# Lab : Implementation of Load Balancer

The role of a load balancer is to improve the availability of services by distributing the load to a pool of back end servers. When it comes to load balancing, Azure has two types of lad balancer

- A **public load balancer** can provide outbound connections for virtual machines (VMs) inside your virtual network. These connections are accomplished by translating their private IP addresses to public IP addresses. Public Load Balancers are used to load balance internet traffic to your VMs.
- An **internal (or private) load balancer** is used where private IPs are needed at the frontend only. Internal load balancers are used to load balance traffic inside a virtual network. A load balancer frontend can be accessed from an on-premises network in a hybrid scenario

**Like many other load balancers, Azure load balancer also has the following components.**

- **Frontend/Virtual IP address** – This is the load balancer IP address that works as a front door to clients. After clients initiate connections to a frontend IP address, the traffic will be distributed to the back-end servers.
- **Server pool** – The back-end application servers will be group together in a pool to serve an incoming request from a load balancer.
- **Rules** – The incoming traffic will be distributed to the backend servers according to the rules defined in the load balancer.
- **Probes** – If a back-end server is down, load balancer needs to know. Then it can stop distributing traffic to the faulty server. The load balancer uses probs to detect the health of the back-end servers.
- **Inbound NAT rules** – Inbound NAT rules define how the traffic is forward from the load balancer to the back-end server.

In this Lab I am going to demonstrate how we can load balance a web application using Azure standard load balancer. This demo includes the following tasks,

*1. Setup new resource group*
*2. Setup two new windows VM*
*3. Setup IIS with sample web page*
*4. Create Azure load balancer*
*5. Create a backend pool*

*6. Create health probes*

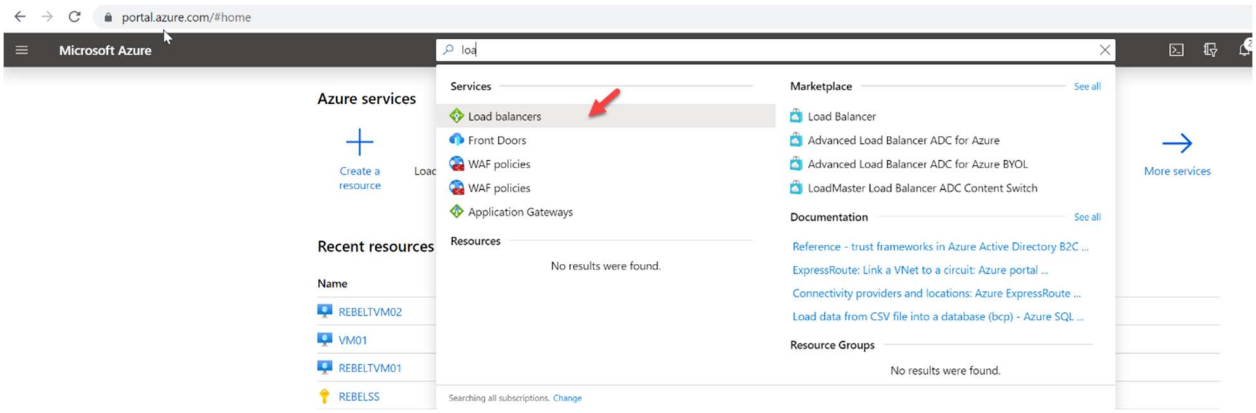
*7. Create load balancer rule*

*8. Testing*

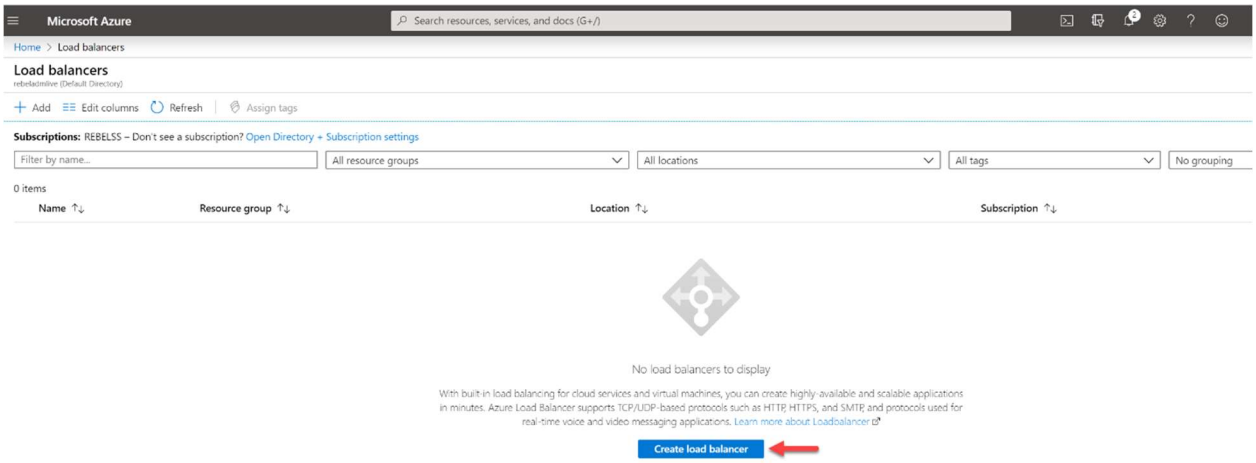
---

Step 1. Log in to Azure portal (https://portal.azure.com/)

Step 2. In the search box type "**load balancer**"




Step 3. Then in load balancer home page click on **Create load balancer**.




