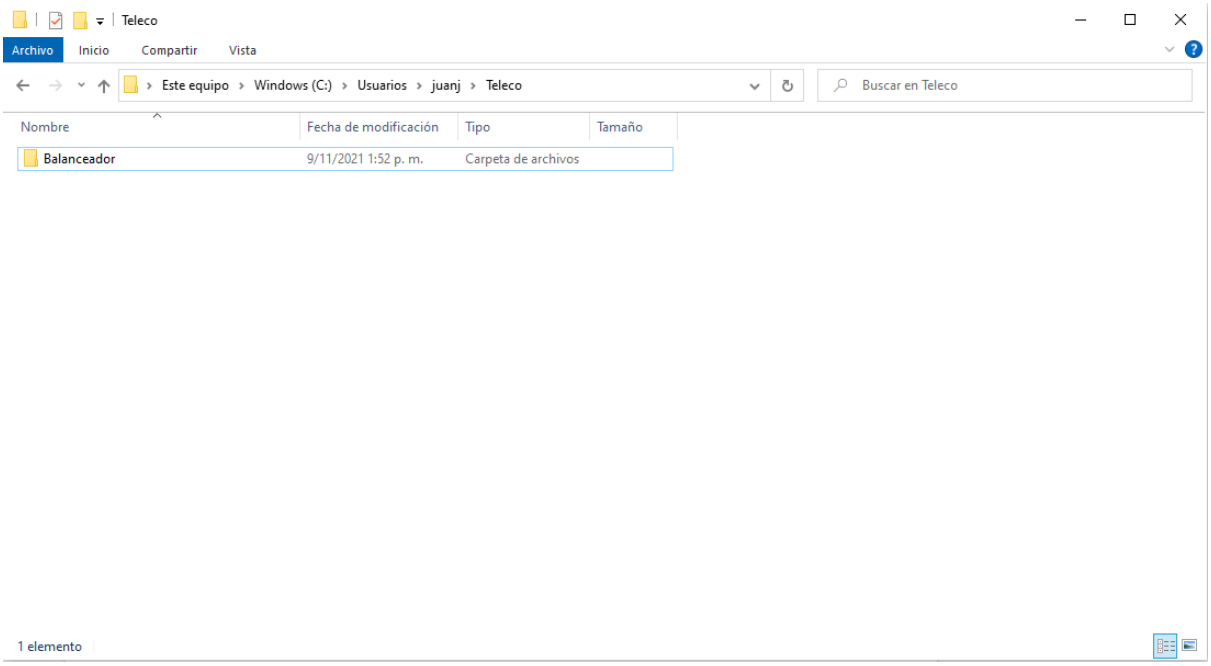
Guia Desarrollo balanceador de carga con Haproxy.

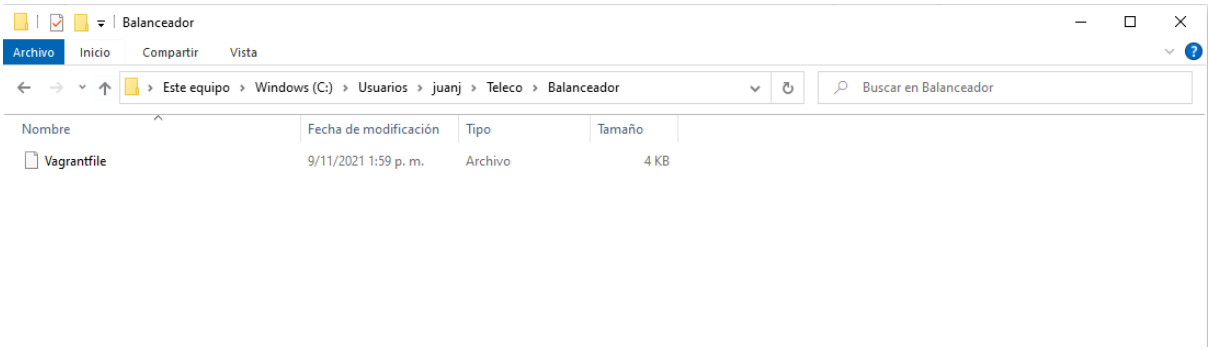
Para empezar creamos una carpeta en la cual alojaremos las máquinas virtuales.



Una vez creada la carpeta, accedemos a ella desde el CMD de windows o desde windows powershell. y ejecutamos el comando “vagrant init” y presionamos la tecla “enter”

```
PS C:\Users\juanj\Documents\Balanceador> vagrant init
```

Una vez hecho esto, verificamos que en la carpeta creada aparece un archivo de nombre “Vagrantfile”



