ELASTIC LOAN BALANCING

* Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP address, and Lambda functions.
* It can handle the varying load of your application traffic in a **single Availability Zone or across multiple Availability Zones**.
* Elastic Loan Balancing offers three types of load balancers that all feature the high availability, automatic scaling, and robust security necessary to make your applications fault tolerant.

Three are 3 types of Load Balancers

* **Application Load Balancer**
* **Network Load Balancer**
* **Classic Load Balancer (works only with EC2 Classic)**

Why we need Elastic Load Balancer

* + Spread load across multiple downstream instances
  + Expose a single point of access (DNS) to your application
  + Do regular health checks to your instances
  + Provide SSL termination (HTTP) for websites
  + Enforce stickiness with cookies
  + High availability across zones
  + Separate public traffic from private traffic
  + Seamlessly handle failures of downstream instances

Load Balancers

* Classic Load Balancers are Deprecated
* Application Load Balancers for HTTP / HTTPs
* Network Load Balancer for TCP
* CLB, ALB & NLB support SSL certificates and provide SSL termination
* All Load Balancers have health check capability
* ALB can route on based on hostname / path
* ALB is a great fit with ECS (Docker)

Any Load Balancer (CLB, ALB, NLB) has a static host name. Do not resolve and use underlying IP

**APPLICATION LOAN BALANCER**

* Operates at the individual request level (Layer 7), Application Load Balancer routes traffic to a target within Amazon Virtual Private Cloud (Amazon VPC) based on the content of the request.
* Application Load Balancer is best suited for load balancing of HTTP and HTTPS traffic and provides advanced request routing targeted at the delivery of modern application architectures, including microservices and containers.
* Basically, they’re awesome for micro services & container-based application (example: Docker & Amazon ECS)
* Load balancing has ***path based* and *host based*** in URL
* Support routing based on hostname

(users.example.com & payments.example.com)

* Support routing based on path (example.com/users & example.com/payments)
* Support for micro services and container-based applications
* Content-based routing allows requests to be routed to different applications behind the single load balancer
* If we want to route the traffic to multiple TG and multiple application, we need to use ALB
* Application Loan Balancer simplifies and improves the security of your application, by ensuring that the latest **SSL/TLS** ciphers and protocols are used
* It supports a round-robin load-balancing algorithm
* It also supports a **slow start mode** with the round-robin algorithm that allows you to add new targes without overwhelming them with a flood of requests.

**Listener**

* Define the port and protocol which the load balancer must listen on
* Each application load balancer needs at least one listener to accept traffic
* Each application load balancer can have up 50 listeners
* Routing rules are defined on listeners
* Listener should be port and protocol based, it will listen to the traffic and route the traffic to TG servers

**Target**

* Supports for EC2 instances and ECS containers and IP address
* EC2 instance can be registered with the same target group using multiple ports
* A single target can be registered with multiple target groups

**Target Group**

* + - Logical grouping of targets behind the load balancer
    - Target groups can be existing independently from the load balancer
    - Reginal construct that can be associated with an ASG
    - Target Groups can contain up to 1,000 targets

