           running
 428   bib-worker1          running
 429   bib-worker2          running
```

Selanjutnya, melakukan instalasi kluster kubernetes untuk semua VM. Di mulai dengan menambahkan konfigurasi *networking* untuk semua VM.

```
root@bib-master:~# sudo timedatectl set-timezone Asia/Jakarta
root@bib-master:~# cat <<EOF | sudo tee /etc/modules-
load.d/containerd.conf
overlay
br_netfilter
EOF
root@bib-master:~# sudo modprobe overlay
root@bib-master:~# sudo modprobe br_netfilter"
root@bib-master:~# cat <<EOF | sudo tee /etc/sysctl.d/99-
kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF
root@bib-master:~# sudo sysctl –system
----
root@bib-worker1:~# sudo timedatectl set-timezone Asia/Jakarta
root@bib-worker1:~# cat <<EOF | sudo tee /etc/modules-
load.d/containerd.conf
overlay
br_netfilter
EOF
root@bib-worker1:~# sudo modprobe overlay
root@bib-worker1:~# sudo modprobe br_netfilter"
root@bib-worker1:~# cat <<EOF | sudo tee /etc/sysctl.d/99-
kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF
root@bib-worker1:~# sudo sysctl –system
---
root@bib-worker2:~# sudo timedatectl set-timezone Asia/Jakarta
root@bib-worker2:~# cat <<EOF | sudo tee /etc/modules-
load.d/containerd.conf
overlay
br_netfilter
```

```
EOF
root@bib-worker2:~# sudo modprobe overlay
root@bib-worker2:~# sudo modprobe br_netfilter"
root@bib-worker2:~# cat <<EOF | sudo tee /etc/sysctl.d/99-
kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF
root@bib-worker2:~# sudo sysctl --system
```

Tahap selanjutnya, melakukan instalasi Containerd pada ketiga VM yang nantinya digunakan untuk wadah dari pod.

```
root@bib-master:~# sudo apt-get update && sudo apt-get install -y
containerd
root@bib-master:~# sudo mkdir -p /etc/containerd
root@bib-master:~# sudo containerd config default | sudo tee
/etc/containerd/config.toml
root@bib-master:~# sudo systemctl restart containerd
root@bib-master:~# sudo systemctl enable containerd
```

Tahap selanjutnya, melakukan instalasi kubernetes di semua VM, tapi sebelumnya swap pada linux perlu di matikan, karena akan menyebabkan pod gagal. Setelah proses instalasi selesai, kubernetes perlu di *hold* terlebih dahulu, karena pada kubernetes belum disetting untuk subnet yang digunakan oleh pod dan *endpoint* pada master.

```
root@bib-master:~# sudo swapoff -a
root@bib-master:~# sudo sed -i '/ swap / s/^\(.*\)$/#\1/g'
/etc/fstab
root@bib-master:~# curl -s
https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key
add -
cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
root@bib-master:~# sudo apt update
root@bib-master:~# sudo apt-get install -y kubelet kubeadm
kubectl
root@bib-master:~# sudo apt-mark hold kubelet kubeadm kubectl
```

Tahap selanjutnya, menambahkan endpoint pada master dan subnet untuk pod yang nantinya digunakan sebagai *network* pada setiap pod. Untuk proses ini hanya berlaku di node master.

```
root@bib-master:~# cat kubeadm-config.yaml
apiVersion: kubeadm.k8s.io/v1beta2
kind: ClusterConfiguration
kubernetesVersion: stable
```

```
controlPlaneEndpoint: "9.8.8.11:6443"
networking:
    podSubnet: "10.10.0.0/16"
root@bib-master:~# kubeadm init --config=kubeadm-config.yaml --
upload-certs
```

Tahap selanjutnya, menambahkan bib-worker1 dan bib-worker2 ke kluster kubernetes yang telah diinisialiasi oleh bib-master. Langkah ini cukup di eksekusi pada VM worker.

```
root@bib-worker1:~# kubeadm join 9.8.8.11:6443 --token
nuge16.qwof3f58d9audunf \
> --discovery-token-ca-cert-hash
sha256:86544638e5c4effaf4be3f911553270f63f5939da780063df6299c27f9
e39a65

root@bib-worker2:~# kubeadm join 9.8.8.11:6443 --token
nuge16.qwof3f58d9audunf \
> --discovery-token-ca-cert-hash
sha256:86544638e5c4effaf4be3f911553270f63f5939da780063df6299c27f9
e39a65
Run 'kubectl get nodes' on the control-plane to see this node
join the cluster.
```

Tahap selanjutnya, men-*deploy* CNI calico sebagai network interface yang nanti digunakan oleh pod. Sebelum men-*deploy* CNI, node berstatus *NotReady* dikarenakan pod *dependency* kubernetes tidak memiliki *network interface*.

