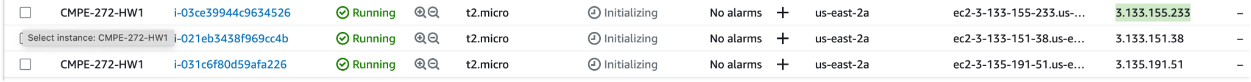
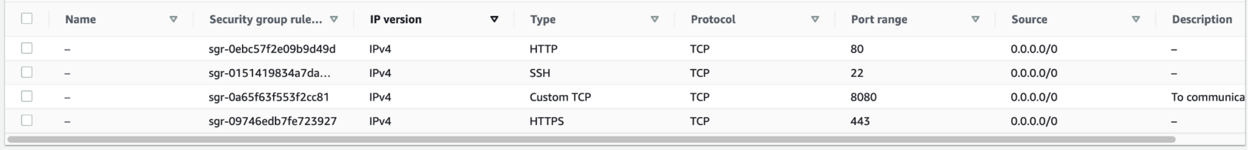
Link to the github page: https://github.com/Bryan-Az/CMPE-272-HW1

Step 1: Creating the Virtual Machines (Hosted on AWS - Couldn’t access the dCloud on Cisco)



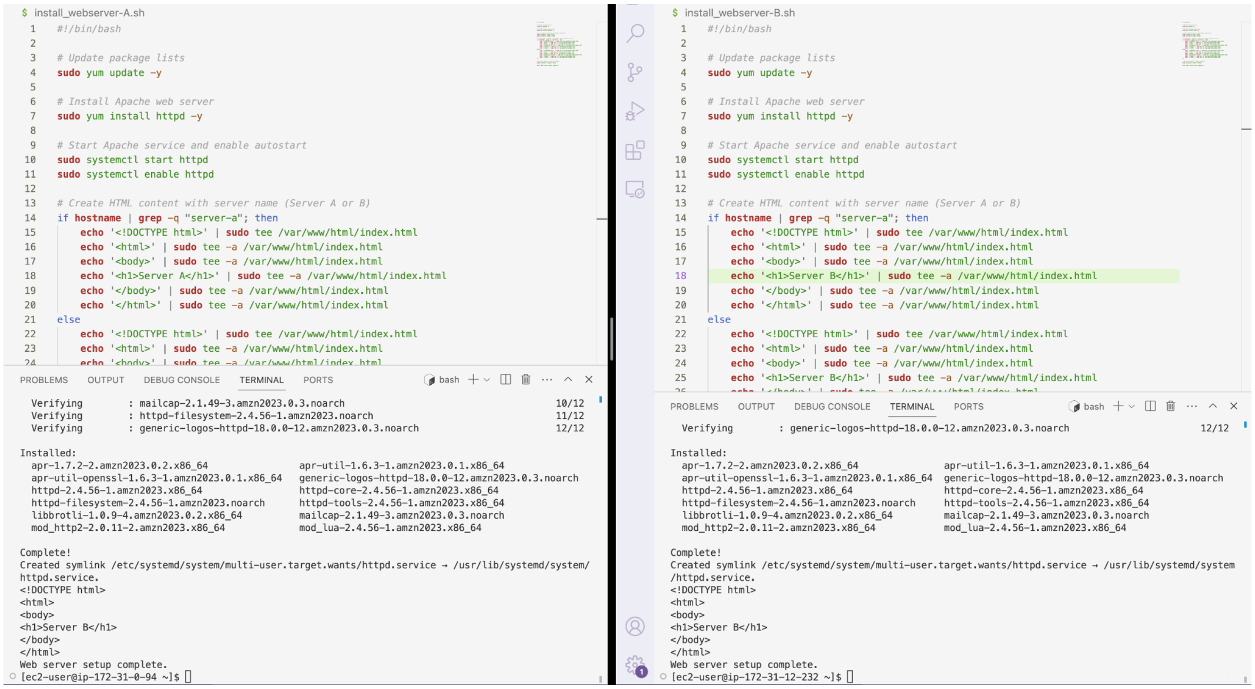
☐	CMPE-272-HW1	i-03ce39944c9634526	✓ Running	⊕⊖	t2.micro	Initializing	No alarms	+	us-east-2a	ec2-3-133-155-233.us-...	3.133.155.233	–
☐	Select instance: CMPE-272-HW1	i-021eb3438f969cc4b	✓ Running	⊕⊖	t2.micro	Initializing	No alarms	+	us-east-2a	ec2-3-133-151-38.us-e...	3.133.151.38	–
☐	CMPE-272-HW1	i-031c6f80d59afa226	✓ Running	⊕⊖	t2.micro	Initializing	No alarms	+	us-east-2a	ec2-3-135-191-51.us-e...	3.135.191.51	–

