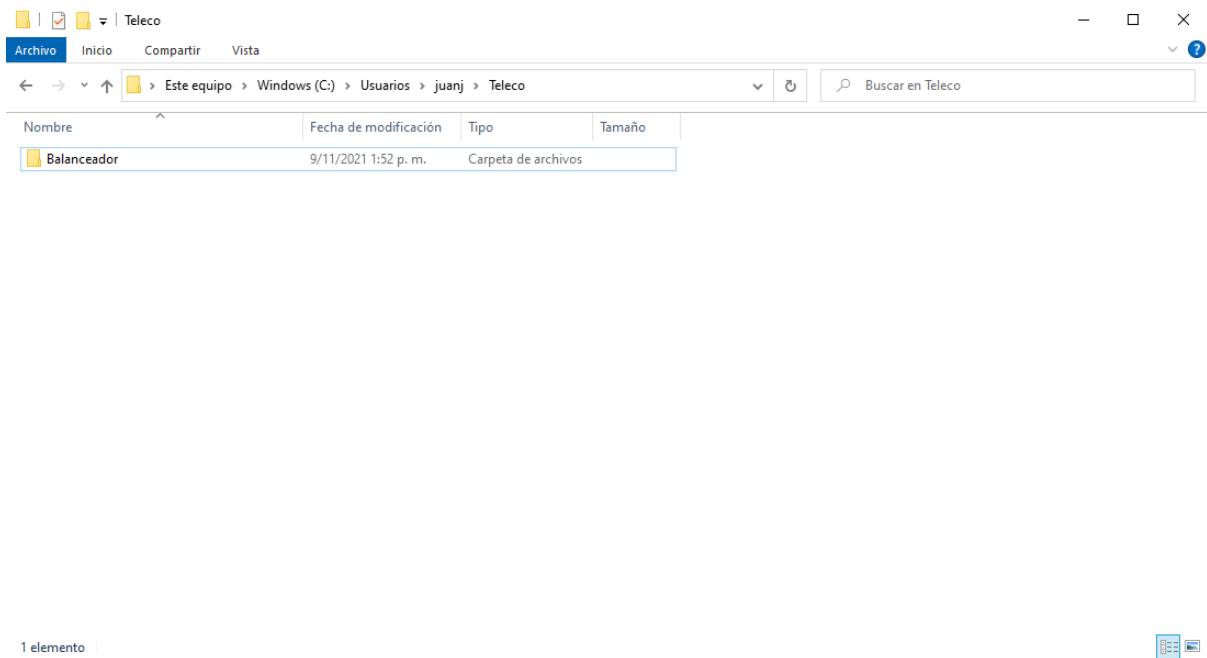


Guia Desarrollo balanceador de carga con Mod_proxy_balancer.

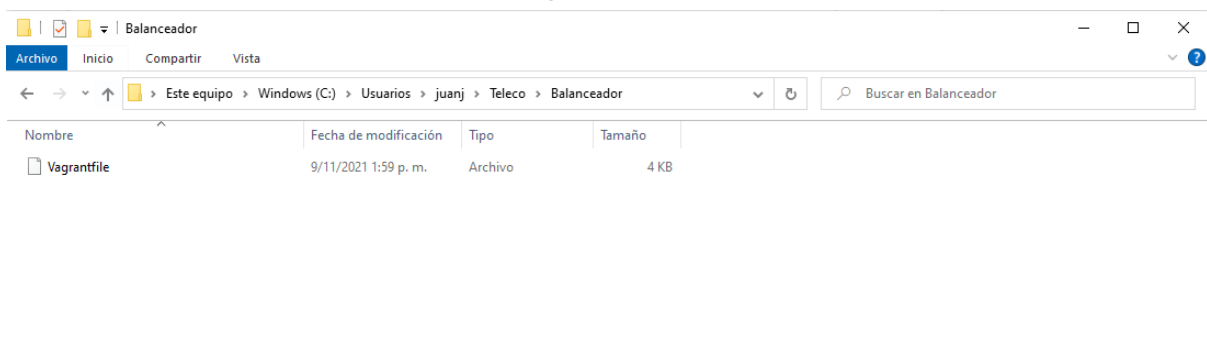
- Para empezar, se crea una carpeta en la cual se alojarán las máquinas virtuales.



- Una vez creada la carpeta, se accede a ella desde el CMD de windows o desde windows powershell. Se ejecuta el comando `vagrant init`

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

- Una vez hecho esto, se verifica que en la carpeta creada aparezca un archivo con el nombre de “Vagrantfile”



- Modificar el texto presente en el archivo “Vagrantfile”, como se muestra en la siguiente imagen.

```
1 $script = <<-'SCRIPT'
2 sudo -i
3 yum install -y httpd
4 SCRIPT
5
6 Vagrant.configure("2") do |config|
7
8     config.vm.define :balanceador do |balanceador|
9         balanceador.vm.box = "bento/centos-7.9"
10        balanceador.vm.network :private_network, ip: "192.168.50.20"
11        balanceador.vm.hostname = "balanceador"
12        balanceador.vm.provision "shell", inline: $script
13    end
14
15    config.vm.define :web1 do |web1|
16        web1.vm.box = "bento/centos-7.9"
17        web1.vm.network :private_network, ip: "192.168.50.21"
18        web1.vm.hostname = "web1"
19        web1.vm.provision "shell", inline: $script
20    end
21
22    config.vm.define :web2 do |web2|
23        web2.vm.box = "bento/centos-7.9"
24        web2.vm.network :private_network, ip: "192.168.50.22"
25        web2.vm.hostname = "web2"
26        web2.vm.provision "shell", inline: $script
27    end
28 end
```

