**Designing Cloud Application for Resiliency**

* Application Design Practices for Highly Available Applications
* Application Analytics
* Building High Performance Applications by Using Asp.net
* Common Cloud Application Patterns
* Caching Application Data

**Application Design Practices for Highly Available Applications**

Partitioning Workload – workload is divided into segments

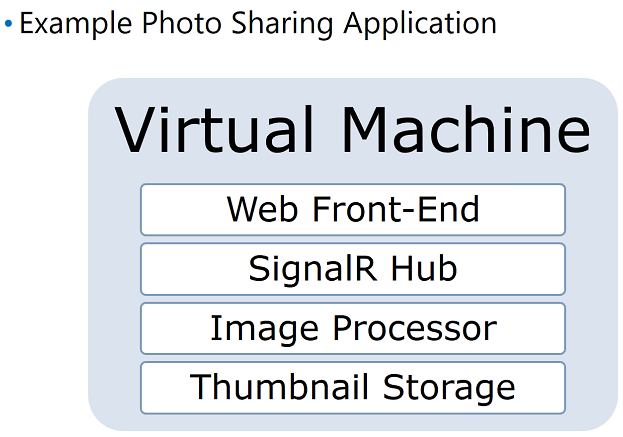
E.g. – I have an application divided into front end and backend. Dividing the workload so that management is much easy.

Scaling up or down for a particular component is another reason.

When designing web applications, split your business processes into partitioning workloads.

Partitioning workload can be handled in modular websites, cloud services, or virtual machines.

It provides the ability to scale the different components of your application in isolation.

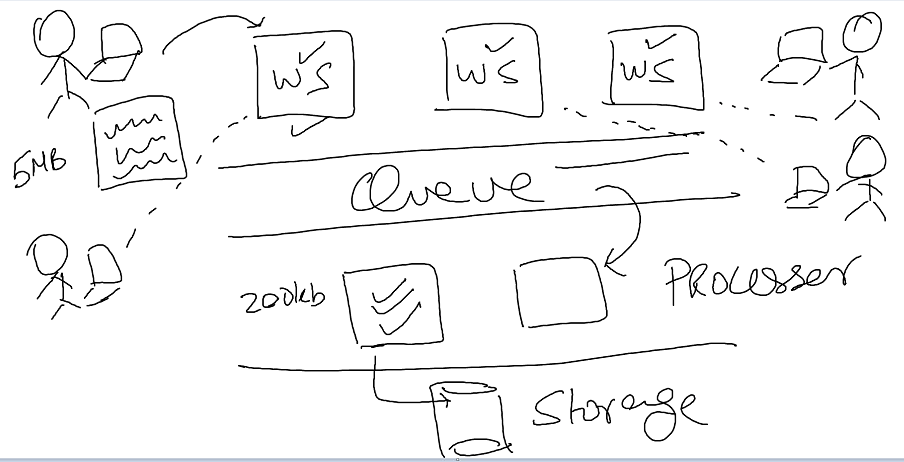


I have 3 machines which handle the web role/website. Then I have queues, then I have my image processors and I have my storage.

Suppose a user uses this website. The application allows the user to save information along with his image. When the user upload the image. The picture is uploaded at the web end. Then the processor compressed the picture and then save the information to storage account.

Suppose more users trying to access this application and upload their images and the processor is already doing its task in that case, the coming tasks will be stored in queue.

But in queue only information is stored that processor is doing some work, and there are certain requests which needs to be processed and are stored in storage.



Load Balancing

* Provide same service from many instances and use a load balancer to distribute requests across all of the instances.
* Considerations for selecting a load balancing strategy:

Hardware or software load balancers

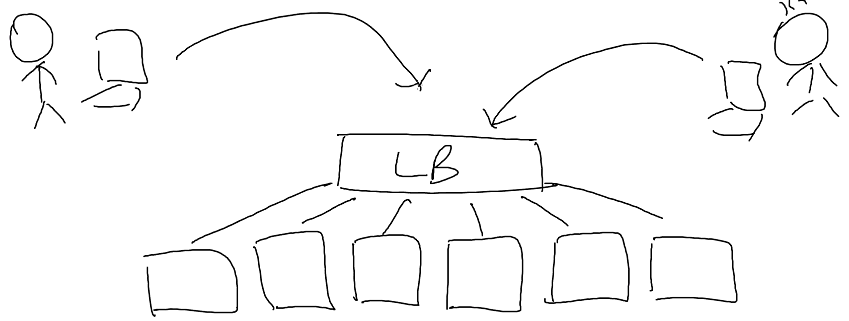
Load balancing algorithms (round robin)