Step 2: Opening the port 8080 on all instances within the security group



☐	Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
☐	–	sgr-0ebc57f2e09b9d49d	IPv4	HTTP	TCP	80	0.0.0.0/0	–
☐	–	sgr-0151419834a7da...	IPv4	SSH	TCP	22	0.0.0.0/0	–
☐	–	sgr-0a65f63f553f2cc81	IPv4	Custom TCP	TCP	8080	0.0.0.0/0	To communica
☐	–	sgr-09746edb7fe723927	IPv4	HTTPS	TCP	443	0.0.0.0/0	–

Step 3: Starting the web server within the two load-balancing ec2 server instances



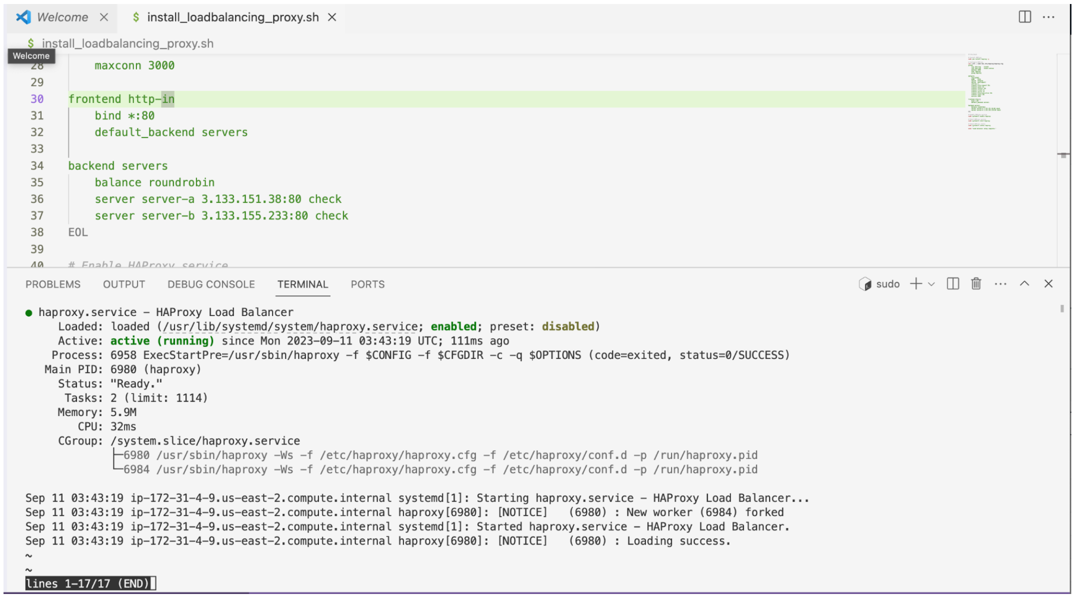
```bash
#!/bin/bash

# Update package lists
sudo yum update -y

# Install Apache web server
sudo yum install httpd -y

# Start Apache service and enable autostart
sudo systemctl start httpd
sudo systemctl enable httpd

# Create HTML content with server name (Server A or B)
if hostname | grep -q "server-a"; then
    echo '<!DOCTYPE html>' | sudo tee /var/www/html/index.html
    echo '<html>' | sudo tee -a /var/www/html/index.html
    echo '<body>' | sudo tee -a /var/www/html/index.html
    echo '<h1>Server A</h1>' | sudo tee -a /var/www/html/index.html
    echo '</body>' | sudo tee -a /var/www/html/index.html
    echo '</html>' | sudo tee -a /var/www/html/index.html
else
    echo '<!DOCTYPE html>' | sudo tee /var/www/html/index.html
    echo '<html>' | sudo tee -a /var/www/html/index.html
    echo '<body>' | sudo tee -a /var/www/html/index.html
```

```
  Verifying        : mailcap-2.1.49-3.amzn2023.0.3.noarch                              10/12
  Verifying        : httpd-filesystem-2.4.56-1.amzn2023.noarch                         11/12
  Verifying        : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                 12/12

Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64                 apr-util-1.6.3-1.amzn2023.0.1.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64    generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-2.4.56-1.amzn2023.x86_64                  httpd-core-2.4.56-1.amzn2023.x86_64
  httpd-filesystem-2.4.56-1.amzn2023.noarch       httpd-tools-2.4.56-1.amzn2023.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64           mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-2.0.11-2.amzn2023.x86_64              mod_lua-2.4.56-1.amzn2023.x86_64