- En el código se configura las máquinas virtuales que se utilizarán.
- Una vez modificado el contenido del Vagrantfile, guardar los cambios y dirigirse al cmd o powershell. Una vez ahí, ejecuta 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 ~]$ |
```

- Sí el servicio no está activo lo encendemos de la siguiente manera. En primer lugar nos logueamos como root con el siguiente comando

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

- Después usamos el comando `service httpd start`.
- Una vez verificado que el servicio esté instalado y encendido, 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|
```

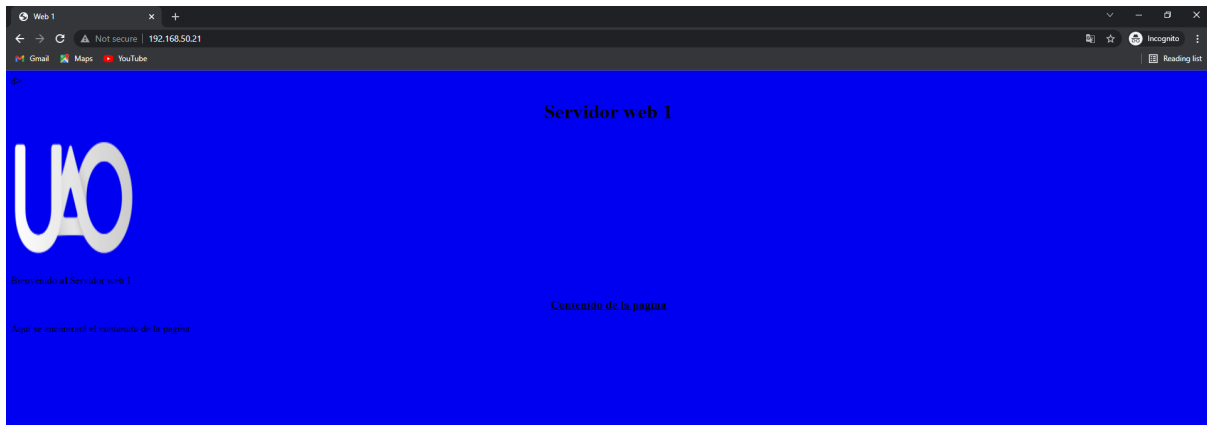
- Lo siguiente es crear el archivo `index.html`, para ello usamos el siguiente comando.

```
[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 se procede a probar, para ello abrimos el navegador y en el buscador escribimos la dirección 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 httpd esté instalado y corriendo; para ello usamos el comando `service httpd status`.

```
[root@balanceador ~]# service httpd status|
```

- Debería aparecer el servicio corriendo de la siguiente manera.

```
[root@balanceador ~]# 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 20:11:41 UTC; 6h ago
     Docs: man:httpd(8)
           man:apachectl(8)
  Main PID: 849 (httpd)
   Status: "Total requests: 0; Current requests/sec: 0; Current traffic:  0 B/sec"
    CGroup: /system.slice/httpd.service
            └─849 /usr/sbin/httpd -DFOREGROUND
              └─878 /usr/sbin/httpd -DFOREGROUND
                └─879 /usr/sbin/httpd -DFOREGROUND
                  └─880 /usr/sbin/httpd -DFOREGROUND
                    └─881 /usr/sbin/httpd -DFOREGROUND
                      └─882 /usr/sbin/httpd -DFOREGROUND

Nov 09 20:11:41 balanceador systemd[1]: Starting The Apache HTTP Server...
Nov 09 20:11:41 balanceador httpd[849]: AH00558: httpd: Could not reliably determine the server's ful...sage
Nov 09 20:11:41 balanceador systemd[1]: Started The Apache HTTP Server.
Hint: Some lines were ellipsized, use -l to show in full.
[root@balanceador ~]# |
```

- Sí el servicio no está activo lo encendemos con el comando `service httpd start`

- Por defecto los módulos necesarios vienen con la instalación del servicio http, para verificarlos usamos el siguiente comando.

```
[root@balanceador ~]# httpd -M | grep proxy|
```

- Aparecen los módulos de proxy necesarios para configurar la máquina como balanceador

```
[root@balanceador ~]# httpd -M | grep proxy
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set
the 'ServerName' directive globally to suppress this message
proxy_module (shared)
proxy_ajp_module (shared)
proxy_balancer_module (shared)
proxy_connect_module (shared)
proxy_express_module (shared)
proxy_fcgi_module (shared)
proxy_fdpass_module (shared)
proxy_ftp_module (shared)
proxy_http_module (shared)
proxy_scgi_module (shared)
proxy_wstunnel_module (shared)
[root@balanceador ~]# |
```

Los cuales ya vienen habilitados por defecto, los módulos que se necesitan para configurar la máquina como balanceador son:

- proxy_module
- proxy_balancer_module
- proxy_connect_module
- proxy_http_module
- Ahora creamos el archivo de configuración vhost_default_00.conf en la dirección /etc/httpd/conf.d con el siguiente comando.