Load balancer stickiness

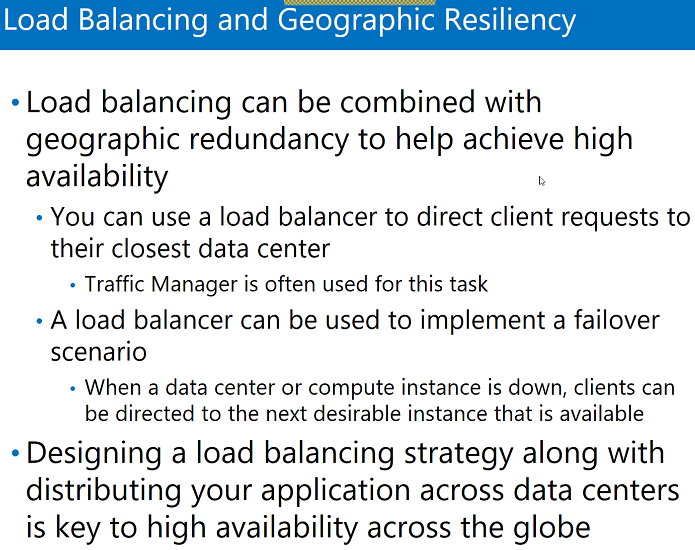
* Load balancing becomes critical even if you have a single service instance as it offers the capability to scale seamlessly.

e.g.

I have servers and two users access these servers. There are many servers how user will decide which server to user. Actually load balancer balancing the load and distributing the traffic among the servers.



There are four types of load balancer:



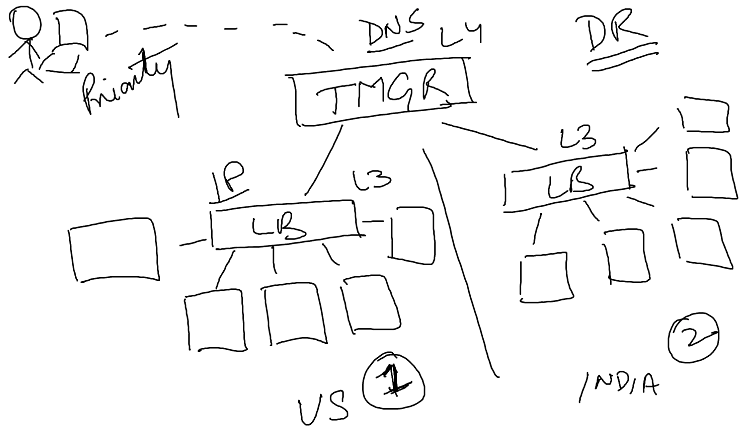
Traffic Manager and Load Balancer

Let’s we have one traffic manager and this TM redirects two load balancer and LBs will have applications running on VMs. Let’s One is US and another is India.

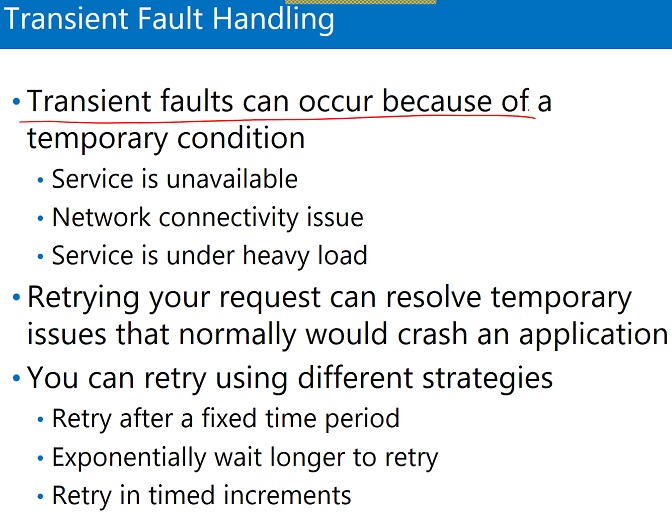
1. So if user is login from US he will be redirected to US and if he is login from India he will be redirected to India.
2. Suppose US site is the main site and India site is the secondary site, so the user will always redirected to site US and if there is any server down, the user will redirected to secondary site. This is Disaster recover scenario.

