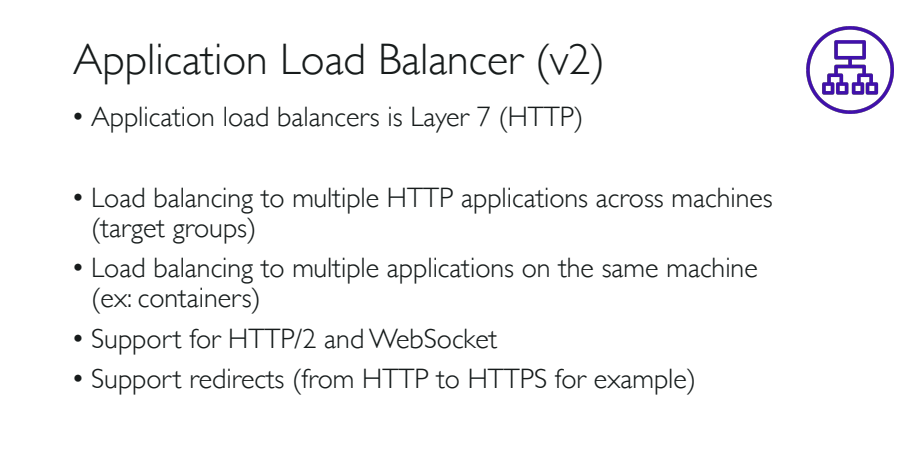
**Application Load Balancer (ALB):**



NOTE:

**Layer 7 load balancing** allows the **load balancer** to route a request based on information in the request itself, such as what kind of content is being requested. So now a request for an image or video can be routed to the servers that store it and are highly optimized to serve up multimedia content.

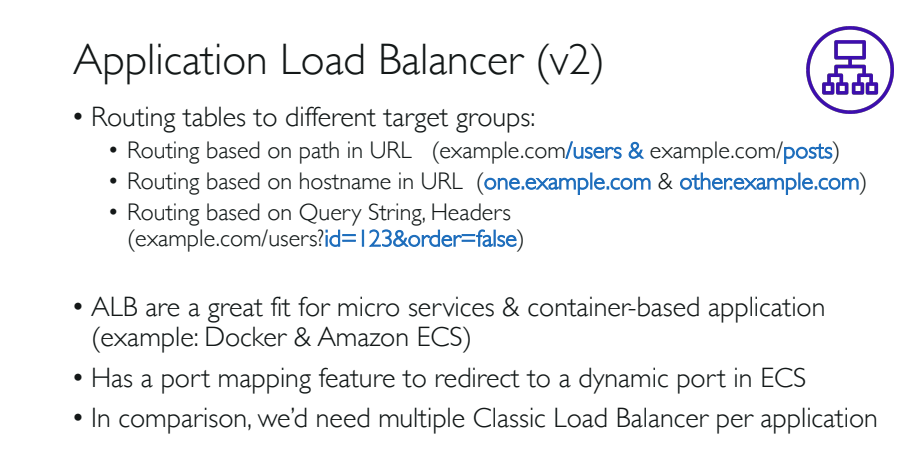
At **Layer 4**, a **load balancer** has visibility on network information such as application ports and protocol (TCP/UDP). ... At **Layer 7**, a **load balancer** has application awareness and can use this additional application information to make more complex and informed **load balancing** decisions.

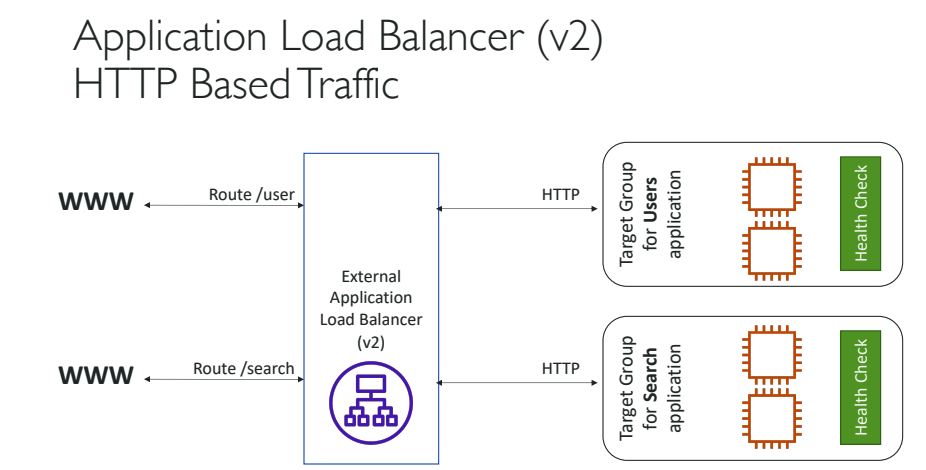
NOTE:

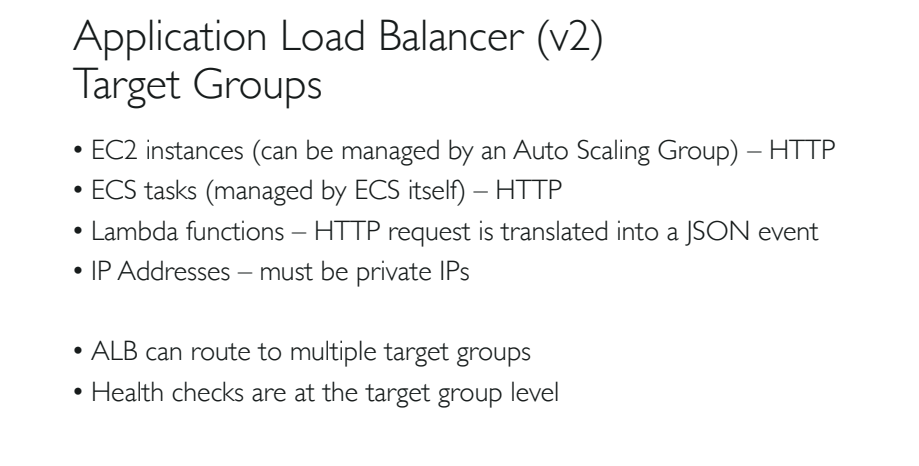
**HTTPS** secures the connection between a visitor's browser **and** the web server, ensuring data is encrypted **and** protected from intermediary actions. **HTTP**/**2** allows for multiplexing (requesting multiple files at the same time). This significantly improves both site performance **and** server efficiency.

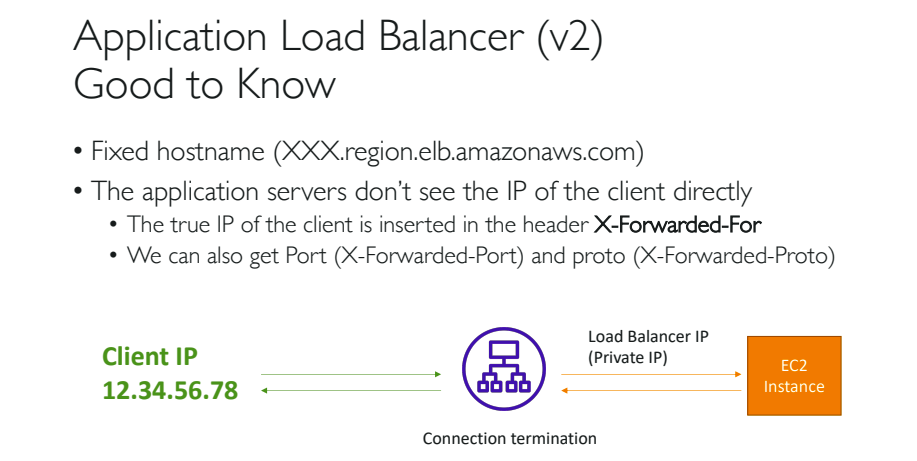
NOTE:

The **WebSocket** API is an advanced technology that makes it possible to open a two-way interactive communication session between the user's browser and a server. With this API, you can send messages to a server and receive event-driven responses without having to poll the server for a reply.