```
[root@balanceador ~]# vi /etc/httpd/conf.d/vhost_default_00.conf|
```

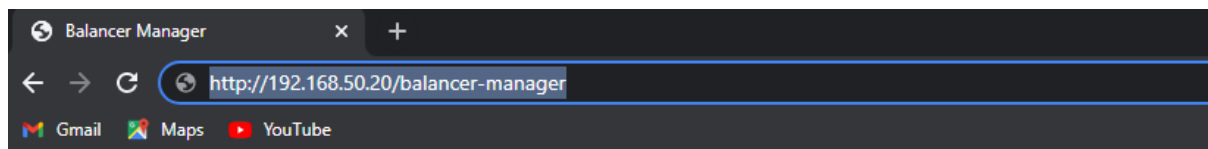
- Una vez abierto el archivo, presionamos la tecla “A” para activar la edición y lo modificamos como está en el repositorio. Una vez modificado el archivo presionamos la tecla “ESC” y posteriormente “:wq” para salir guardando cambios.

```

1  <VirtualHost *:80>
2      ProxyRequests off
3
4      <Location /balancer-manager>
5          SetHandler balancer-manager
6          Require all granted
7      </Location>
8
9      <Proxy balancer://cluster1>
10         BalancerMember http://192.168.50.21:80
11         BalancerMember http://192.168.50.22:80
12         ProxySet lbmethod=byrequests
13         ProxySet lbmethod=bytraffic
14     </Proxy>
15
16     ProxyPreserveHost On
17     ProxyPass /balancer-manager !
18
19     ProxyPass / balancer://cluster1
20     ProxyPassReverse / balancer://cluster1
21
22 </VirtualHost>

```

- Para probar que el balanceador está funcionando correctamente, en el navegador ponemos el siguiente enlace. <http://192.168.50.20/balancer-manager>. y cargará la página donde se monitorea el estado del balanceador.



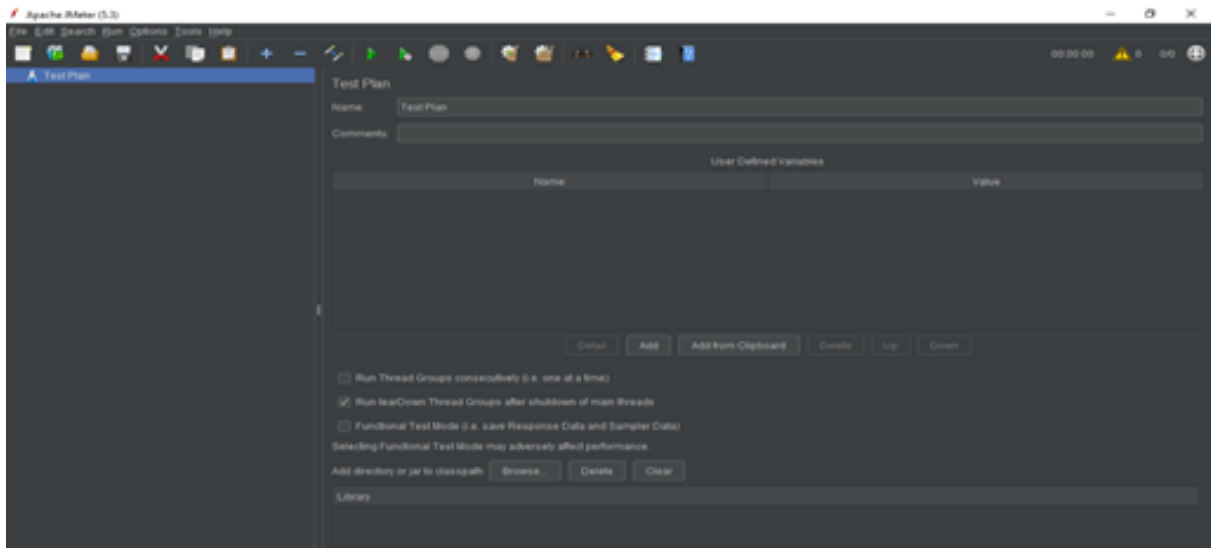
Load Balancer Manager for 192.168.50.20

Server Version: Apache/2.4.6 (CentOS)
Server Built: Oct 19 2021 13:53:40

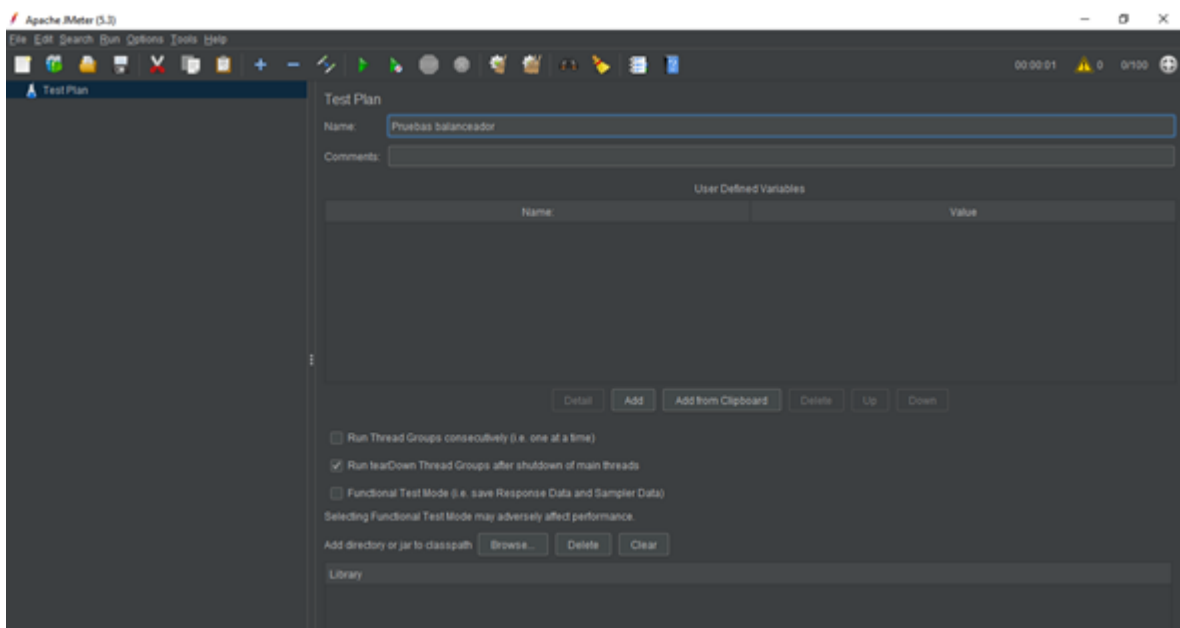
LoadBalancer Status for [balancer://cluster1](#) [p5bfbdf7e_cluster1]

