LAPORAN

PROYEK SISTEM TERDISTRIBUSI

HIGH AVAILABLE WEB SERVER

Disusun Oleh:

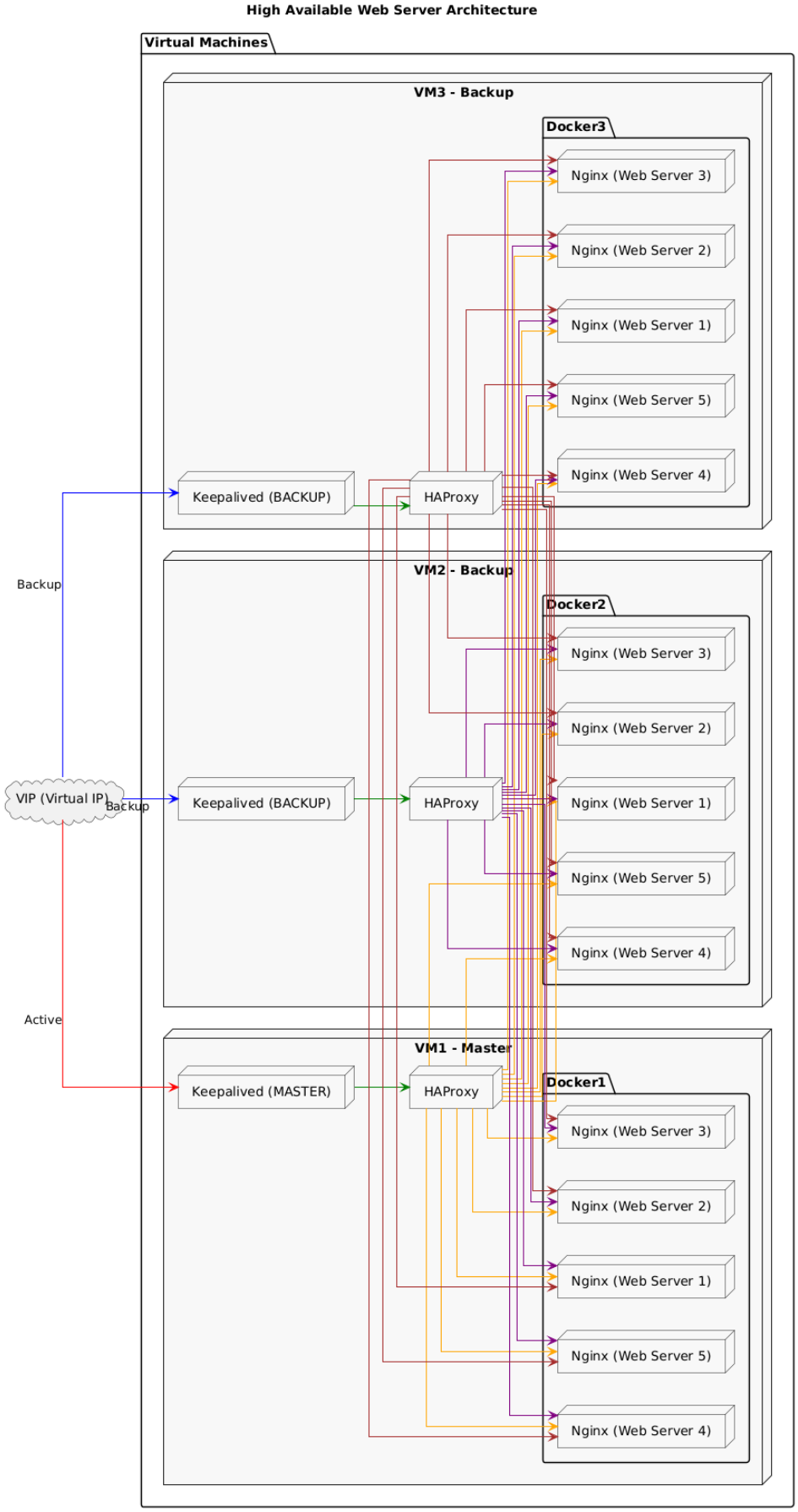
Alfath Tauhidillah - A11.2022.14573

FAKULTAS ILMU KOMPUTER

UNIVERSITAS DIAN NUSWANTORO

2024

ARSITEKTUR


High Available Web Server Architecture
Virtual Machines
VM3 - Backup
Docker3
Nginx (Web Server 3)
Nginx (Web Server 2)
Nginx (Web Server 1)
Nginx (Web Server 5)
Nginx (Web Server 4)
Keepalived (BACKUP)
HAProxy
VM2 - Backup
Docker2
Nginx (Web Server 3)
Nginx (Web Server 2)
Nginx (Web Server 1)
Nginx (Web Server 5)
Nginx (Web Server 4)
Keepalived (BACKUP)
HAProxy
VM1 - Master
Docker1
Nginx (Web Server 3)
Nginx (Web Server 2)
Nginx (Web Server 1)
Nginx (Web Server 5)
Nginx (Web Server 4)
Keepalived (MASTER)
HAProxy
Backup
VIP (Virtual IP)
Backup
Active

PROSES PEMBUATAN

1. Server

   Membuat 5 web server nginx menggunakan docker container:

   a. Install docker

      i. menambahkan repositori docker

```
alfath@server1:~$ # Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/debian \
  $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

      ii. install docker

OBJ

```
alfath@server1:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin_
```

      iii. memberi ijin user untuk mengelola docker

```
alfath@server1:~$ sudo usermod -aG docker alfath_
```

      keluar lalu login kembali untuk memastikan ijin user sudah teraplikasi

      iv. tes docker

```
alfath@server1:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

   b. Membuat container

      i. membuat file compose.yml

```
alfath@server1:~/project$ mkdir web1
alfath@server1:~/project$ cd web1/
alfath@server1:~/project/web1$ nano compose.yml
```

```
  GNU nano 7.2                                                compose.yml
services:
  nginx:
    image: nginx:latest