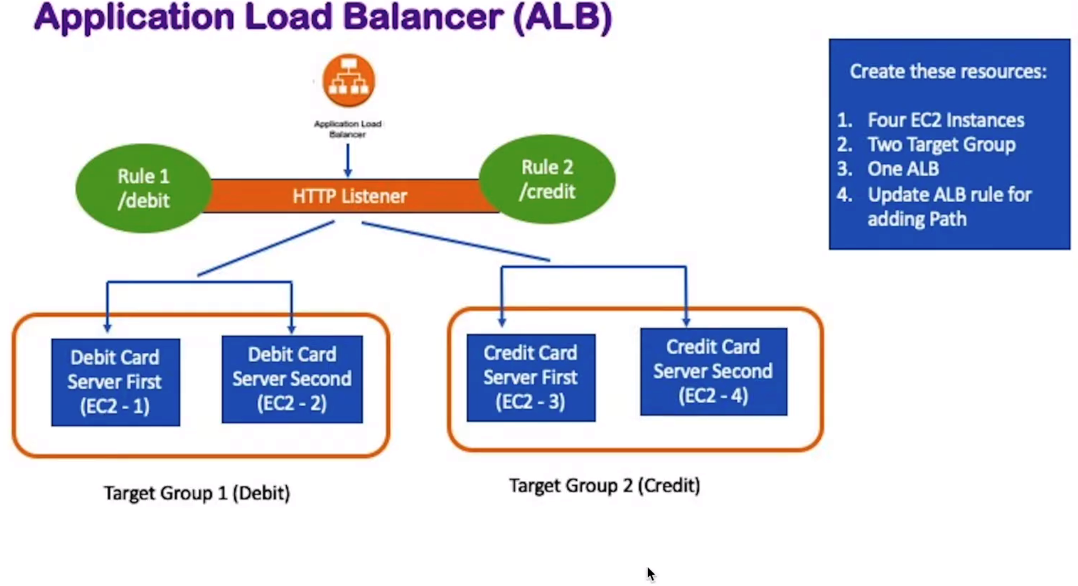








**Below is an example scenario to understand ALB more clearly.**



1. First create four instances which has the following script pasted in the ‘User Data’ while creating each instance.
2. Name each instance as ‘Debit Server-1’, ‘Debit Server-2’, ‘Credit Server-1’ and ‘Credit Server-2’.
3. This script will create html files in the given location of the instance and also install and start the apache ‘httpd’ web server in each instance.

**For Instance, 1:**

#! /bin/bash  
sudo su  
yum install httpd -y  
echo "<html><h1>This is Debit Card Processing - First  
Server</h1></html>" > /var/www/html/index.html  
echo "<html><h1>Please wait until we process your debit card  
transaction from Debit server First</h1></html>" >  
/var/www/html/debit.html  
service httpd start  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**For Instance, 2:**

#! /bin/bash  
sudo su  
yum install httpd -y  
echo "<html><h1>This is Debit Card Processing - Second  
Server</h1></html>" > /var/www/html/index.html  
echo "<html><h1>Please wait until we process your debit card  
transaction from Debit server Second</h1></html>" >  
/var/www/html/debit.html  
service httpd start  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**For Instance, 3:**

#! /bin/bash  
sudo su  
yum install httpd -y  
echo "<html><h1>This is Credit Card Processing - First  
Server</h1></html>" > /var/www/html/index.html  
echo "<html><h1>Please wait until we process your debit card  
transaction from Credit server First</h1></html>" >  
/var/www/html/credit.html  
service httpd start  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**For Instance, 4:**

#! /bin/bash  
sudo su  
yum install httpd -y  
echo "<html><h1>This is Credit Card Processing - Second  
Server</h1></html>" > /var/www/html/index.html  
echo "<html><h1>Please wait until we process your credit card  
transaction from Credit server Second</h1></html>" >  
/var/www/html/credit.html  
service httpd start  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Allow port 22 for SSH and port 80 for HTTP while creating instances, in the option ‘Configure Security Groups’
2. You can select an ‘existing keypair’ or ‘create a new keypair for the instances.
3. Copy the ipv4 address of each instance and paste it on the browser url to test its implementation. This will execute the html code written on the “User Data” while creating the instance.