```
root@bib-master:~# kubectl get nodes
NAME          STATUS     ROLES           AGE     VERSION
bib-master    NotReady   control-plane   14m     v1.17.1
bib-worker1   NotReady   <none>          9m46s   v1.17.1
bib-worker2   NotReady   <none>          8m59s   v1.17.1

root@bib-master:~# kubectl apply -f
https://docs.projectcalico.org/v3.17/manifests/calico.yaml
```

Tahap selanjutnya, coba kembali cek status node pada kubernetes, dan pastikan semua pod *dependency* kubernetes dan CNI Calico sudah *running* semua

```
root@bib-master:~#  kubectl get pod -A
NAMESPACE     NAME                                       READY
STATUS    RESTARTS   AGE
kube-system   calico-kube-controllers-77bcf68848-2xrg5   1/1
Running   0          19m
kube-system   calico-node-kclxx                          1/1
Running   0          19m
kube-system   calico-node-qbqtb                          1/1
Running   0          19m
kube-system   calico-node-wkhzm                          1/1
Running   0          19m
```

```
kube-system   coredns-5d78c9869d-4fxp6                  1/1
Running   0          33m
kube-system   coredns-5d78c9869d-r24q7                  1/1
Running   0          33m
kube-system   etcd-bib-master                           1/1
Running   0          33m
kube-system   kube-apiserver-bib-master                 1/1
Running   0          33m
kube-system   kube-controller-manager-bib-master        1/1
Running   0          33m
kube-system   kube-proxy-8t42l                          1/1
Running   0          29m
kube-system   kube-proxy-gcthd                          1/1
Running   0          33m
kube-system   kube-proxy-td6jl                          1/1
Running   0          28m
kube-system   kube-scheduler-bib-master                 1/1
Running   0          33m
root@bib-master:~# kubectl get nodes
NAME          STATUS   ROLES           AGE   VERSION
bib-master    Ready    control-plane   17m   v1.17.1
bib-worker1   Ready    <none>          12m   v1.17.1
bib-worker2   Ready    <none>          11m   v1.17.1
```

3.5 *DEPLOY* OPENLITESPEED DAN NGINX DI KLUSTER KUBERNETES

Pada tahap ini, perlu membuat *source code* untuk mendefinisikan *deployment* yang nanti akan digunakan. Pertama, untuk pod openlitespeed dengan nama *file* openlitespeed-deploy.yaml, seperti kode dibawah ini

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: openlitespeed-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: openlitespeed
  template:
    metadata:
      labels:
        app: openlitespeed
    spec:
      containers:
      - name: openlitespeed
        image: litespeedtech/openlitespeed
        ports:
        - containerPort: 80
        volumeMounts:
        - name: openlitespeed-config
          mountPath: /var/www/vhosts/localhost/html
      volumes:
```

```
            - name: openlitespeed-config
              configMap:
                name: openlitespeed-configmap
---
      ---
      apiVersion: v1
      kind: ConfigMap
      metadata:
        name: openlitespeed-configmap
      data:
        index.html: |
          <html>
          <head>
            <title>Webpage Openlitespeed</title>
          </head>
          <body>
            <h1>Example webpage index.html</h1>
          </body>
          </html>
        lulus.html: |
          <!DOCTYPE html>
          <html>
          <head>
            <title>Portofolio Landing Page</title>
            <style>
              body {
                margin: 0;
                padding: 0;
                font-family: Arial, sans-serif;
              }

              .navbar {
                background-color: rgba(0, 0, 0, 0.5);
                position: absolute;
                top: 0;
                right: 0;
                padding: 10px;
                z-index: 1;
              }

              .navbar ul {
                list-style-type: none;
                margin: 0;
                padding: 0;
                text-align: center;
              }

              .navbar li {
                display: inline-block;
                margin-right: 10px;
              }

              .navbar a {
                color: #fff;
                text-decoration: none;
                padding: 5px 10px;
                border-radius: 5px;
```

```
                font-size: 18px;
              }

              .navbar a:hover {
                background-color: #fff;
                color: #333;
              }

              .container {
                background-image:
url('https://images.unsplash.com/photo-1682687220742-
aba13b6e50ba?ixlib=rb-
4.0.3&q=85&fm=jpg&crop=entropy&cs=srgb&dl=neom-gOqBe7ropxM-
unsplash.jpg');
                background-size: cover;
                background-position: center;
                height: calc(100vh - 40px);
                display: flex;
                align-items: center;
                justify-content: center;
                text-align: center;
              }

              .content {
                background-color: rgba(255, 255, 255, 0.7);
                padding: 20px;
                border-radius: 10px;
              }

              h1 {
                font-size: 36px;
                color: #333;
                margin-bottom: 20px;
              }

              p {
                font-size: 18px;
                color: #666;
                margin-bottom: 10px;
              }
            </style>
          </head>
          <body>
            <div class="navbar">
              <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#">About</a></li>
                <li><a href="#">Portfolio</a></li>
                <li><a href="#">Contact</a></li>
              </ul>
            </div>
            <div class="container">
              <div class="content">
                <h1>Thobib Khoirul Annas</h1>
                <p>This   is   a   Website   for   Benchmark   of
Openlitespeed And Nginx.</p>
                <p>Wish You A Beautiful Day</p>
```

```
              </div>
            </div>
          </body>
          </html>
---
       apiVersion: v1
       kind: Service
       metadata:
         name: openlitespeed-service
       spec:
         selector:
           app: openlitespeed
         ports:
         - name: http
           port: 80
           targetPort: 80
  type: LoadBalancer
```