Step 4. It will open up the configuration page. In my demo configuration, I am using the following,

**Resource Group** : REBELRG1 ( This is the same resource group I used for VMs and VNet)

**Name** : REBELLB1

**Region** : East US ( Same region as back end servers)

**Type** : Public ( We are going to load balance internet traffic )

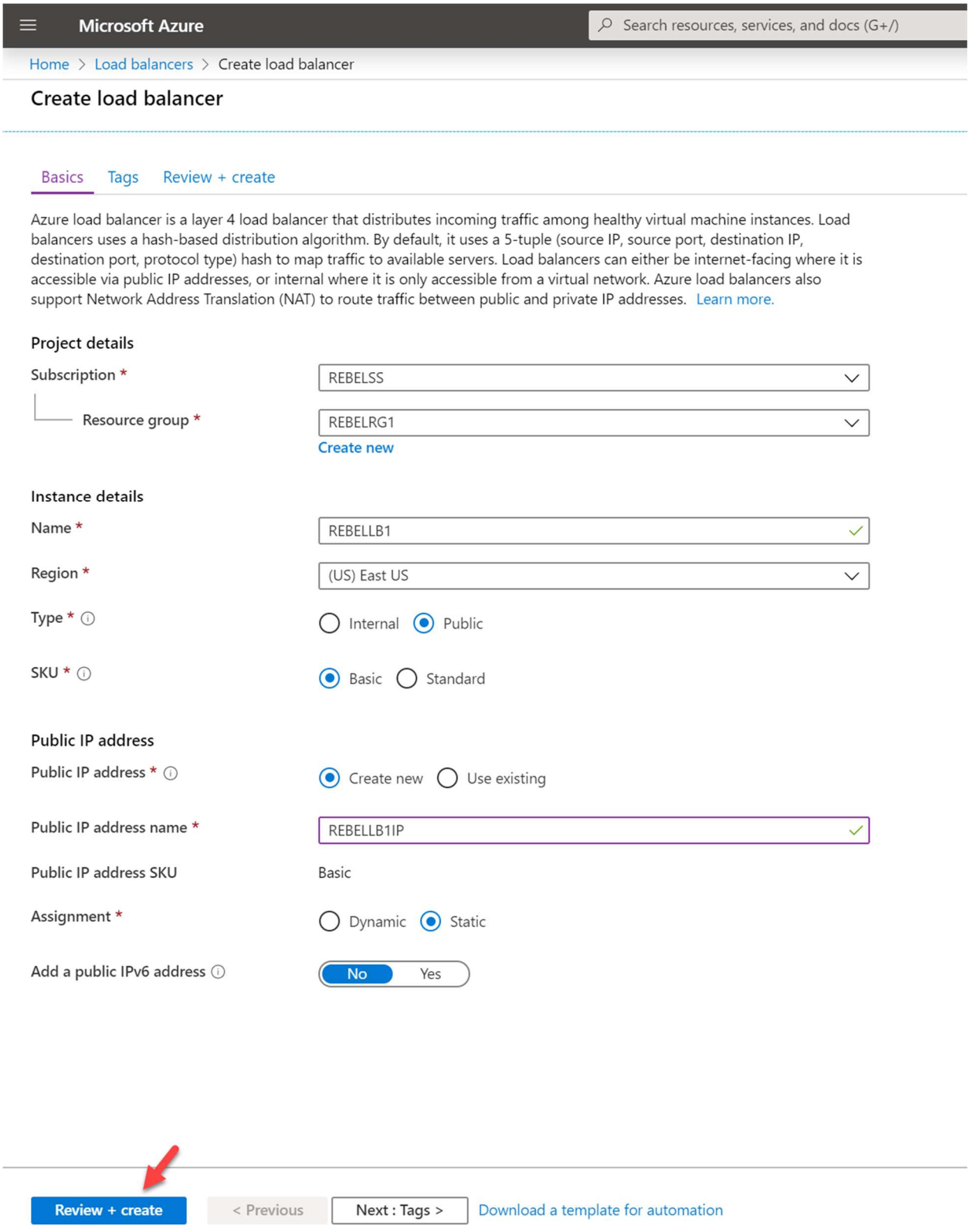
**SKU** : Basic ( Difference between version explained in here https://docs.microsoft.com/en-us/azure/load-balancer/load-balancer-standard-overview)

**Public IP Address** : Create New

**Public IP address name** : REBELLB1IP

**Assignment**: Static

At the end click on **Review + Create** button to create the load balancer.