MaxMembers	StickySession	DisableFailover	Timeout	FailoverAttempts	Method	Path	Active
2 [2 Used]	(None)	Off	0	1	byrequests	/	Yes

Worker URL	Route	RouteRedir	Factor	Set	Status	Elected	Busy	Load	To	From
http://192.168.50.21			1	0	Init Ok	0	0	0	0	0
http://192.168.50.22			1	0	Init Ok	0	0	0	0	0

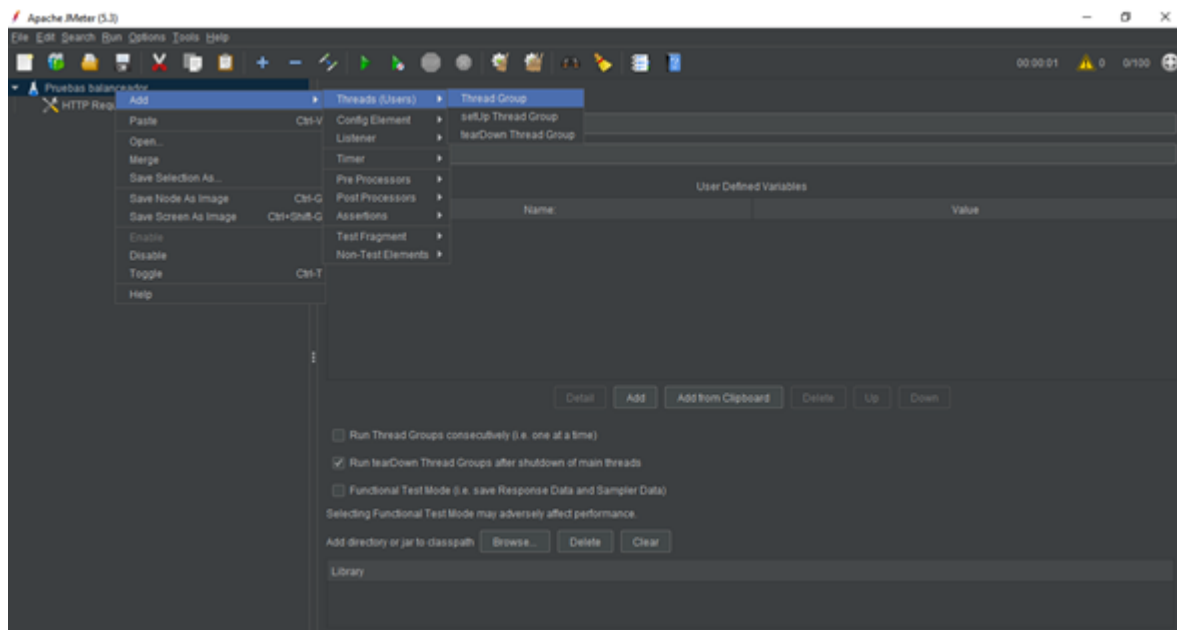


- Se procede a configurar Jmeter para realizar las pruebas.
- Inicialmente se renombra el Test plan de la siguiente manera:
- Haciendo click sobre **Test Plan** se despliegan las opciones a la derecha y en **Name**, asignar el nombre del proyecto por el de Pruebas Balanceador



- A continuación, damos click derecho sobre el proyecto y después de desplegar las opciones se escoge **add, threads (Users), Thread Group**.
- De esta manera se crea un grupo de usuarios que van a generar las peticiones hacia nuestro balanceador.

Nota: No es necesario guardar los cambios ya que se actualiza el nombre en tiempo real.



- De esta manera se muestran las opciones para configurar el grupo de usuarios ya creado.

Thread Group

Name:

Comments:

Action to be taken after a Sampler error

☒ Continue ☐ Start Next Thread Loop ☐ Stop Thread ☐ Stop Test ☐ Stop Test Now

Thread Properties

Number of Threads (users):

Ramp-up period (seconds):

Loop Count: ☐ Infinite

☐ Same user on each iteration

☐ Delay Thread creation until needed

☐ Specify Thread lifetime

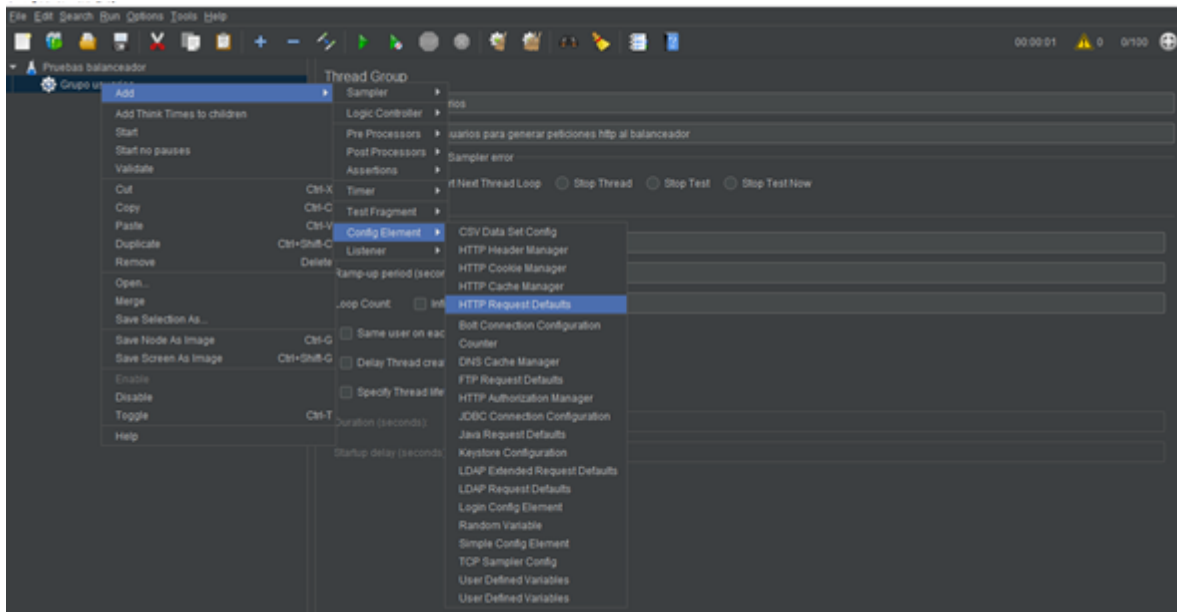