    container_name: app1
    restart: always
    ports:
      - "8081:80"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
      - ./html:/usr/share/nginx/html:ro
      - ./logs:/var/log/nginx
```

      ii. membuat file nginx.conf

```
alfath@server1:~/project/web1$ nano nginx.conf_
```

```
 GNU nano 7.2                                                    nginx.conf *
events {}

http {
    server {
        listen 80;
        server_name localhost;

        location / {
            root /usr/share/nginx/html;
            index index.html;
        }

        location /server-info {
            return 200 "Server 1 - NGINX\n";
        }

        error_page 404 /404.html;
        location = /404.html {
            root /usr/share/nginx/html;
        }
    }
}_
```

iii. copy folder

```
alfath@server1:~/project/web1$ cd ..
alfath@server1:~/project$ cp -r web1 web2
alfath@server1:~/project$ cp -r web1 web3
alfath@server1:~/project$ cp -r web1 web4
alfath@server1:~/project$ cp -r web1 web5
```

iv. edit port dan container_name pada file compose.yml untuk setiap folder hasil copy

```
alfath@server1:~/project/web2$ nano compose.yml_
```

```
 GNU nano 7.2                                                    compose.yml *
services:
  nginx:
    image: nginx:latest
    container_name: app1
    restart: always
    ports:
      - "8082:80"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
      - ./html:/usr/share/nginx/html:ro
      - ./logs:/var/log/nginx
```

web1 = 8081:80 dan app1
web2 = 8082:80 dan app2
web3 = 8083:80 dan app3
web4 = 8084:80 dan app4

web5 = 8085:80 dan app5

v. edit nama server pada file nginx.conf untuk setiap folder hasil copy

```
alfath@server1:~/project/web2$ nano nginx.conf_
```

```
location /server-info {
    return 200 "Server 2_- NGINX\n";
}
```