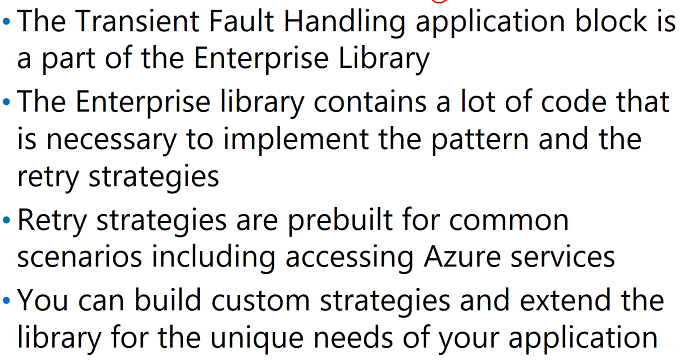
Difference:

1. TM works on L4 (DNS) layer and LB works on L3 (IP addresses) layer.
2. TM works on different regions and LB works on same region.

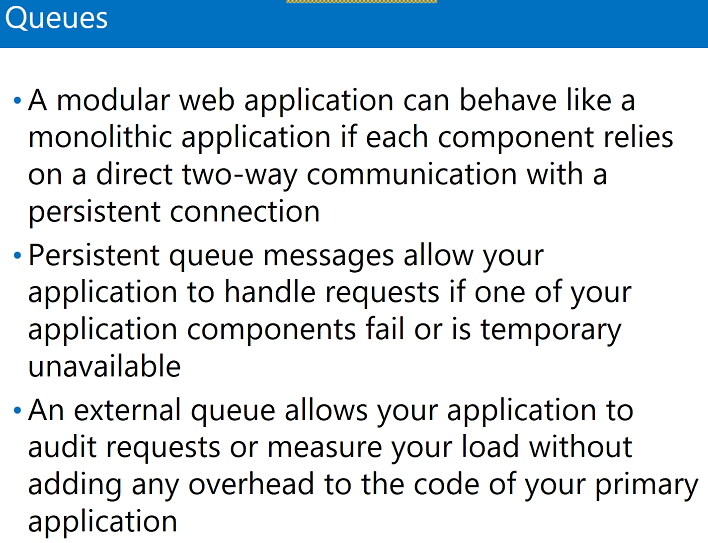


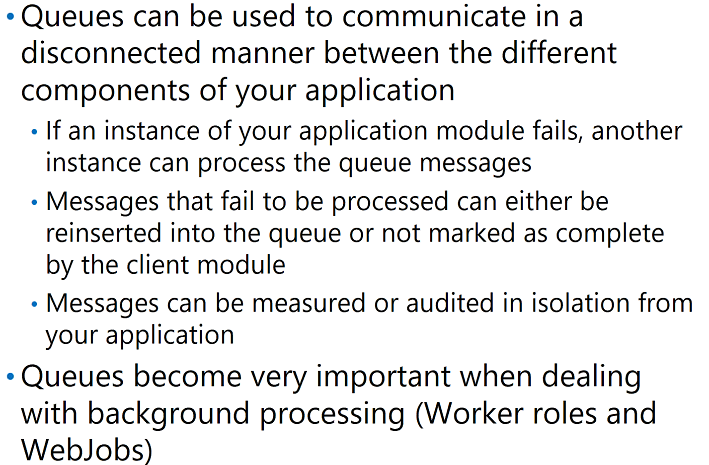
Transient Fault Handling





Queues





**Application Analytics**

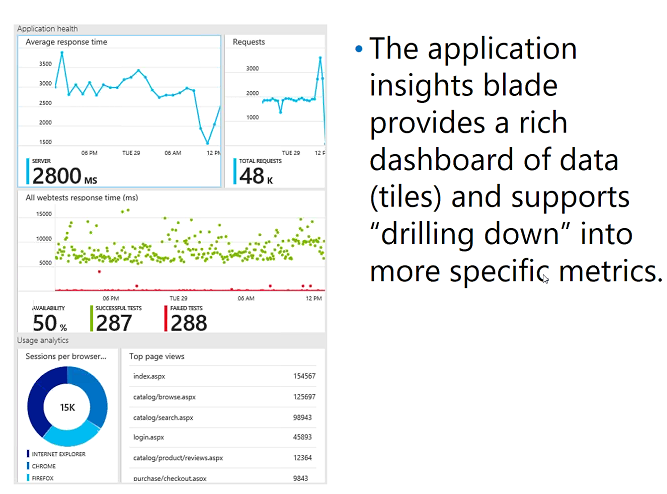


Application Insights is an analytics and monitoring service available for our applications (getting the complete information in terms of analytics – App dynamics).