Step 5. In the next page after config validation, click on **Create** button to complete the process.

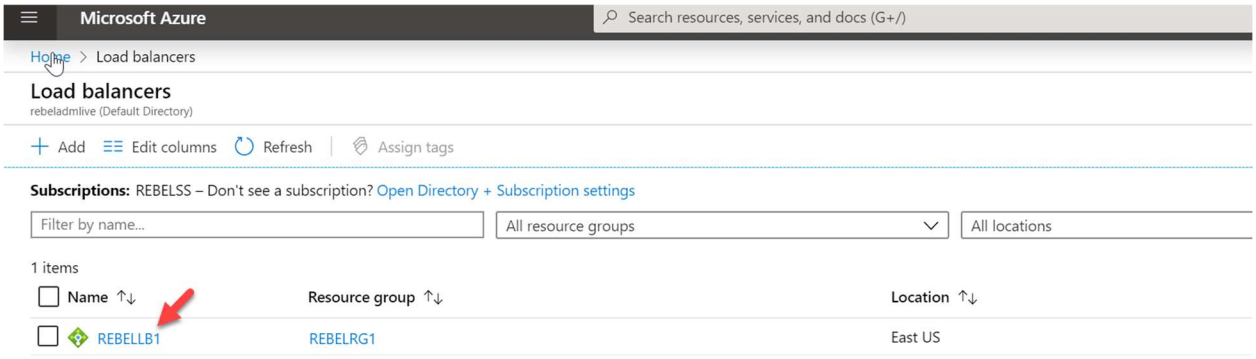
**Create a backend pool**

Step 6 : To create a back end pool with newly added VMs,

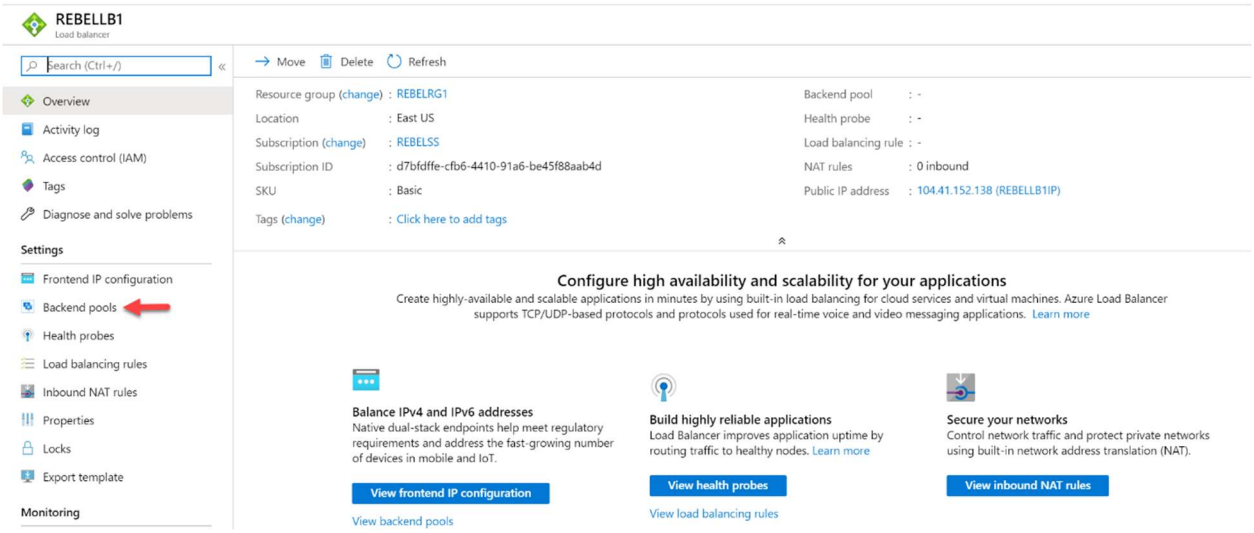
1. Log in to Azure portal (https://portal.azure.com/) as **Global Administrator**