web2 = Server 2
web3 = Server 3
web4 = Server 4
web5 = Server 5

vi. buat container

```
alfath@server1:~/project/web1$ docker compose up -d
 [+] Running 2/2
  ✔ Network web1_default  Created                                    0.1s
  ✔ Container app1        Started                                    0.6s
alfath@server1:~/project/web1$ cd ../web2
alfath@server1:~/project/web2$ docker compose up -d
 [+] Running 2/2
  ✔ Network web2_default  Created                                    0.2s
  ✔ Container app2        Started                                    0.4s
alfath@server1:~/project/web2$ cd ../web3
alfath@server1:~/project/web3$ docker compose up -d
 [+] Running 2/2
  ✔ Network web3_default  Created                                    0.2s
  ✔ Container app3        Started                                    0.4s
alfath@server1:~/project/web3$ cd ../web4
alfath@server1:~/project/web4$ docker compose up -d
 [+] Running 2/2
  ✔ Network web4_default  Created                                    0.2s
  ✔ Container app4        Started                                    0.6s
alfath@server1:~/project/web4$ cd ../web5
alfath@server1:~/project/web5$ docker compose up -d
 [+] Running 2/2
  ✔ Network web5_default  Created                                    0.2s
  ✔ Container app5        Started                                    0.4s
```

vii. mengubah ijin file container

```
alfath@server1:~/project$ sudo chown -R alfath:alfath web1
alfath@server1:~/project$ sudo chown -R alfath:alfath web2
alfath@server1:~/project$ sudo chown -R alfath:alfath web3
alfath@server1:~/project$ sudo chown -R alfath:alfath web4
alfath@server1:~/project$ sudo chown -R alfath:alfath web5
```

viii. membuat file index.html

```
alfath@server1:~/project/web1$ cd html/
alfath@server1:~/project/web1/html$ nano index.html_
```

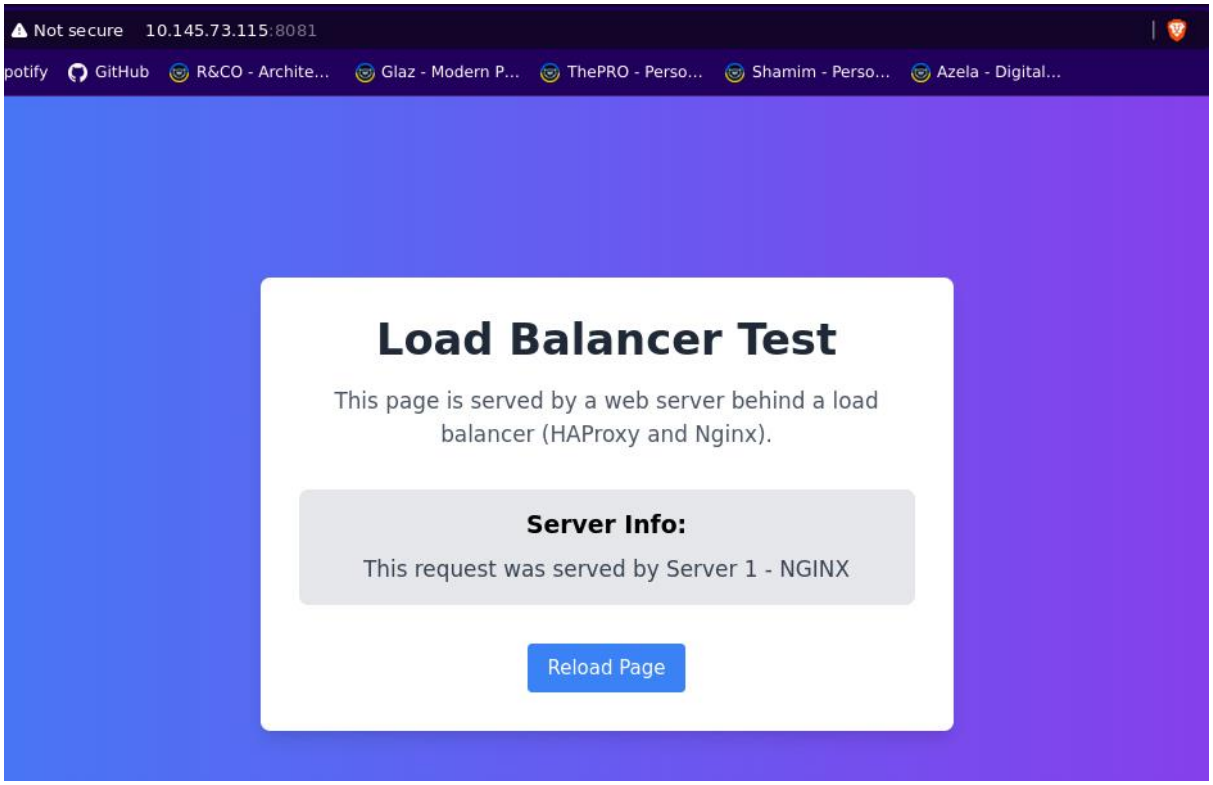
```
 GNU nano 7.2                                                  index.html *
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Load Balancer Test</title>
    <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gray-100 font-sans antialiased">

    <div class="flex items-center justify-center min-h-screen bg-gradient-to-r from-blue-500 to-purple-600">
        <div class="text-center p-8 bg-white rounded-lg shadow-lg max-w-xl w-full">
            <h1 class="text-4xl font-bold text-gray-800 mb-4">Load Balancer Test</h1>
            <p class="text-lg text-gray-600 mb-8">This page is served by a web server behind a load balancer (HAProxy and Nginx).</p>

            <div class="bg-gray-200 p-4 rounded-lg">
                <p class="text-xl font-semibold">Server Info:</p>
                <p id="server-info" class="text-lg text-gray-700 mt-2">Loading...</p>
            </div>

            <div class="mt-8">
                <button id="reload-btn" class="px-4 py-2 bg-blue-500 text-white rounded hover:bg-blue-700 transition">
                    Reload Page
                </button>
            </div>
        </div>
    </div>

    <script>
        // Simulate server response
        document.addEventListener('DOMContentLoaded', () => {
            const serverInfo = document.getElementById('server-info');

            // Simulate fetching the server name or ID (could come from backend)
            fetch('/server-info')
                .then(response => response.text())
                .then(data => {
                    serverInfo.textContent = `This request was served by ${data}`;
                })
                .catch(error => {
                    serverInfo.textContent = "Error fetching server info.";
                });

            // Reload button functionality (optional)
            const reloadBtn = document.getElementById('reload-btn');
            reloadBtn.addEventListener('click', () => {
                location.reload();
            });
        });
    </script>

</body>
</html>
```