* It monitors the performance of app
* Monitor traffic if we need more server etc
* Cicso bought recently
* View exception stack traces
* Monitor CPU and resource usage
* Periodically test URLs from worldwide data centers
* Monitor usage of our application and most popular requests

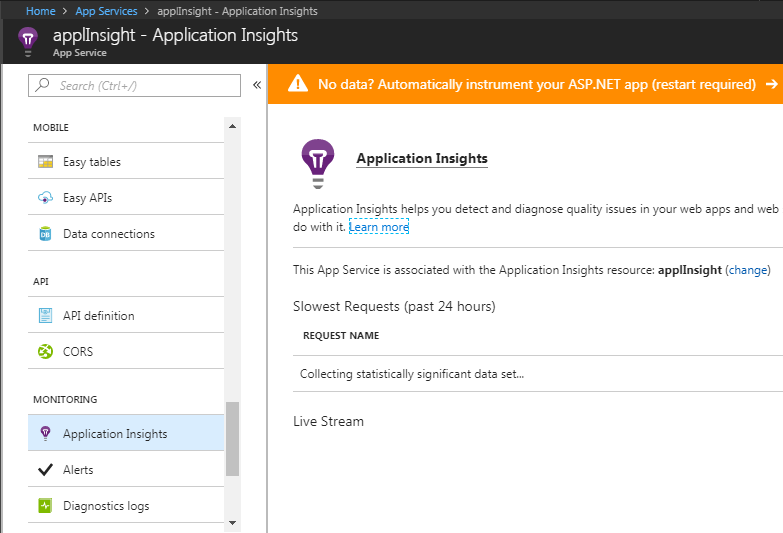
It can be used with .NET or JAVA

Applications do not specifically need to be hosted in Azure



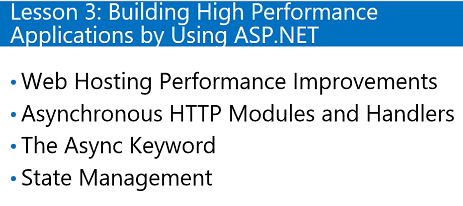
Labs:

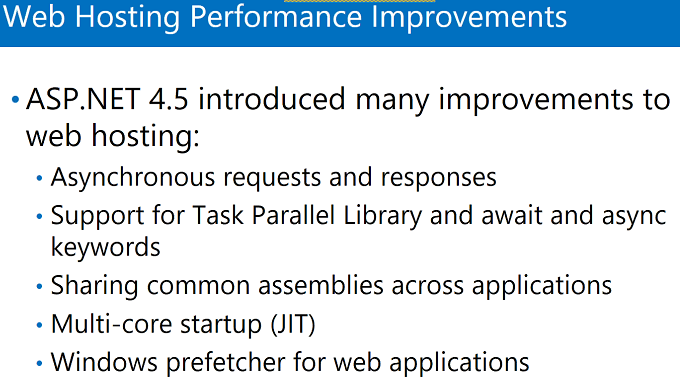
1. Create Web Application
2. Turn on Application insight
3. Create
4. Create web app in Visual studio
5. Right click application And add application insight