2. In the search box type "**load balancer**" and click on it once it is appearing in the search result.

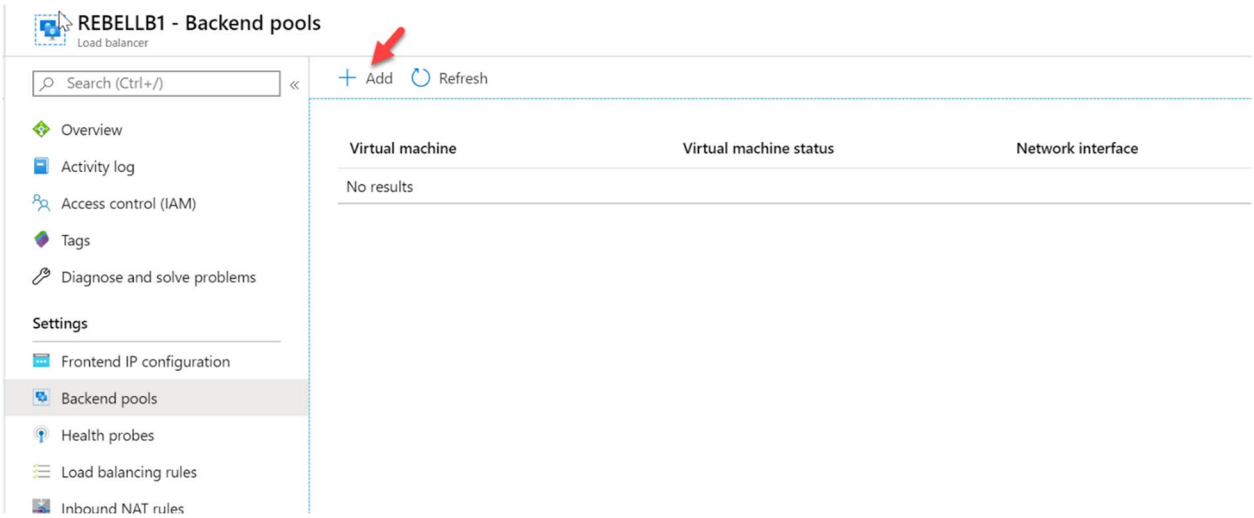
3. Then in load balancer home page click on **REBELLB1**




4. In the properties page, click on **Backend pools**




5. Click on **+Add**




6. In the configuration page, I am using the following settings,

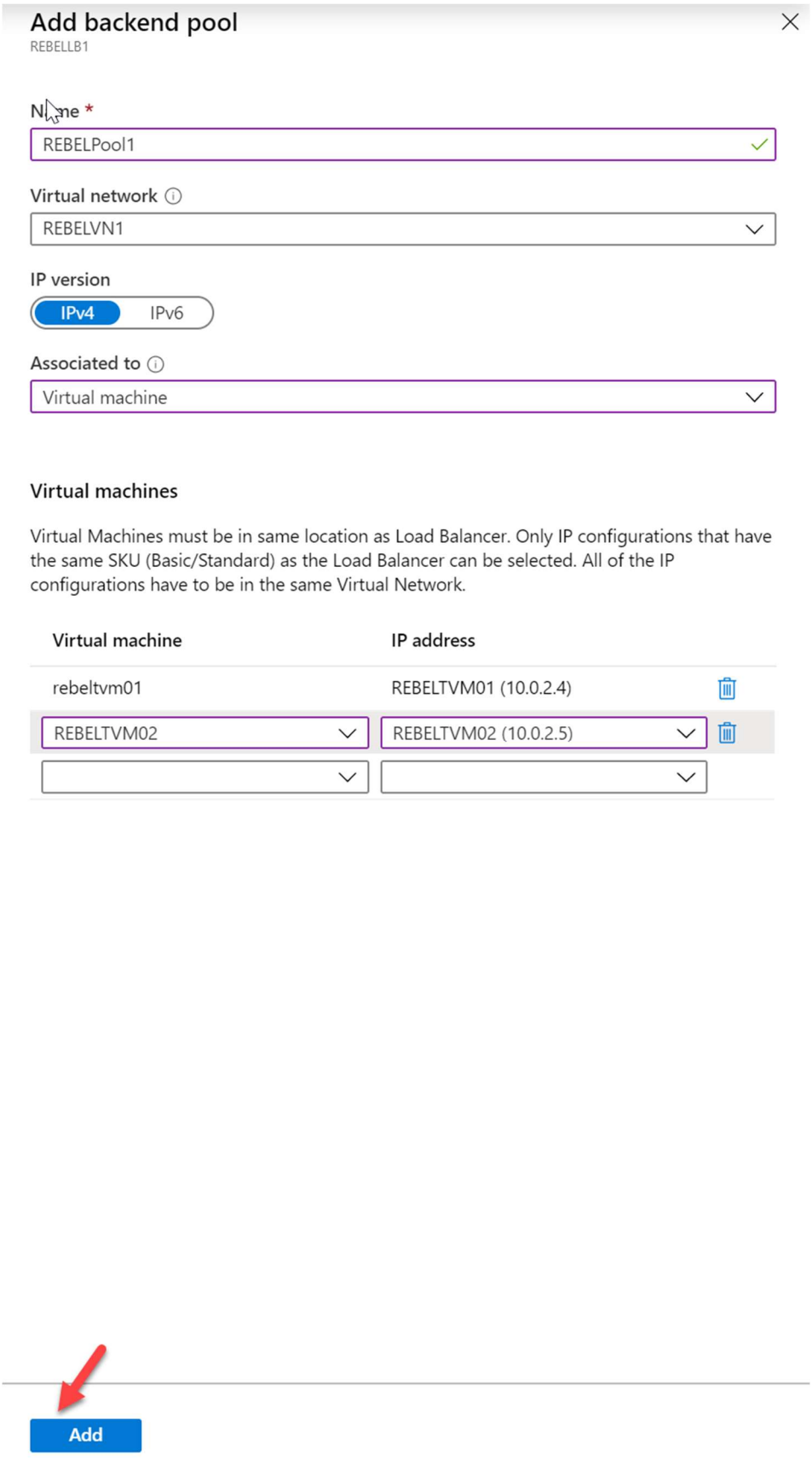
**Name** : REBELPool1

**Virtual Network** : REBELVN1 ( This is the virtual network we setup in earlier step)

**Associated to** : Virtual Machine

Then under the virtual machine section, I have selected the two VM we created in the previous section.