Duration (seconds):

Startup delay (seconds):

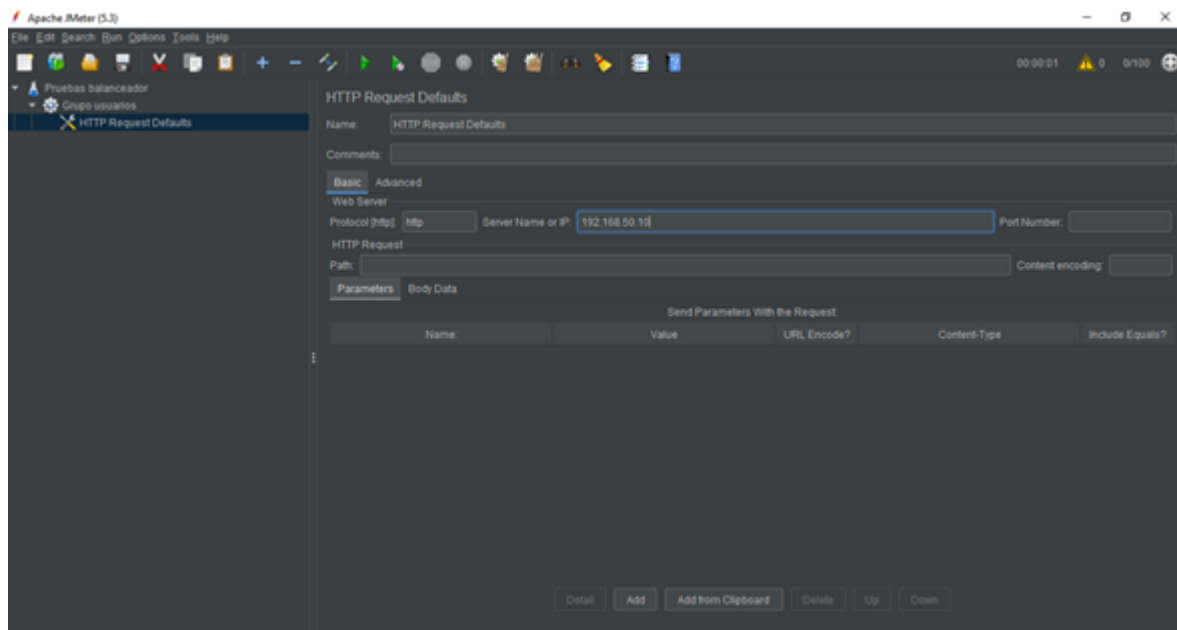
- **Action to taken after a Sample error – continue:** Permite continuar generando las peticiones, sí se llega a presentar un error con el fin de no interrumpir el proceso.
- **Thread Properties:** Permite personalizar las propiedades de las peticiones que se van a generar.
- En este caso se generarán peticiones de 2.000 usuarios cada segundo en un único bucle, de esta manera se producen 2.000 peticiones hacia el balanceador.
- Es importante que la opción **same user each iteration** y **Specify Thread lifetime** estén deshabilitadas, para que las peticiones recibidas por el balanceador sean generadas por diferentes usuarios, de esta manera no

ocurra que el **keep alive** al ser el mismo usuario se pueda incurrir que se envíen solicitudes al mismo servidor sin alternar.

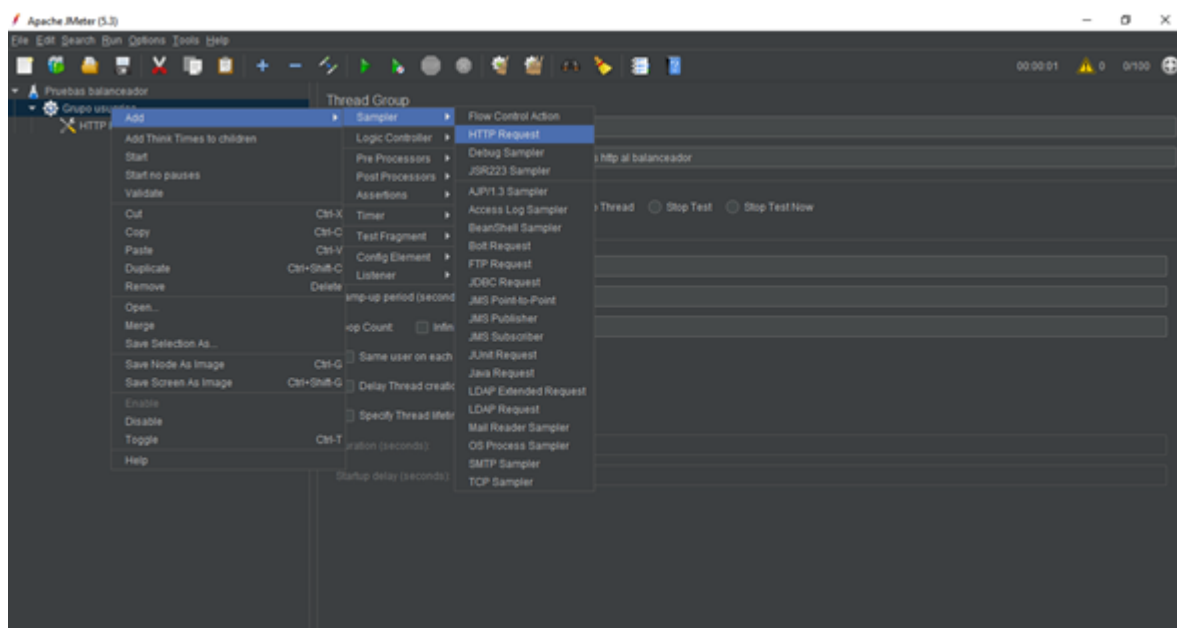
- A continuación, al dar click derecho sobre el grupo de usuarios y escoger la opción **Add, Config Element, HTTP Request Defaults**, se inicia con las configuraciones de los elementos necesarios para poder llevar a cabo las pruebas en el balanceador como se muestra a continuación.



- Crear un nuevo elemento por defecto para peticiones HTTP al balanceador, en el que únicamente se va a configurar el tipo de protocolo y la dirección IP o host del balanceador, con el fin de que las peticiones generadas sean dirigidas al balanceador y este las pueda atender de acuerdo con la configuración realizada previamente.