ix. copy file index.html ke setiap folder

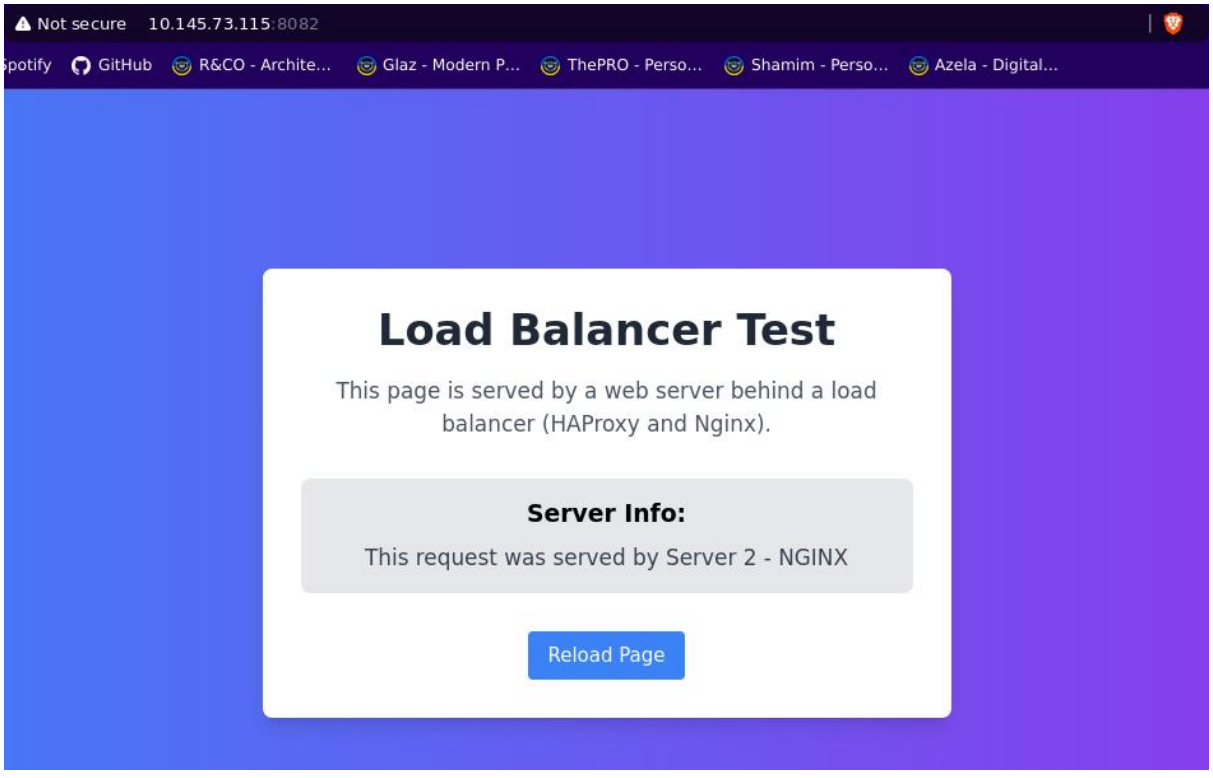
```
alfath@server1:~/project/web1/html$ cp index.html ../../web2/html/index.html
alfath@server1:~/project/web1/html$ cp index.html ../../web3/html/index.html
alfath@server1:~/project/web1/html$ cp index.html ../../web4/html/index.html
alfath@server1:~/project/web1/html$ cp index.html ../../web5/html/index.html
```