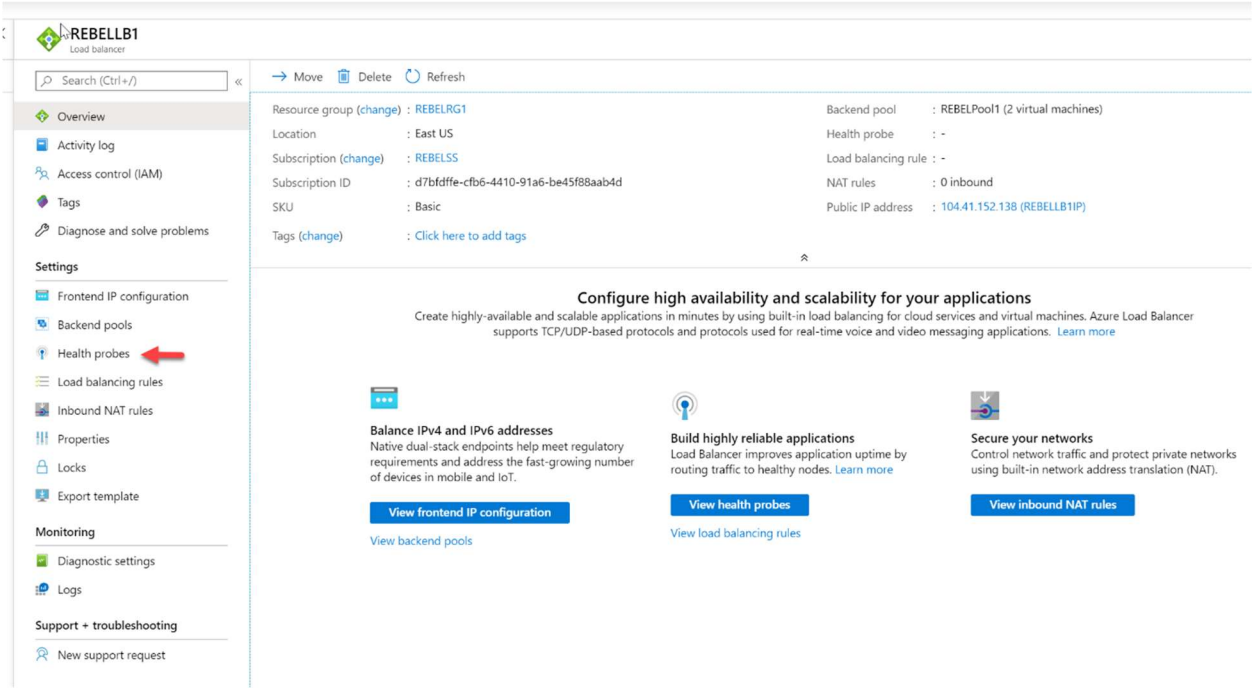
Once settings are in place, click on **Add** button to create a Backend pool.


Add backend pool
REBELLB1
Name *
REBELPool1
Virtual network
REBELVN1
IP version
IPv4
IPv6
Associated to
Virtual machine
Virtual machines
Virtual Machines must be in same location as Load Balancer. Only IP configurations that have the same SKU (Basic/Standard) as the Load Balancer can be selected. All of the IP configurations have to be in the same Virtual Network.
Virtual machine
IP address
rebeltvm01
REBELTVM01 (10.0.2.4)
REBELTVM02
REBELTVM02 (10.0.2.5)
Add