Modificamos el texto presente en el archivo “Vagrantfile” por un código como el siguiente.

```
Vagrant.configure("2") do |config|

    config.vm.define :balanceador do |balanceador|
        balanceador.vm.box = "bento/centos-7.9"
        balanceador.vm.network :private_network, ip: "192.168.50.10"
        balanceador.vm.hostname = "balanceador"
        balanceador.vm.provision "shell", path: "script-pf2/balanceador.sh"
    end
        config.vm.define :web1 do |web1|
        web1.vm.box = "bento/centos-7.9"
        web1.vm.network :private_network, ip: "192.168.50.11"
        web1.vm.hostname = "web1"
        web1.vm.provision "shell", path: "script-pf2/webserver11.sh"
    end
        config.vm.define :web2 do |web2|
        web2.vm.box = "bento/centos-7.9"
        web2.vm.network :private_network, ip: "192.168.50.12"
        web2.vm.hostname = "web2"
        web2.vm.provision "shell", path: "script-pf2/webserver12.sh"
    end
end
```

En el código se configura las máquinas virtuales que se utilizaran. Es importante ubicar la carpeta “script-pf2”, la cual contiene los archivos de aprovisionamiento, en la misma carpeta donde se encuentra el vagrantfile.
Una vez modificado el contenido del Vagrantfile, guardamos los cambios y nos dirigimos al cmd o al windows powershell y una vez ahí ejecutamos el comando “vagrant up” para crear y encender las máquinas virtuales.

```
PS C:\Users\juanj\Documents\Balanceador> vagrant up
```

Una vez terminado el proceso de creación de las máquinas virtuales verificamos que las máquinas estén encendidas, para ello utilizamos el comando “vagrant status” y se debería ver algo como lo siguiente.

```
PS C:\Users\juanj\Documents\Balanceador> vagrant status
Current machine states:

balanceador               running (virtualbox)
web1                      running (virtualbox)
web2                      running (virtualbox)

This environment represents multiple VMs. The VMs are all listed
above with their current state. For more information about a specific
VM, run `vagrant status NAME`.
PS C:\Users\juanj\Documents\Balanceador>
```

En caso de que las máquinas no estén encendidas volver a utilizar el comando "vagrant up".
Una vez encendidas las máquinas virtuales, procedemos a configurar los servidores web, para ello entramos a la máquina web1 con el comando
"vagrant ssh web1".

```
PS C:\Users\juanj\Documents\Balanceador> vagrant ssh web1
```

Una vez dentro, verificamos que el servicio httpd esté instalado. para ello utilizamos el comando "service httpd status"

```
[vagrant@web1 ~]$ service httpd status
```

Debería aparecer de la siguiente manera.

```
[vagrant@web1 ~]$ service httpd status
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2021-11-09 20:34:52 UTC; 29min ago
     Docs: man:httpd(8)
           man:apachectl(8)
 Main PID: 840 (httpd)
   Status: "Total requests: 870; Current requests/sec: 0.5; Current traffic: 204 B/sec"
   CGroup: /system.slice/httpd.service
           ├─ 840 /usr/sbin/httpd -DFOREGROUND
           ├─ 873 /usr/sbin/httpd -DFOREGROUND
           ├─ 874 /usr/sbin/httpd -DFOREGROUND
           ├─ 875 /usr/sbin/httpd -DFOREGROUND
           ├─ 876 /usr/sbin/httpd -DFOREGROUND
           ├─ 878 /usr/sbin/httpd -DFOREGROUND
           └─3058 /usr/sbin/httpd -DFOREGROUND
[vagrant@web1 ~]$
```

Una vez verificamos que el servicio está instalado y encendido, nos logueamos como root con el comando “sudo -i”

```
[vagrant@web1 ~]$ sudo -i
```

Ya como root nos dirigimos al archivo de configuración “httpd.conf” usando el siguiente comando.

```
[root@web1 ~]# vim /etc/httpd/conf/httpd.conf
```

Una vez en el archivo, presionamos la tecla “A” para habilitar su edición y lo modificamos de tal manera que quede como el que está en el repositorio.
Una vez modificado el archivo de configuración presionamos la tecla “ESC” y después “:wq” para salir guardando cambios.
Una vez afuera reiniciamos el servicio con el comando “service httpd restart”

```
[root@web1 ~]# service httpd restart
```

Lo siguiente es crear el archivo index.html, para ello usamos el siguiente comando.

```
[root@web1 html]# vim /var/www/html/index.html
```

al presionar enter se abrirá el archivo en blanco; presionamos la tecla “A” para habilitar la edición y se modifica el archivo para que quede como el del repositorio.
Una vez hecho esto presionamos la tecla “ESC” y posteriormente “:wq” para salir guardando cambios.
Lo siguiente es darle permisos al archivo index.html, para ello usamos el siguiente comando.

```
[root@web1 html]# chmod 755 /var/www/html/index.html
```

Después de lo anterior solo queda probar, para ello abrimos el navegador y en el buscador escribimos la ip del servidor web 1 y deberá aparecer la página descrita en el archivo index.html.

Para configurar el servidor web 2 seguir los mismos pasos que para el servidor web 1, usando el archivo index.html para el servidor web 2.