Tahap selanjutnya, untuk untuk pod nginx dengan nama *file* nginx-deploy.yaml, sesuai dengan kode dibawah ini

```
       apiVersion: apps/v1
       kind: Deployment
       metadata:
         name: nginx-deployment
       spec:
         replicas: 1
         selector:
           matchLabels:
             app: nginx
         template:
           metadata:
             labels:
               app: nginx
           spec:
             containers:
             - name: nginx
               image: nginx:latest
               ports:
               - containerPort: 80
               volumeMounts:
               - name: nginx-config
                 mountPath: /usr/share/nginx/html
             volumes:
             - name: nginx-config
               configMap:
                 name: nginx-configmap
       ---
       apiVersion: v1
       kind: ConfigMap
       metadata:
         name: nginx-configmap
       data:
         index.html: |
           <html>
           <head>
```

```
            <title>Webpage Nginx</title>
          </head>
          <body>
            <h1>Example webpage index.html</h1>
          </body>
          </html>
        lulus.html: |
          <!DOCTYPE html>
          <html>
          <head>
            <title>Portofolio Landing Page</title>
            <style>
              body {
                margin: 0;
                padding: 0;
                font-family: Arial, sans-serif;
              }

              .navbar {
                background-color: rgba(0, 0, 0, 0.5);
                position: absolute;
                top: 0;
                right: 0;
                padding: 10px;
                z-index: 1;
              }

              .navbar ul {
                list-style-type: none;
                margin: 0;
                padding: 0;
                text-align: center;
              }

              .navbar li {
                display: inline-block;
                margin-right: 10px;
              }

              .navbar a {
                color: #fff;
                text-decoration: none;
                padding: 5px 10px;
                border-radius: 5px;
                font-size: 18px;
              }

              .navbar a:hover {
                background-color: #fff;
                color: #333;
              }

              .container {
                background-image:
url('https://images.unsplash.com/photo-1682687220742-
aba13b6e50ba?ixlib=rb-
```

```
4.0.3&q=85&fm=jpg&crop=entropy&cs=srgb&dl=neom-gOqBe7ropxM-
unsplash.jpg');
                background-size: cover;
                background-position: center;
                height: calc(100vh - 40px);
                display: flex;
                align-items: center;
                justify-content: center;
                text-align: center;
              }

              .content {
                background-color: rgba(255, 255, 255, 0.7);
                padding: 20px;
                border-radius: 10px;
              }

              h1 {
                font-size: 36px;
                color: #333;
                margin-bottom: 20px;
              }

              p {
                font-size: 18px;
                color: #666;
                margin-bottom: 10px;
              }
            </style>
          </head>
          <body>
            <div class="navbar">
              <ul>
                <li><a href="#">Home</a></li>
                <li><a href="#">About</a></li>
                <li><a href="#">Portfolio</a></li>
                <li><a href="#">Contact</a></li>
              </ul>
            </div>
            <div class="container">
              <div class="content">
                <h1>Thobib Khoirul Annas</h1>
                <p>This   is   a   Website   for   Benchmark   of
Openlitespeed And Nginx.</p>
                <p>Wish You A Beautiful Day</p>
              </div>
            </div>
          </body>
          </html>
      ---
      apiVersion: v1
      kind: Service
      metadata:
        name: nginx-service
      spec:
        selector:
          app: nginx
```

```
    ports:
    - name: http
      port: 80
      targetPort: 80
    type: LoadBalancer
```