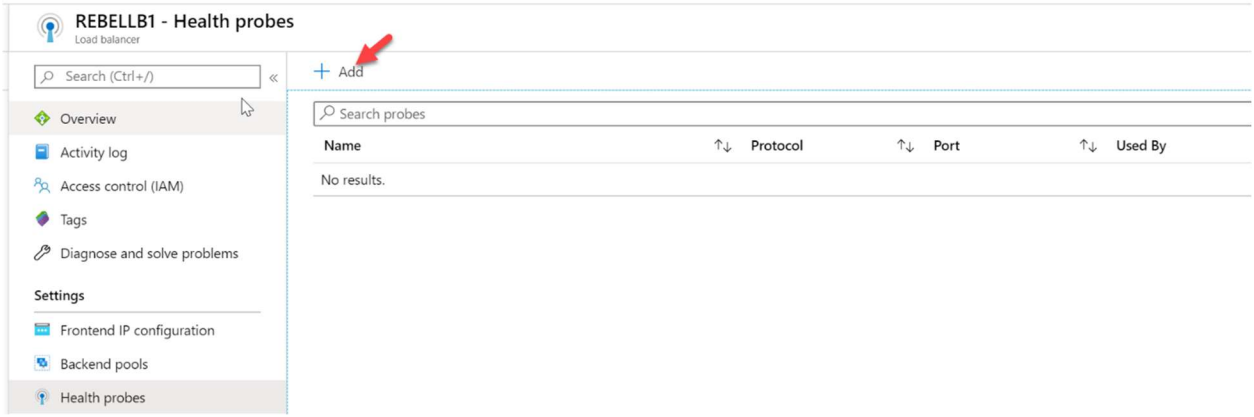
**Step 7 :** Create health probes

We need health probs to monitor the service status of the back-end servers. To setup probe,

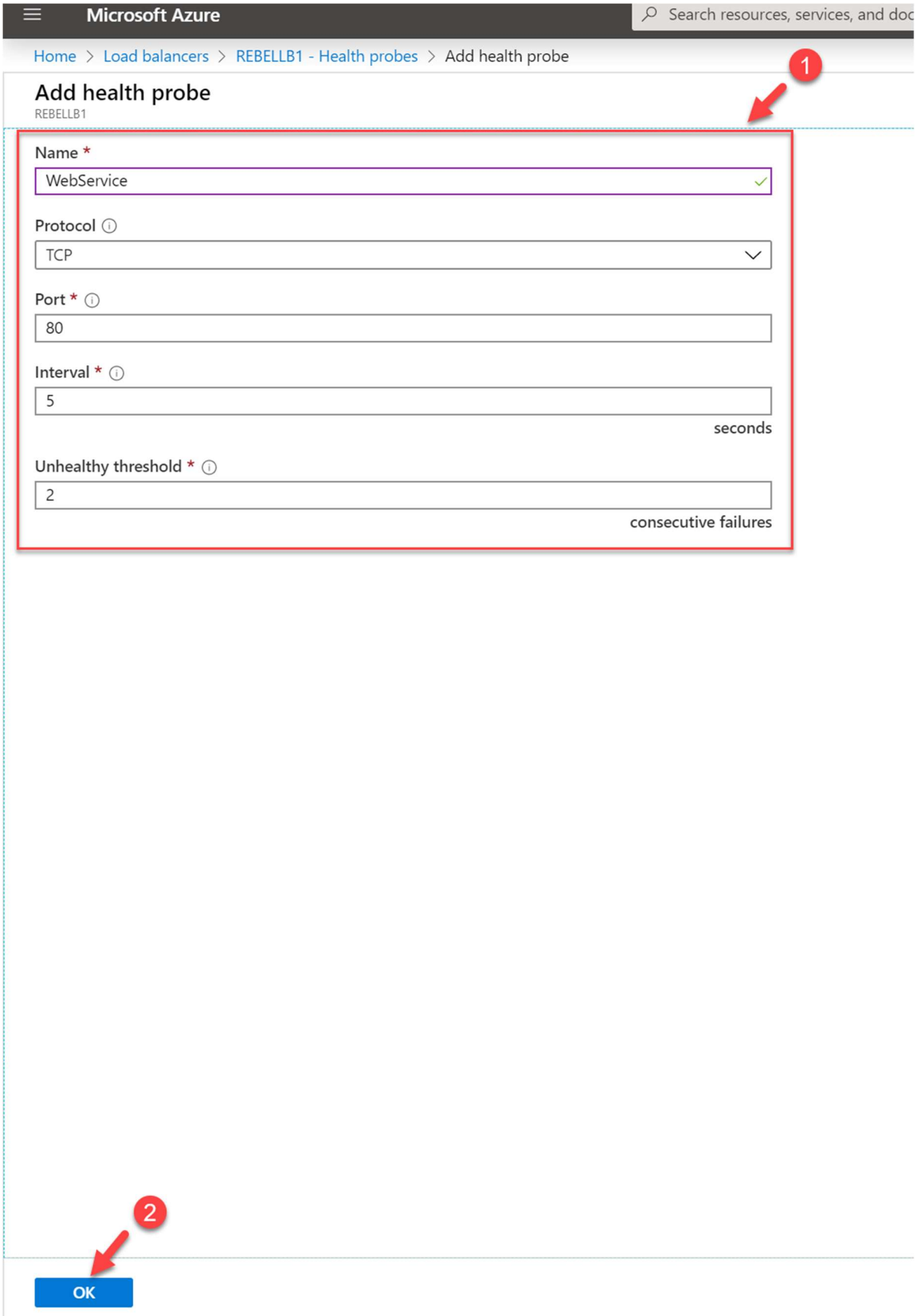
1. Go to **REBELLB1** load balancer properties page
2. Click on **Health Probes**




3. Click on + **Add**




4. In the form provide a name for the probe. Then leave the protocol like **TCP.** We are running web service on **port 80** so leave the default value as it is.