c. testing

i. <alamat ip server>:8081 pada web browser



<alamat ip server>:8082 pada web browser



dan seterusnya.

2. Load Balancer

   Membuat load balancer menggunakan haproxy:

   a. install haproxy

```
alfath@server1:~/project$ sudo apt install haproxy_
```

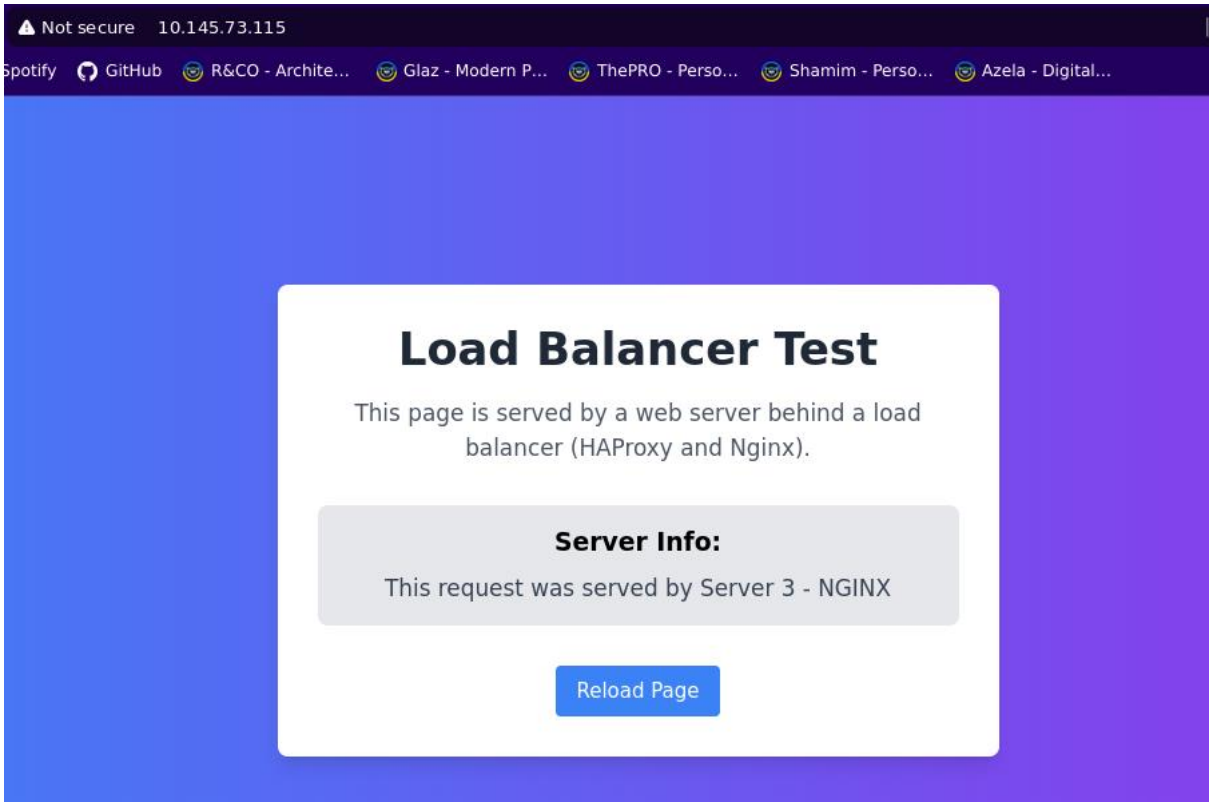
   b. konfigurasi haproxy

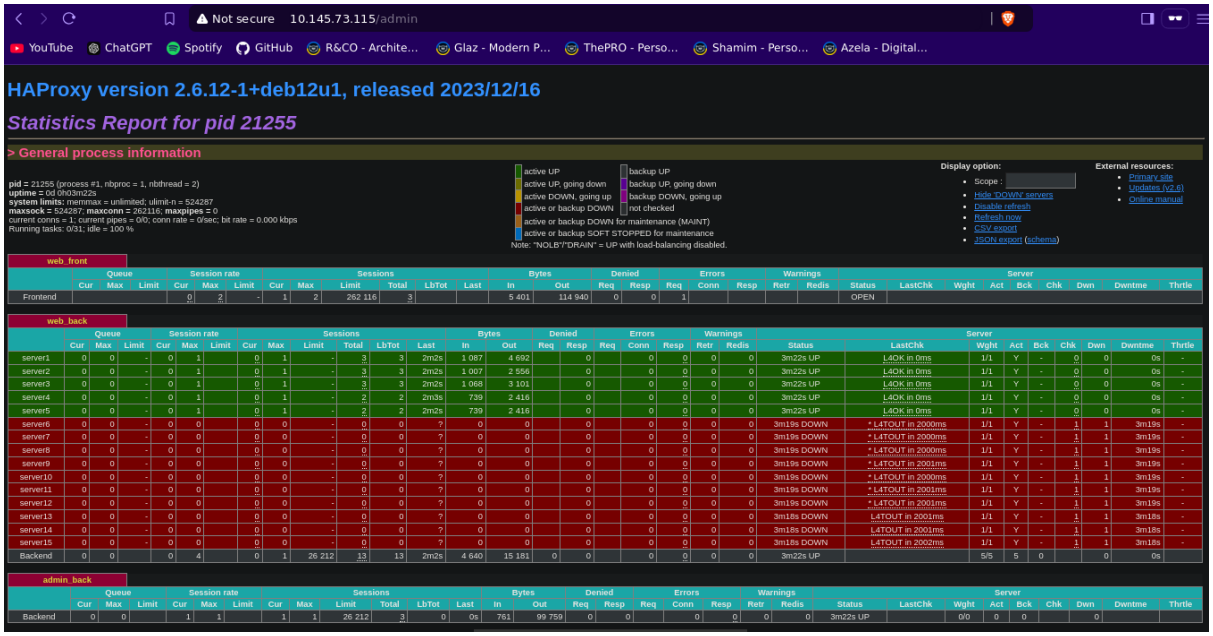
```
  GNU nano 7.2                                          /etc/haproxy/haproxy.cfg *
frontend web_front
        bind *:80
        acl is_admin path_beg /admin
        use_backend admin_back if is_admin
        default_backend web_back

backend web_back
        balance roundrobin
        server server1 10.145.73.115:8081 check
        server server2 10.145.73.115:8082 check
        server server3 10.145.73.115:8083 check
        server server4 10.145.73.115:8084 check
        server server5 10.145.73.115:8085 check
        server server6 10.145.73.116:8081 check
        server server7 10.145.73.116:8082 check
        server server8 10.145.73.116:8083 check
        server server9 10.145.73.116:8084 check
        server server10 10.145.73.116:8085 check
        server server11 10.145.73.117:8081 check
        server server12 10.145.73.117:8082 check
        server server13 10.145.73.117:8083 check
        server server14 10.145.73.117:8084 check
        server server15 10.145.73.117:8085 check

backend admin_back
        stats enable
        stats uri /admin
        stats realm Haproxy\ Statistics
        stats refresh 10s_
```