Complete!
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/
httpd.service.
<!DOCTYPE html>
<html>
<body>
<h1>Server B</h1>
</body>
</html>
Web server setup complete.
○ [ec2-user@ip-172-31-0-94 ~]$ 
```

```bash
#!/bin/bash

# Update package lists
sudo yum update -y

# Install Apache web server
sudo yum install httpd -y

# Start Apache service and enable autostart
sudo systemctl start httpd
sudo systemctl enable httpd

# Create HTML content with server name (Server A or B)
if hostname | grep -q "server-a"; then
    echo '<!DOCTYPE html>' | sudo tee /var/www/html/index.html
    echo '<html>' | sudo tee -a /var/www/html/index.html
    echo '<body>' | sudo tee -a /var/www/html/index.html
    echo '<h1>Server B</h1>' | sudo tee -a /var/www/html/index.html
    echo '</body>' | sudo tee -a /var/www/html/index.html
    echo '</html>' | sudo tee -a /var/www/html/index.html
else
    echo '<!DOCTYPE html>' | sudo tee /var/www/html/index.html
    echo '<html>' | sudo tee -a /var/www/html/index.html
    echo '<body>' | sudo tee -a /var/www/html/index.html
    echo '<h1>Server B</h1>' | sudo tee -a /var/www/html/index.html
    echo '</body>' | sudo tee -a /var/www/html/index.html
```

```
  Verifying        : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch                 12/12

Installed:
  apr-1.7.2-2.amzn2023.0.2.x86_64                 apr-util-1.6.3-1.amzn2023.0.1.x86_64
  apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64    generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  httpd-2.4.56-1.amzn2023.x86_64                  httpd-core-2.4.56-1.amzn2023.x86_64
  httpd-filesystem-2.4.56-1.amzn2023.noarch       httpd-tools-2.4.56-1.amzn2023.x86_64
  libbrotli-1.0.9-4.amzn2023.0.2.x86_64           mailcap-2.1.49-3.amzn2023.0.3.noarch
  mod_http2-2.0.11-2.amzn2023.x86_64              mod_lua-2.4.56-1.amzn2023.x86_64

Complete!
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system
/httpd.service.
<!DOCTYPE html>
<html>
<body>
<h1>Server B</h1>
</body>
</html>
Web server setup complete.
○ [ec2-user@ip-172-31-12-232 ~]$ 
```

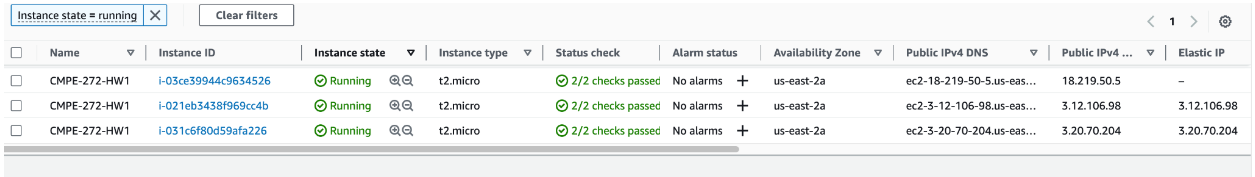
Step 4: Setting up the load balancing HAProxy service on the main server