- Debido a que se trabajará sobre el puerto 80, ingresar a http y la dirección IP del balanceador 192.168.50.20, de la siguiente manera:

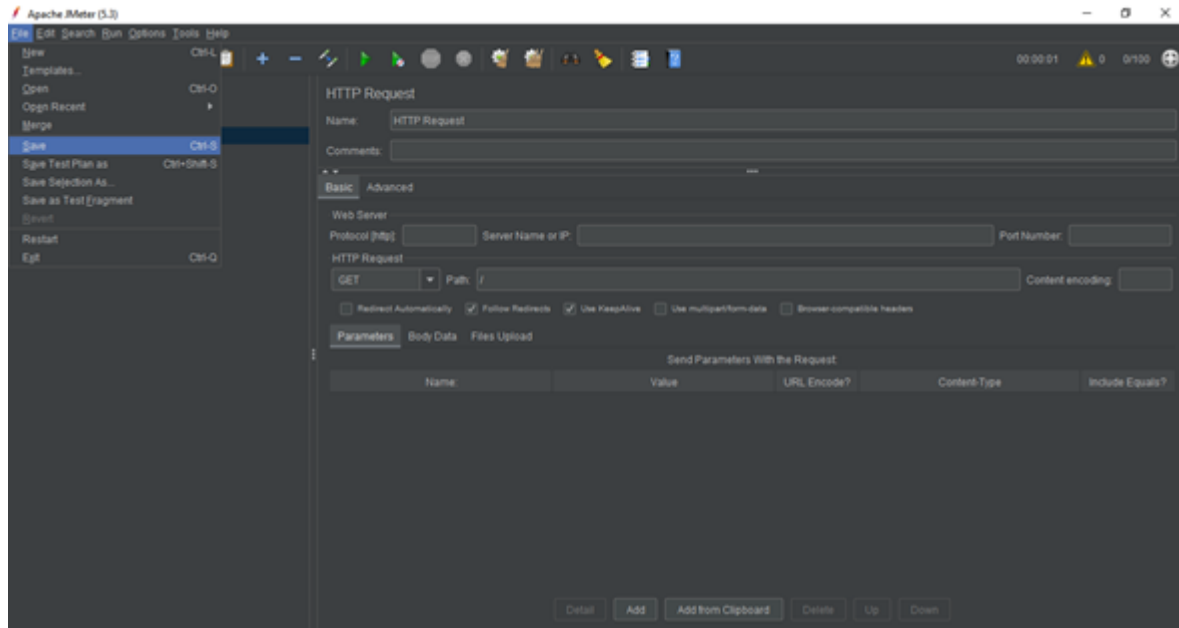


- Continuar modificando la siguiente parte de la configuración de la siguiente manera:
- Click derecho sobre el grupo de usuarios y escoger la opción, **Add, Sampler, HTTP request**

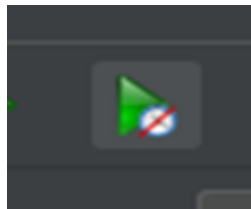


- De esta manera se agrega una nueva pestaña bajo el grupo de usuarios que va a permitir configurar el método que se va a usar para realizar las peticiones, y también permitirá mostrar que parte de la página van a ser dirigidas.
- En este caso va a verificar que el método sea **Get**, ya que es el método estándar para generar peticiones a una página. En **Path** se agrega el

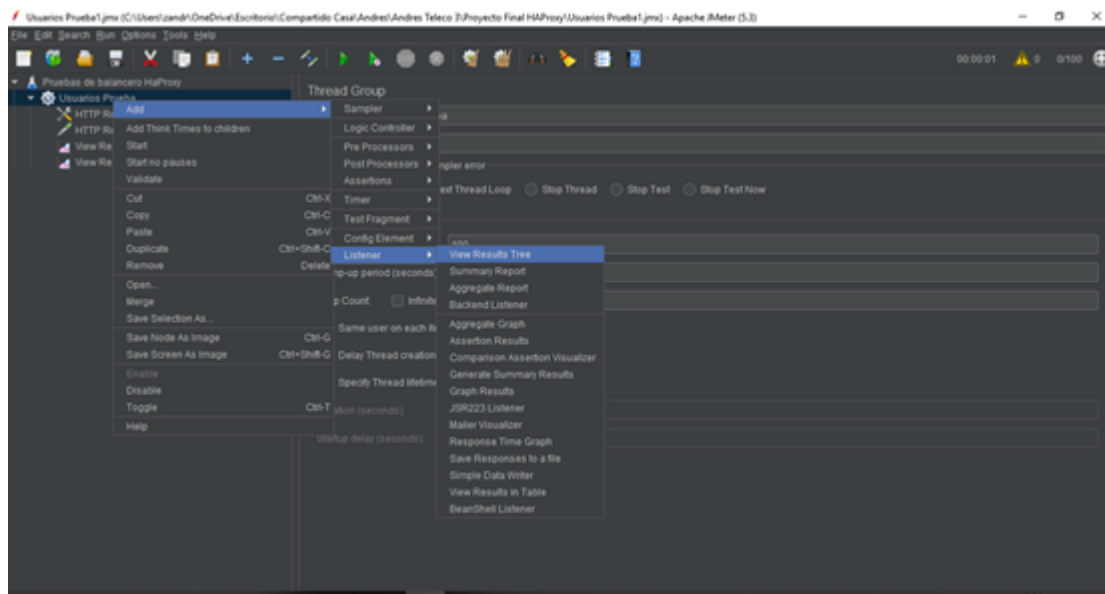
símbolo slash (/), el cual indicará que se quiere enviar las peticiones a la página principal.



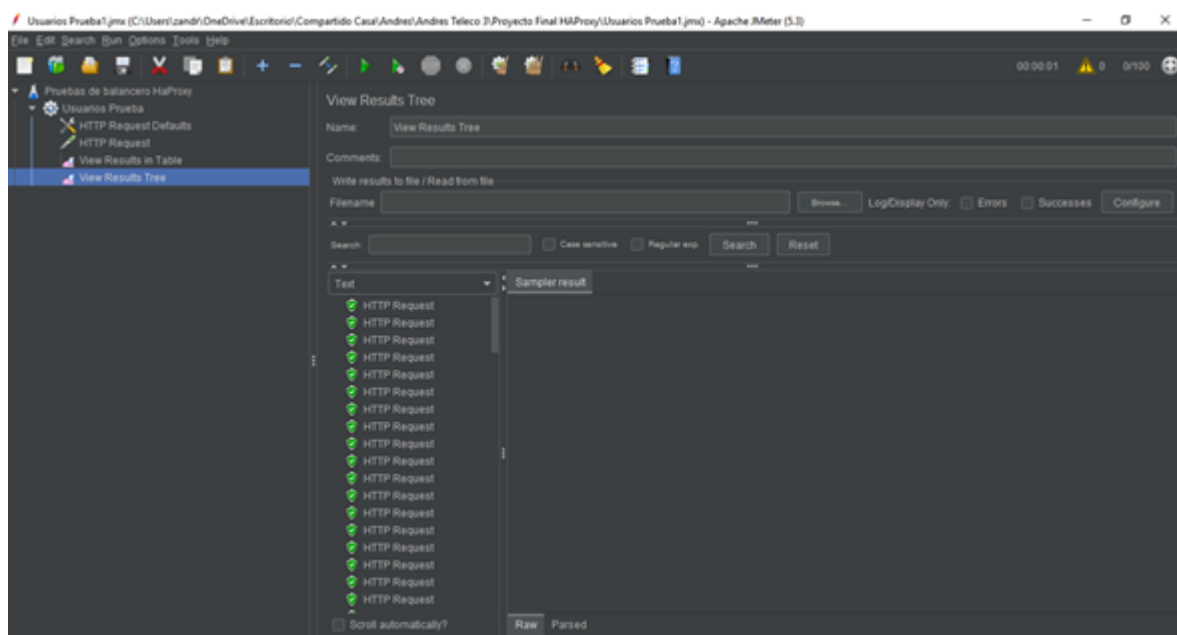
- Dar click en **file** y escoger la opción **save**, para guardar los cambios del generador de peticiones.
- Después de guardar. Correr el proyecto para que se generen las peticiones solicitadas y sean enviadas al balanceador haciendo click sobre la flecha verde que se muestra a continuación





