server1 hingga server5 adalah container1 hingga container5 pada vm server1.
server6 hingga server10 adalah container1 hingga container5 pada vm server2.
server11 hingga server15 adalah container1 hingga container5 pada vm server3.

```
alfath@server1:~/project$ sudo haproxy -c -f /etc/haproxy/haproxy.cfg
Configuration file is valid
alfath@server1:~/project$ sudo systemctl restart haproxy
```

c. testing





pada server6 hingga 15 berwarna merah menandakan server down atau tidak aktif, hal tersebut dikarenakan memang vm server2 dan server3 belum dibuat.

3. Failover

   Membuat failover menggunakan keepalived:

   a. install keepalived

```
alfath@server1:~/project$ sudo apt install keepalived_
```

   b. konfigurasi keepalived

```
alfath@server1:~$ sudo nano /etc/keepalived/keepalived.conf
```

```
 GNU nano 7.2                                  /etc/keepalived/keepalived.conf
vrrp_instance VI_1 {
    state MASTER
    interface enp1s0
    virtual_router_id 51
    priority 101
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        10.145.73.126
    }
}
```

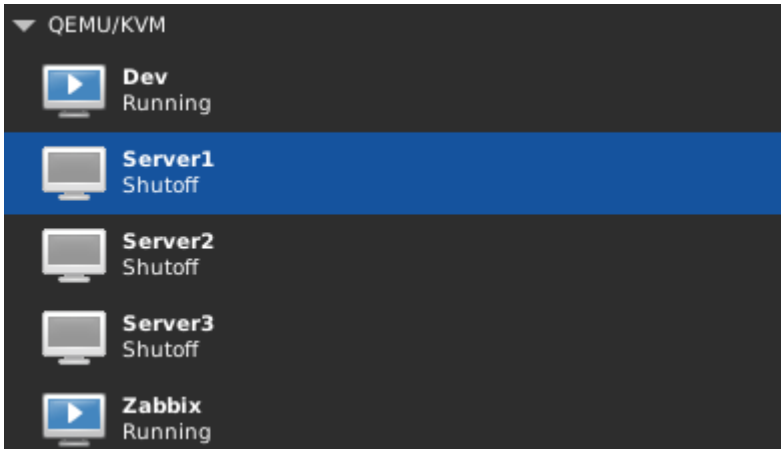
sesuaikan interface dengan interface host
pastikan virtual_ipaddress berada didalam subnet

```
alfath@server1:~/project$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 52:54:00:32:3d:de brd ff:ff:ff:ff:ff:ff
    inet 10.145.73.115/28 brd 10.145.73.127 scope global enp1s0
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fe32:3dde/64 scope link
       valid_lft forever preferred_lft forever
```

```
alfath@server1:~/project$ sudo systemctl restart keepalived
```

4. Clone Vm

klik kanan pada vm lalu pilih clone
pastikan vm dalam kondisi mati terlebih dahulu.



jika sudah berhasil diclone, jalankan semua vm.