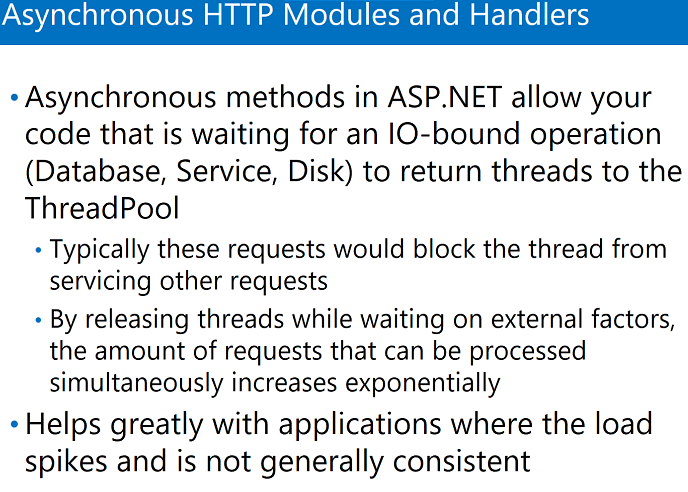


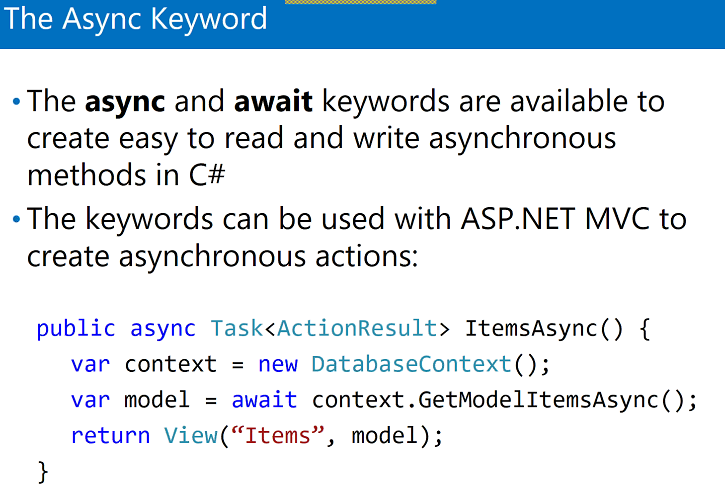
Labs:

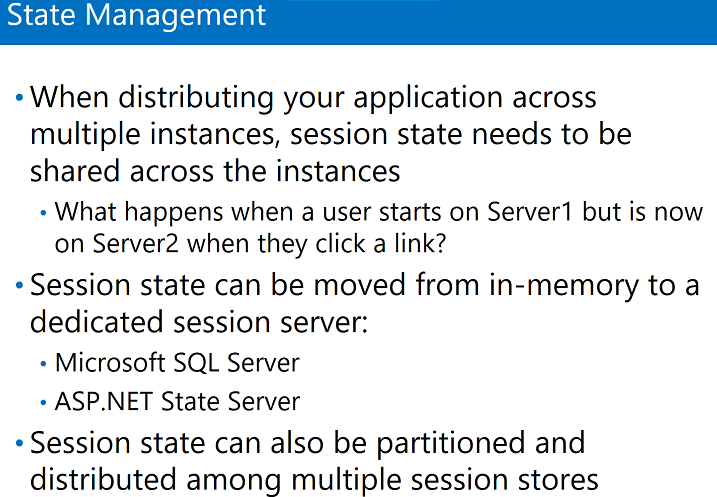
1. Monitors for application we don’t need application insight