- Validar funcionamiento del balanceador de cargas:
- Para visualizar los resultados de las peticiones se debe crear primero un **listener**, que permitirá visualizar cada petición y su comportamiento en el balanceador, de la siguiente manera
- Dar click derecho sobre la pestaña usuarios y seleccionar **Add, Listener, View Results Tree**.



- Se observa una nueva pestaña que es creada con el nombre del listener escogido, seleccionar y dar doble click sobre ella, el cual muestra lo siguiente:



- En la columna test se van a encontrar las 2.000 peticiones realizadas al balanceador.
- Para verificar que el balanceador está cumpliendo su función, se validará cada pestaña haciendo click y comprobar que muestre la información del resultado de la petición hecha.
- Seleccionando la opción **Response data** y se puede ver la respuesta del servidor backend al que fue conectado por medio del balanceador.

Sample #	Start Time	Thread Name	Label	Sample Time(ms)	Status	Bytes	Sent Bytes	Latency	Connect Time(ms)
2109	23:26:38.893	Grupo de usuario...	HTTP Request	1052		877	118	1052	1001
2110	23:26:38.993	Grupo de usuario...	HTTP Request	952		864	118	952	0
2111	23:26:38.992	Grupo de usuario...	HTTP Request	961		877	118	961	1
2112	23:26:38.893	Grupo de usuario...	HTTP Request	1060		877	118	1060	1001
2113	23:26:38.893	Grupo de usuario...	HTTP Request	1060		864	118	1060	1001
2114	23:26:38.992	Grupo de usuario...	HTTP Request	961		877	118	961	1