Microsoft Azure
Search resources, services, and doc
Home > Load balancers > REBELLB1 - Health probes > Add health probe
Add health probe
REBELLB1
1
Name *
WebService
Protocol
TCP
Port *
80
Interval *
5
seconds
Unhealthy threshold *
2
consecutive failures
2
OK

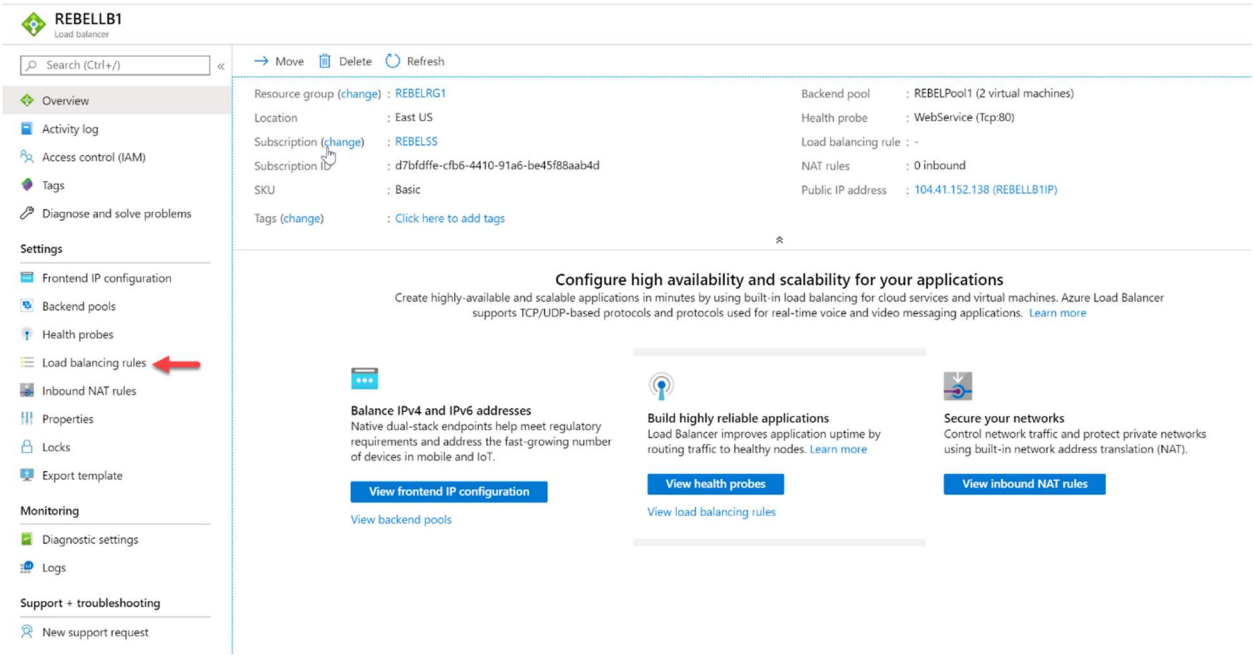
**Step 8 :** Create load balancer rule

Load balancer rule defines how the traffic will be distributed from load balancer to back end pool.