5. Konfigurasi VM clone

```
alfath@server1:~$ sudo nano /etc/network/interfaces
```

sesuaikan dengan konfigurasi jaringan

```
alfath@server1:~$ sudo nano /etc/hosts
```

```
  GNU nano 7.2
127.0.0.1       localhost
10.145.73.116   server2.sister.lab      server2

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

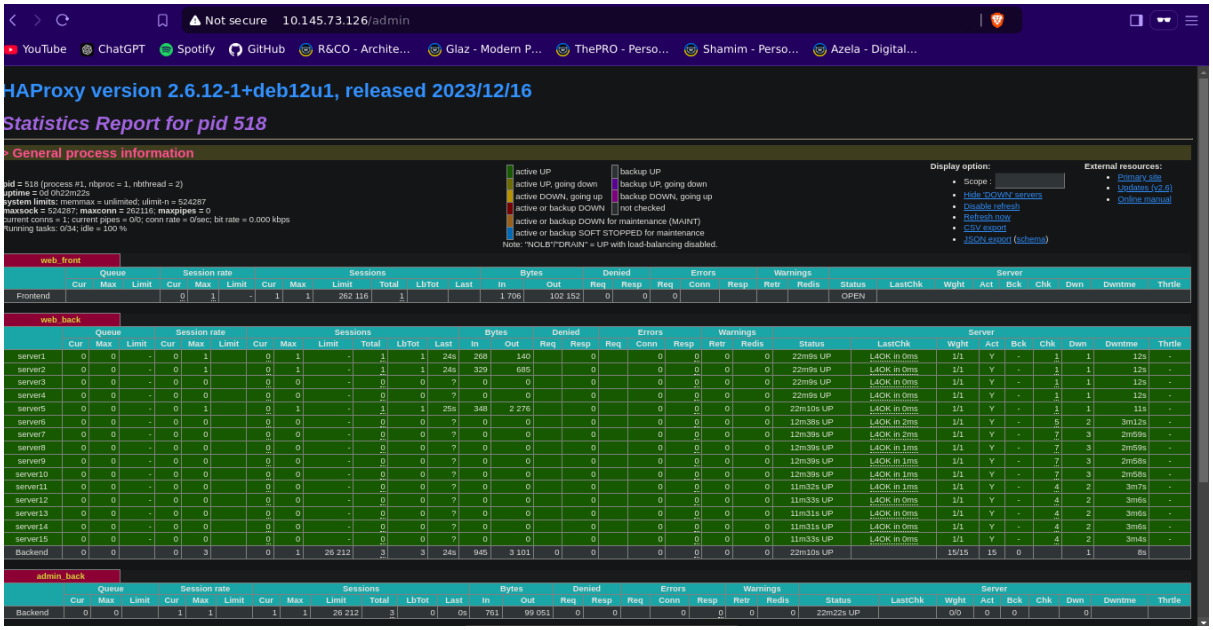
```
alfath@server1:~$ sudo nano /etc/hostname
```

```
alfath@server1:~$ sudo nano /etc/keepalived/keepalived.conf
```

```
  GNU nano 7.2
vrrp_instance VI_1 {
    state BACKUP
    interface enp1s0
    virtual_router_id 51
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass 1111
    }
    virtual_ipaddress {
        10.145.73.126
    }
}
```

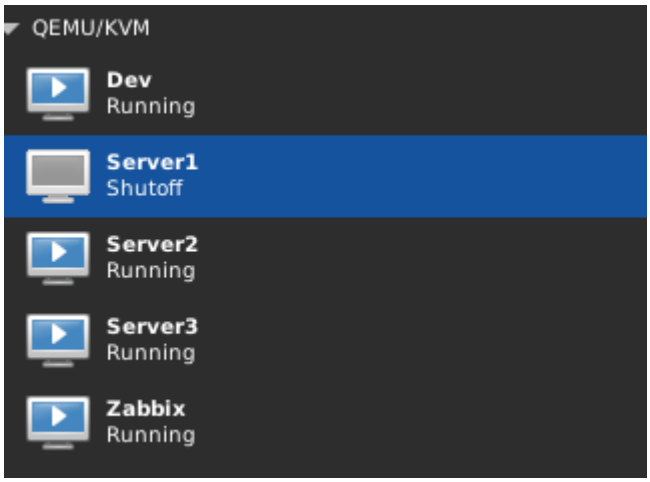
ubah pada state menjadi BACKUP dan priority dibawah priority dari MASTER.
restart vm.
lakukan hal yang sama pada Server3.

6. Testing load balancer



7. Testing failover

matikan vm server 1



akses virtual ip di browser