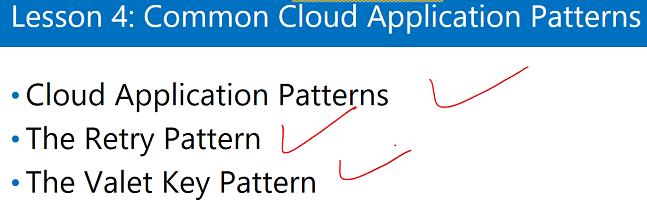


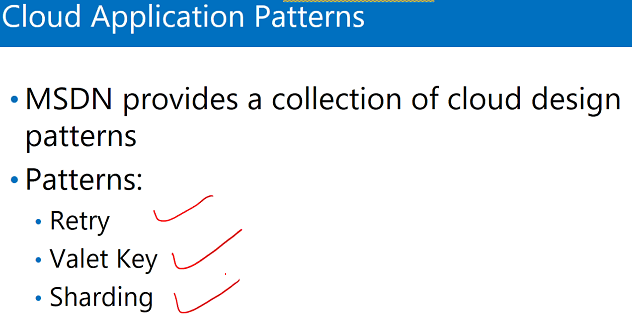


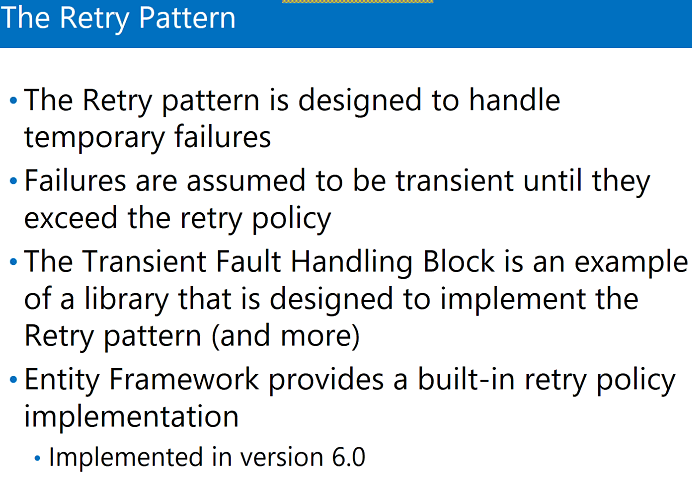


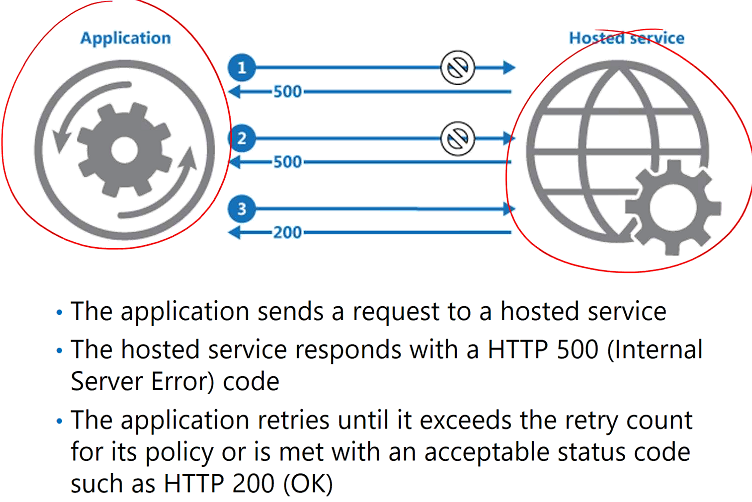


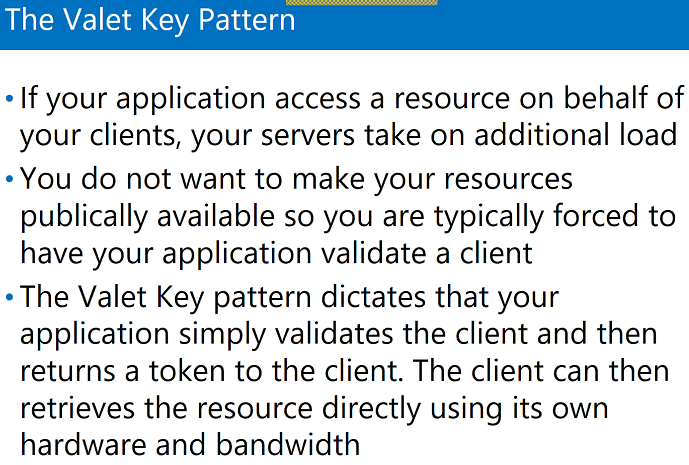


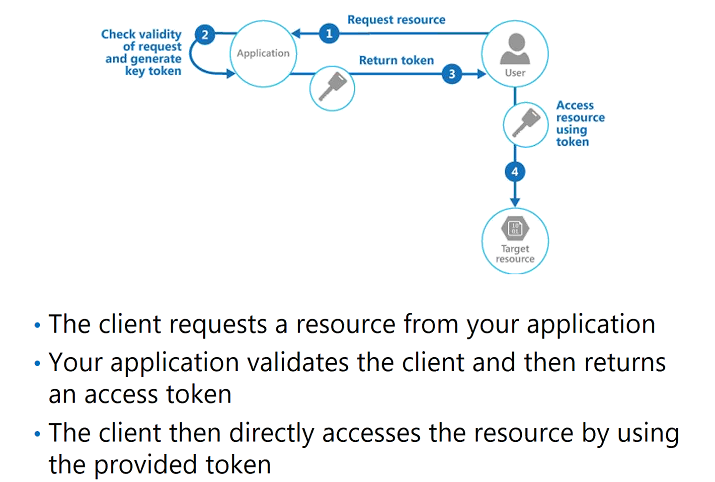


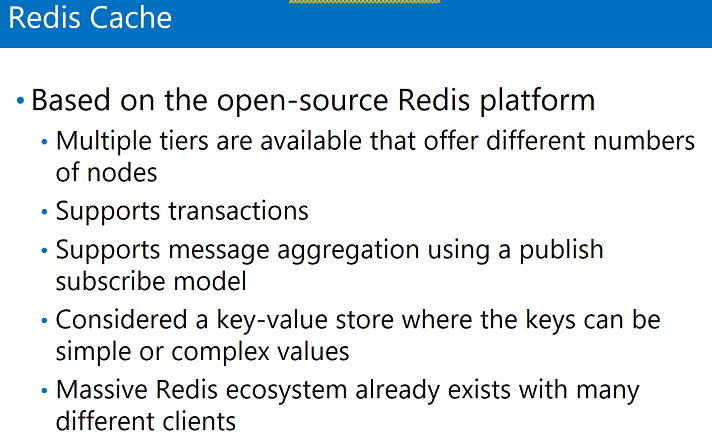










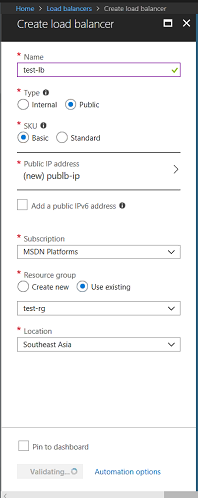


**Labs: Load Balancer**

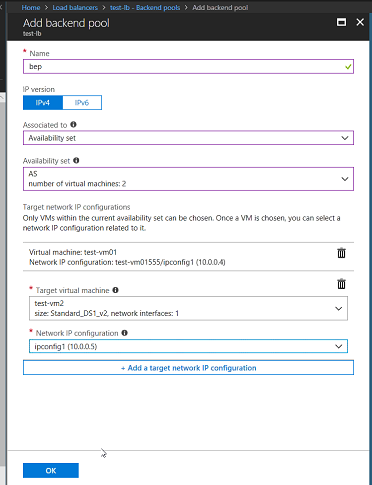
1. Create two VMs (keep both the VMs in same availability set and same network)
2. Set no firewall
3. Create load balancer

Load balancer are of two types internal and public

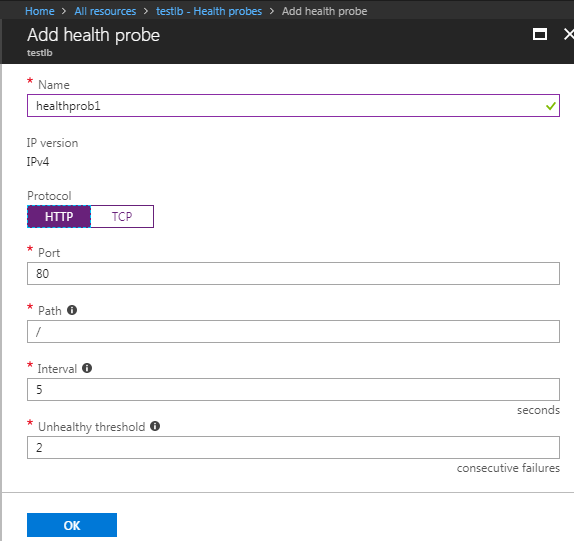
SKU is new, availability zone is new (related to infrastructure)



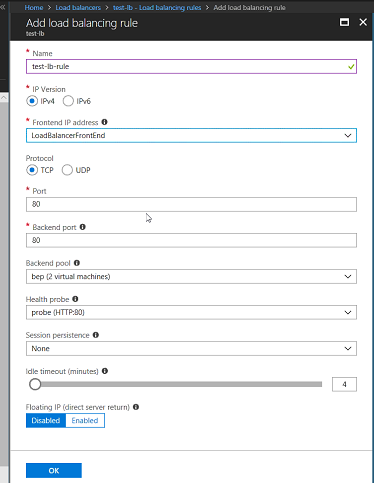
1. Backend pool is where traffic will be routed.



1. Health probes (if one VM stopped working then Load balancer should not send any connection to that VM, health probe keeps checking the service)



1. Load Balancing rules



Load Balancer is ready.

1. If we will install IIs on both the Machines then our Load Balancer will work.

Labs: **Traffic Manager**

1. Create app service in Central Us
2. Create another web app in southeast Asia
3. Add traffic manager

All resources are in different resource group

1. Upload a file using FTP in one web app
2. Go to traffic manager and add two end points
3. Run the respective web app in particular country.

User will get the app which is near to the user’s data center

1. Add another load balancer