Welcome
install_loadbalancing_proxy.sh
$ install_loadbalancing_proxy.sh
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
sudo

```
    maxconn 3000

frontend http-in
    bind *:80
    default_backend servers

backend servers
    balance roundrobin
    server server-a 3.133.151.38:80 check
    server server-b 3.133.155.233:80 check
EOL

# Enable HAProxy service
```

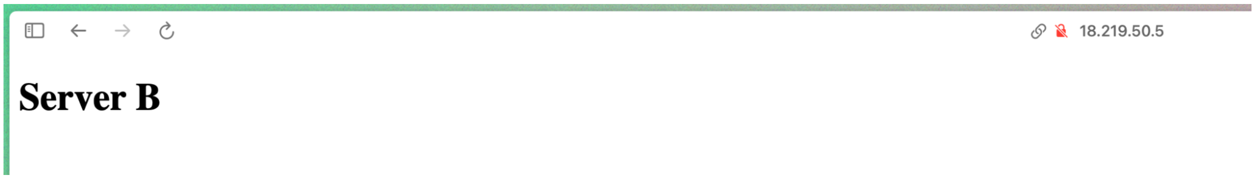
```
● haproxy.service - HAProxy Load Balancer
     Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; preset: disabled)
     Active: active (running) since Mon 2023-09-11 03:43:19 UTC; 111ms ago
    Process: 6958 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -f $CFGDIR -c -q $OPTIONS (code=exited, status=0/SUCCESS)
   Main PID: 6980 (haproxy)
     Status: "Ready."
      Tasks: 2 (limit: 1114)
     Memory: 5.9M
        CPU: 32ms
     CGroup: /system.slice/haproxy.service
             ├─6980 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -f /etc/haproxy/conf.d -p /run/haproxy.pid
             └─6984 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -f /etc/haproxy/conf.d -p /run/haproxy.pid

Sep 11 03:43:19 ip-172-31-4-9.us-east-2.compute.internal systemd[1]: Starting haproxy.service - HAProxy Load Balancer...
Sep 11 03:43:19 ip-172-31-4-9.us-east-2.compute.internal haproxy[6980]: [NOTICE]   (6980) : New worker (6984) forked
Sep 11 03:43:19 ip-172-31-4-9.us-east-2.compute.internal systemd[1]: Started haproxy.service - HAProxy Load Balancer.
Sep 11 03:43:19 ip-172-31-4-9.us-east-2.compute.internal haproxy[6980]: [NOTICE]   (6980) : Loading success.
~
~
lines 1-17/17 (END)
```

Step 5: Configuring Elastic IP addresses (EIPs) for the load-balancing servers to point to the main server (HAProxy) - ignore the ips in the cfg above - those were replaced with the elastic ips.

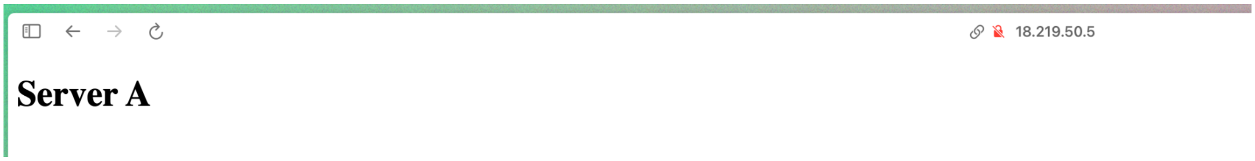


□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
□	CMPE-272-HW1	i-03ce39944c9634526	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-2a	ec2-18-219-50-5.us-eas...	18.219.50.5	–
□	CMPE-272-HW1	i-021eb3438f969cc4b	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-2a	ec2-3-12-106-98.us-eas...	3.12.106.98	3.12.106.98
□	CMPE-272-HW1	i-031c6f80d59afa226	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-2a	ec2-3-20-70-204.us-eas...	3.20.70.204	3.20.70.204

Step 6: Checking the output of requesting the main server's IP.



Server B



Server A