Una vez configurados los servidores http, vamos a configurar el balanceador, para ello en la máquina balanceador nos logueamos como root usando el comando “sudo -i”

```
[vagrant@balanceador ~]$ sudo -i
```

Ya como root vamos a verificar que el servicio haproxy esté instalado y corriendo; para ello usamos el comando “service haproxy status”.

```
[root@balanceador ~]# service haproxy status
```

Debería aparecer el servicio corriendo de la siguiente manera.

```
[root@balanceador ~]# service haproxy status
Redirecting to /bin/systemctl status haproxy.service
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2021-11-09 20:34:24 UTC; 2h 48min ago
 Main PID: 837 (haproxy-systemd)
   CGroup: /system.slice/haproxy.service
           ├─837 /usr/sbin/haproxy-systemd-wrapper -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid
           ├─842 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds
           └─848 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds

Nov 09 20:34:24 balanceador systemd[1]: Started HAProxy Load Balancer.
Nov 09 20:34:24 balanceador haproxy-systemd-wrapper[837]: haproxy-systemd-wrapper: executing /usr/sbin/...Ds
Hint: Some lines were ellipsized, use -l to show in full.
[root@balanceador ~]#
```

Una vez verificamos que el servicio está instalado y corriendo, procedemos a modificar el archivo de configuración “haproxy.cfg” usando el siguiente comando.

```
[root@balanceador ~]# vim /etc/haproxy/haproxy.cfg
```

Una vez en el archivo presionamos la tecla “A” para habilitar la edición y procedemos a modificar el archivo como el que está en el repositorio.

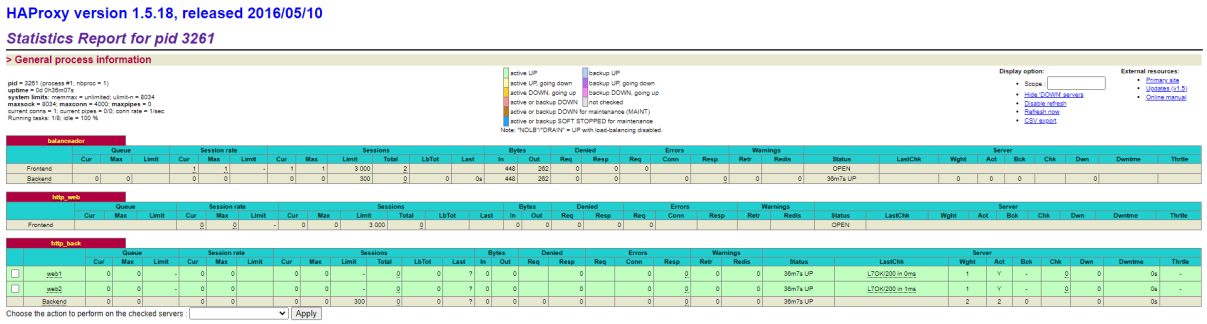
Una vez el archivo está modificado presionamos la tecla “ESC” y después “:wq” para salir guardando cambios.

Una vez afuera procedemos a reiniciar el servicio con el comando

“service haproxy restart”.

```
[root@balanceador ~]# service haproxy restart
```

Para probar que el balanceador está funcionando correctamente, en el navegador ponemos el siguiente enlace. http://192.168.50.10:8080/estadisticas . Posteriormente ingresamos con el usuario y la contraseña especificados en el archivo de configuración y cargará las estadísticas de haproxy.



HAProxy version 1.5.18, released 2016/05/10

Statistics Report for pid 3261

> General process information

pid = 3261 (process #1, nbproc = 1)
uptime = 0d 0h38m07s
system limits: memmax = unlimited; ulimit-n = 8034
maxsock = 8034; maxconn = 4000; maxpipes = 0
current conns = 1; current pipes = 0/0; conn rate = 1/sec
Running tasks: 1/8; idle = 100 %

active UP
active UP, going down
active DOWN, going up
active or backup DOWN
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
backup UP
backup UP, going down
backup DOWN, going up
not checked
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide DOWN servers
- Disable refresh
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v1.5)
- Online manual

balanceador

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				1	1	-	1	1	3 000	2			448	282	0	0	0					OPEN								
Backend	0	0		0	0		0	0	300	0	0	0s	448	282	0	0		0	0	0	0	36m7s UP		0	0	0		0		

http_web

	Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings		Server								
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
Frontend				0	0	-	0	0	3 000	0			0	0	0	0	0					OPEN								

http_back

		Queue			Session rate			Sessions						Bytes		Denied		Errors			Warnings			Server								
		Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	Last	In	Out	Req	Resp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Dwntme	Thrtle
	web1	0	0	-	0	0		0	0	-	0	0	?	0	0			0		0	0	0	36m7s UP	L7OK/200 in 0ms	1	Y	-	0	0	0s	-
	web2	0	0	-	0	0		0	0	-	0	0	?	0	0			0		0	0	0	36m7s UP	L7OK/200 in 1ms	1	Y	-	0	0	0s	-
	Backend	0	0		0	0		0	0	300	0	0	?	0	0	0		0		0	0	0	36m7s UP		2	2	0		0	0s	

Choose the action to perform on the checked servers : Apply