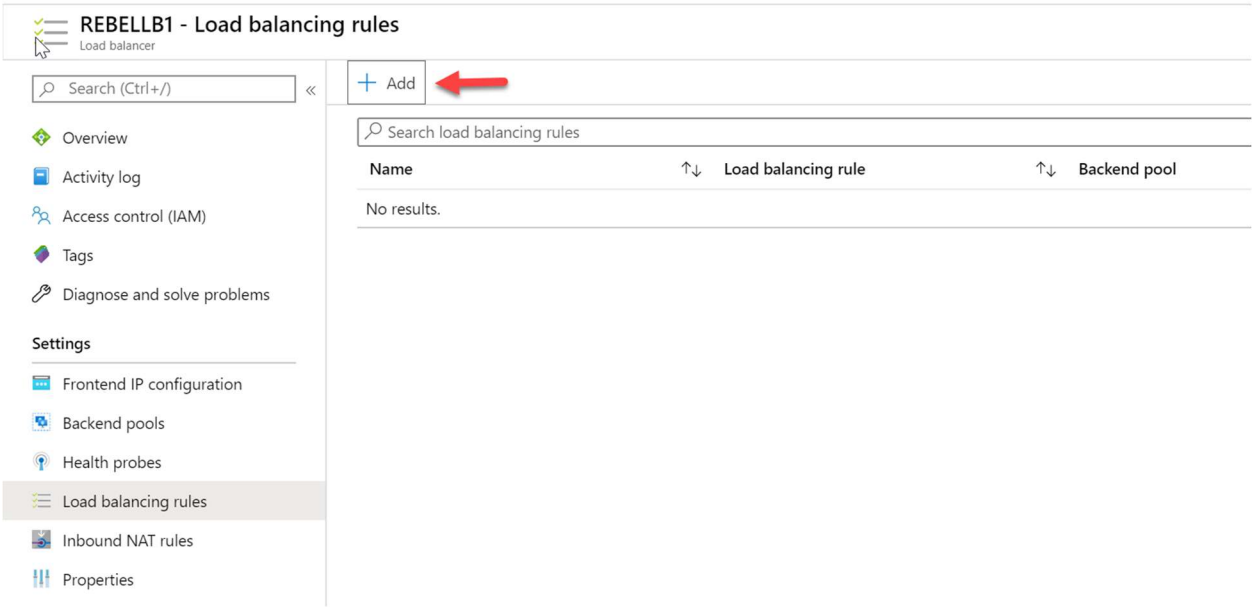
To set up load balancer rule,

1. Go to **REBELLB1** load balancer properties page

2. Click on **Load balancing rules**




3. Click on + **Add**




4. In my setup, I am load balancing **TCP 80** traffic. So my rule configuration as following,

**Name** : LBRule1

**IP Version**: IPv4

**Front End IP address** : Load balancer IP address

**Protocol** : TCP

**Port** : 80

**Backend port** : 80

**Backend pool**: REBELPool1

**Health probe**: Webservice

Once relevant configuration in place, click on **OK** to create the rule.

## Add load balancing rule

REBELLB1

Name *

LBRule1

IP Version *

(•) IPv4 ( ) IPv6

Frontend IP address * ⓘ

104.41.152.138 (LoadBalancerFrontEnd)

Protocol

(•) TCP ( ) UDP

Port *

80

Backend port * ⓘ

80

Backend pool ⓘ

REBELPool1 (2 virtual machines)

Health probe ⓘ

WebService (TCP:80)

Session persistence ⓘ

None

Idle timeout (minutes) ⓘ

4

Floating IP (direct server return) ⓘ

Disabled Enabled

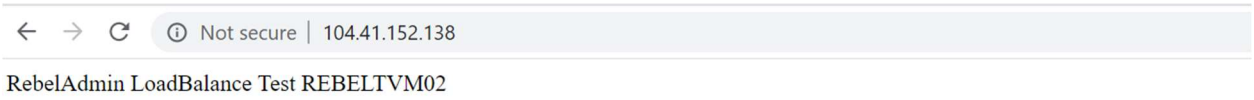
OK

**Step 9 :** Testing

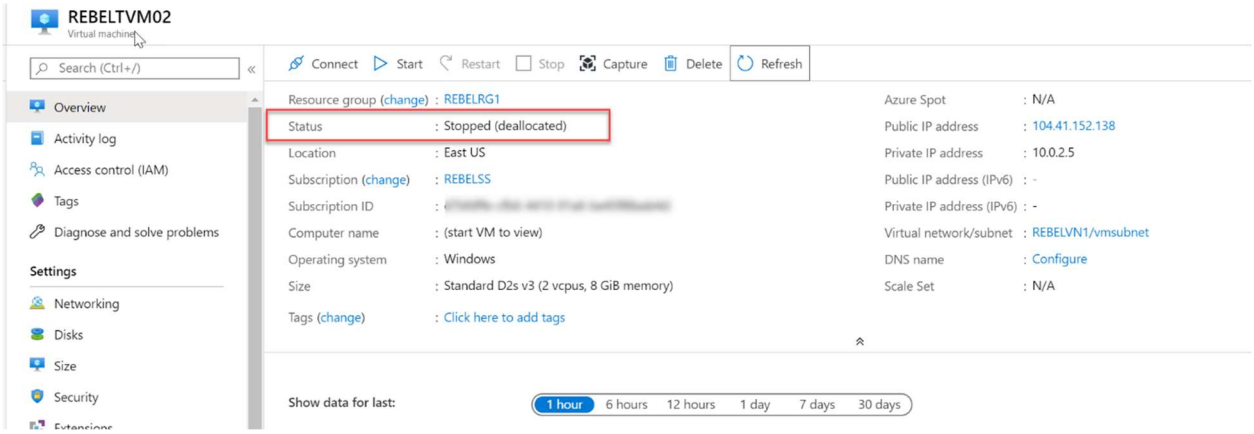
This completes the configuration. It is time for testing.

I went ahead and launch the web browser of my laptop and try to access public ip address of the load balancer.

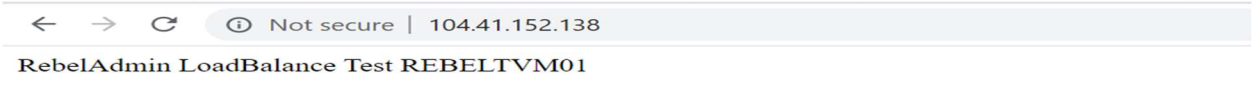
As expected, now I can see the web site running from **REBELTVM02** back end server.




Then I went ahead and shutdown the **REBELTVM02** back end server.




When I refresh the web page again, now I can see the web page from **REBELTVM01** back end server.




This confirms the load balancer is working as expected.