2115	23:26:38.893	Grupo de usuario...	HTTP Request	1061		864	118	1061	1001
2116	23:26:38.993	Grupo de usuario...	HTTP Request	961		864	118	961	0
2117	23:26:38.895	Grupo de usuario...	HTTP Request	1060		864	118	1060	1000
2118	23:26:38.993	Grupo de usuario...	HTTP Request	963		877	118	963	0
2119	23:26:38.892	Grupo de usuario...	HTTP Request	1064		864	118	1064	1004
2120	23:26:38.893	Grupo de usuario...	HTTP Request	1064		877	118	1064	1003
2121	23:26:38.895	Grupo de usuario...	HTTP Request	1062		864	118	1062	1003
2122	23:26:38.992	Grupo de usuario...	HTTP Request	965		877	118	965	1
2123	23:26:38.893	Grupo de usuario...	HTTP Request	1064		864	118	1064	1005
2124	23:26:38.893	Grupo de usuario...	HTTP Request	1065		877	118	1065	1005
2125	23:26:40.051	Grupo de usuario...	HTTP Request	3		2437	0	0	3
2126	23:26:39.931	Grupo de usuario...	HTTP Request	130		2437	0	0	130
2127	23:26:39.944	Grupo de usuario...	HTTP Request	119		2437	0	0	119
2128	23:26:39.945	Grupo de usuario...	HTTP Request	118		2437	0	0	118
2129	23:26:39.943	Grupo de usuario...	HTTP Request	120		2437	0	0	120
2130	23:26:39.944	Grupo de usuario...	HTTP Request	119		2437	0	0	119
2131	23:26:39.944	Grupo de usuario...	HTTP Request	119		2437	0	0	119
2132	23:26:39.944	Grupo de usuario...	HTTP Request	119		2437	0	0	119
2133	23:26:39.945	Grupo de usuario...	HTTP Request	118		2437	0	0	118
2134	23:26:39.944	Grupo de usuario...	HTTP Request	119		2437	0	0	119