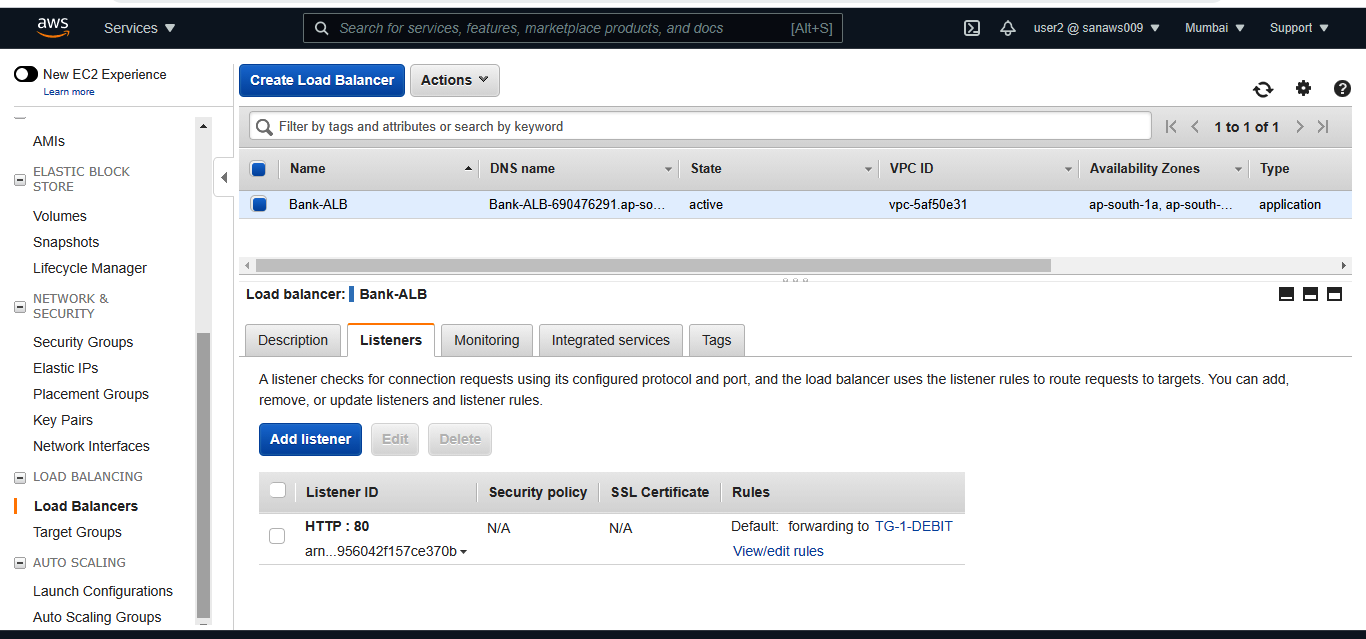
**NOTE**: Check the security group, “user Data” script or manually execute the script by logging into to instance if the script does not execute.

**NOTE**: The ‘User Data’ Script will only execute once at the time of instance is first started.

1. Now we create two ‘Target Groups’ and assign ‘Debit Server-1’ and ‘Debit Server-2’ to the First Target Group ‘TG-Debit’ and ‘Credit Server-1’ and ‘Credit Server-2’ to the second Target Group ‘TG-Credit’.
2. To create Target Group, go to the left-hand side menu bar and select ‘Load Balancing’ option where you can find ‘Target Groups’ option.
3. Allow port 80 for HTTP while creating TG.
4. In ‘Register Targets’ option we can assign instances to the TG.

**Creating Application Load Balancer and assigning TG’s:**

1. To create LB, go to menu bar on left hand side and select ‘Load Balancers’ under ‘Load Balancing’ menu.
2. Select Application Load Balancer.
3. While Configuring LB, allow port 80 for HTTP and select all availability zones.
4. Select a SG which supports HTTP and SSH.
5. In Step 4: Configure Routing, select ‘Existing Target Group’ and select the TG if already available or we can create and assign instances in this step.
6. In the Name option of Configure Routing, if no page is specified then the default page will be the TG specified in the Name option by default.
7. Go to LB and check which TG are assigned to the LB.
8. We can see this by selecting ‘Listeners’ tab under ‘Load Balancers’ like the image below.



1. We can always view/edit the rules of the load balancer here.

NOTE: Add, edit or remove new rules by getting into the above mention option.