Tahap selanjutnya, melakukan *apply deployment* kedua pod openlitespeed dan nginx, sesuai perintah dibawah ini

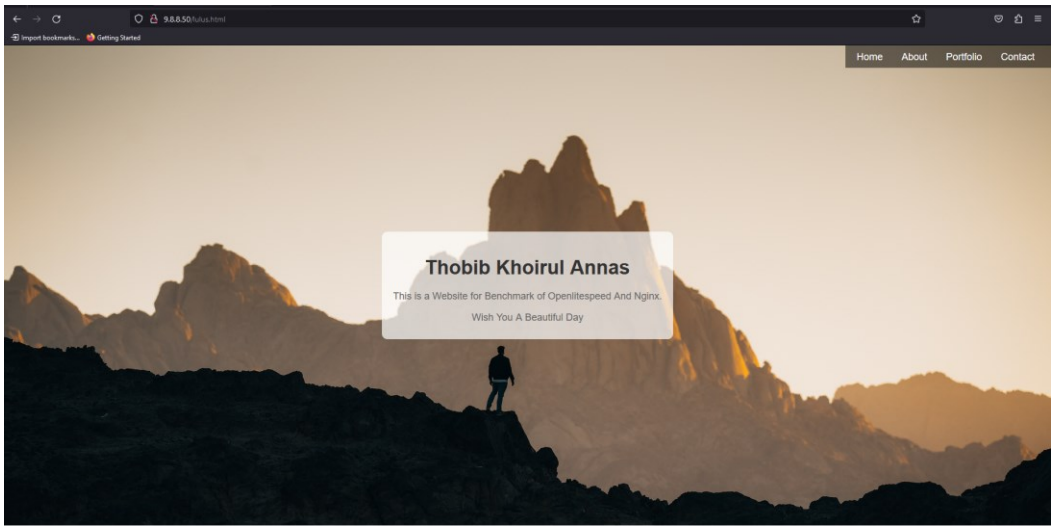
```
root@bib-master:~# kubectl apply -f openlitespeed-deploy.yaml

root@bib-master:~# kubectl apply -f nginx/nginx-deploy.yaml
```

Berikut kondisi pod sudah berjalan, yang menandakan proses *deployment* telah berhasil, sesuai tampilan dibawah ini

```
root@bib-master:~# kubectl get pod
NAME                                      READY   STATUS    RESTARTS
nginx-deployment-849b86845b-psc2j           1/1     Running   1
openlitespeed-deployment-cbd4cd759-dks6s    1/1     Running   0
root@bib-master:~# kubectl get service
NAME                    CLUSTER-IP       EXTERNAL-IP   PORT(S)
kubernetes              10.96.0.1        <none>        443/TCP
nginx-service          10.109.231.31    9.8.8.50      80:30721/TCP
openlitespeed-service   10.100.114.245   9.8.8.51      80:32306/TCP
```

Selanjutnya uji coba akses halaman web lulus.html seperti gambar 3.3, yang di *deploy* pada pod openlitespeed dan nginx.



Gambar 3.3 Tampilan Web lulus.html

3.6 SKENARIO PENGUJIAN PADA WEBSITE OPENLITESPEED DAN NGINX

Pada bagian ini, penelitian dilakukan perlu dilakukan pengujian untuk mengevaluasi performansi kontainer Openlitespeed dan Nginx dengan menggunakan Apache Benchmark sebagai *tool* untuk membuat jumlah koneksi, yang di instal pada VM *bib-wonderful*. Sar sebagai monitor CPU *usage* dan *memory usage* di install pada kedua pod web server, dengan perintah sesuai dibawah ini

```
root@bib-wonderful:~# apt install ab
---
root@nginx-deployment-pod:/# apt install sysstat
root@openlitespeed-deployment-pod:/# apt install sysstat
```

Tcpdump sebagai *capture* lalu lintas packet data, yang *tool* tersebut bawaan dari sisem operasi Linux Ubuntu. Skenario pengujian telah dirancang dengan variasi jumlah koneksi yang berbeda dengan variasi jumlah koneksi sesuai pada tabel 3.6 Nantinya dari hasil pengujian akan didapatkan parameter yang meliputi: pengujian waktu respon, CPU *Usage, memory usage, throughput, packet loss,* dan *delay.*

Tabel 3.6 Skenario Pengujian

No	Jumlah Koneksi	Request per detik
1	1000	100
2	2500	100
3	5000	100
4	7500	100
5	10000	100