1. Launch 3 Ec2 instance with different AZ’s and the user data while launching instance it self

Update the user data content as per below for our reference

WebServer -1 🡪 Welcome To WebServer1

WebServer -2 🡪 Welcome To WebServer2

WebServer -3 🡪 Welcome To WebServer3

**UserData bootstrap script**

#!/bin/bash

yum install httpd -y

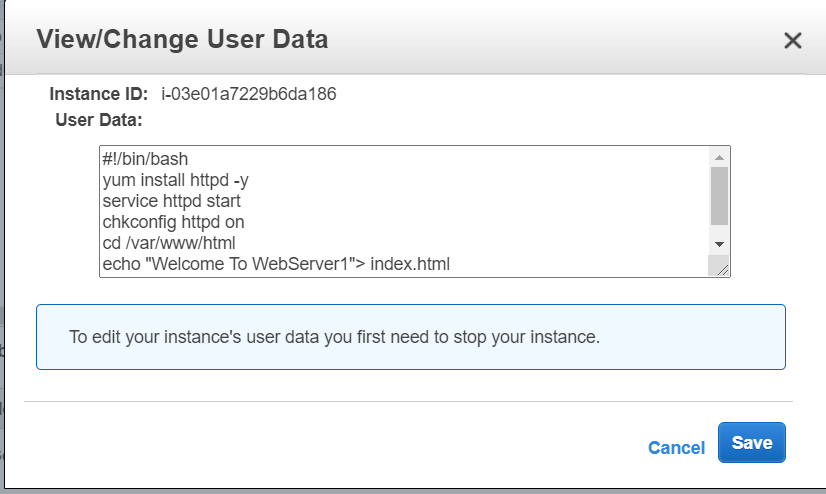
service httpd start

chkconfig httpd on

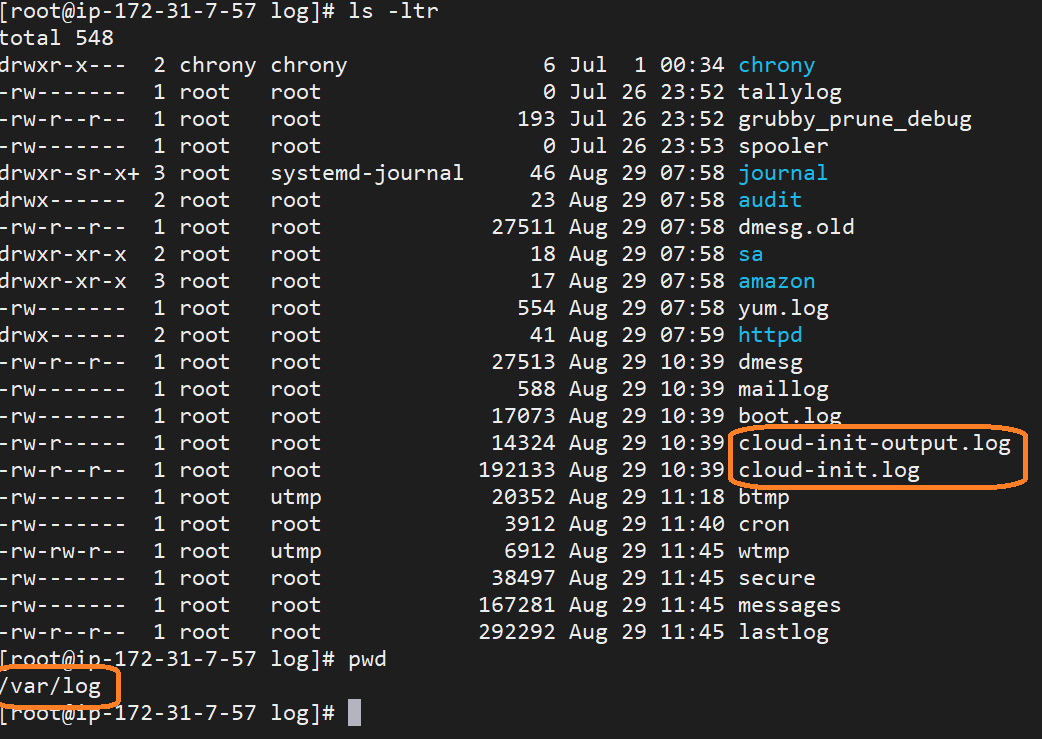
cd /var/www/html

echo "Welcome To WebServer1"> index.html

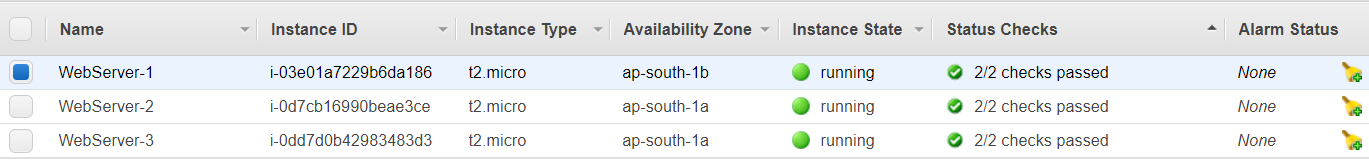
service httpd restart



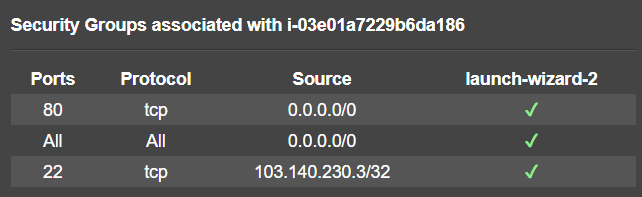
Once instance ready we can check the logs in below path



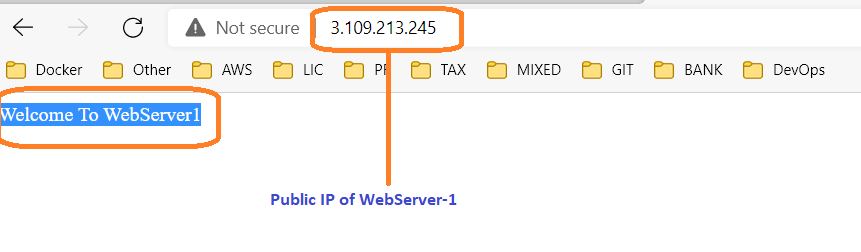
All 3 instance are up and running



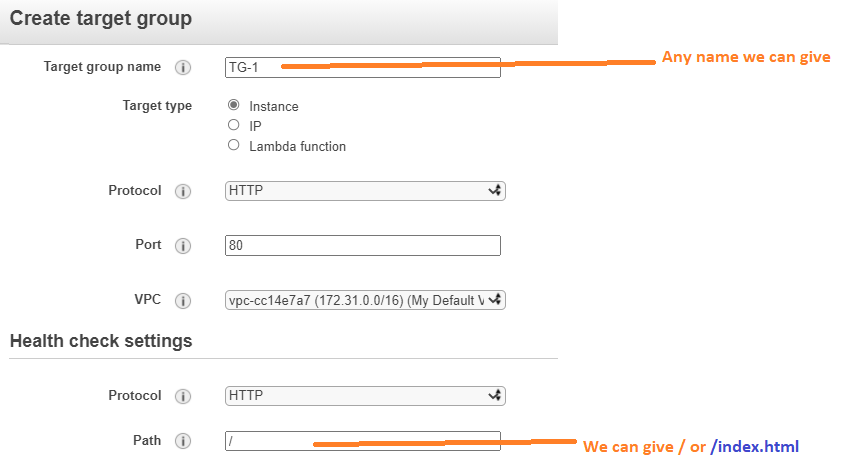
Modify SG by adding HTTP protocol with MYIP and don’t modify default rule (all traffic)

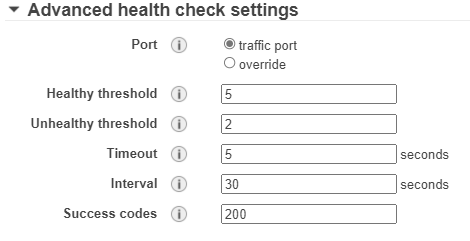


Access the 2 EC2 instances Public IP's on the browser and verify the website is working or not



Create target group

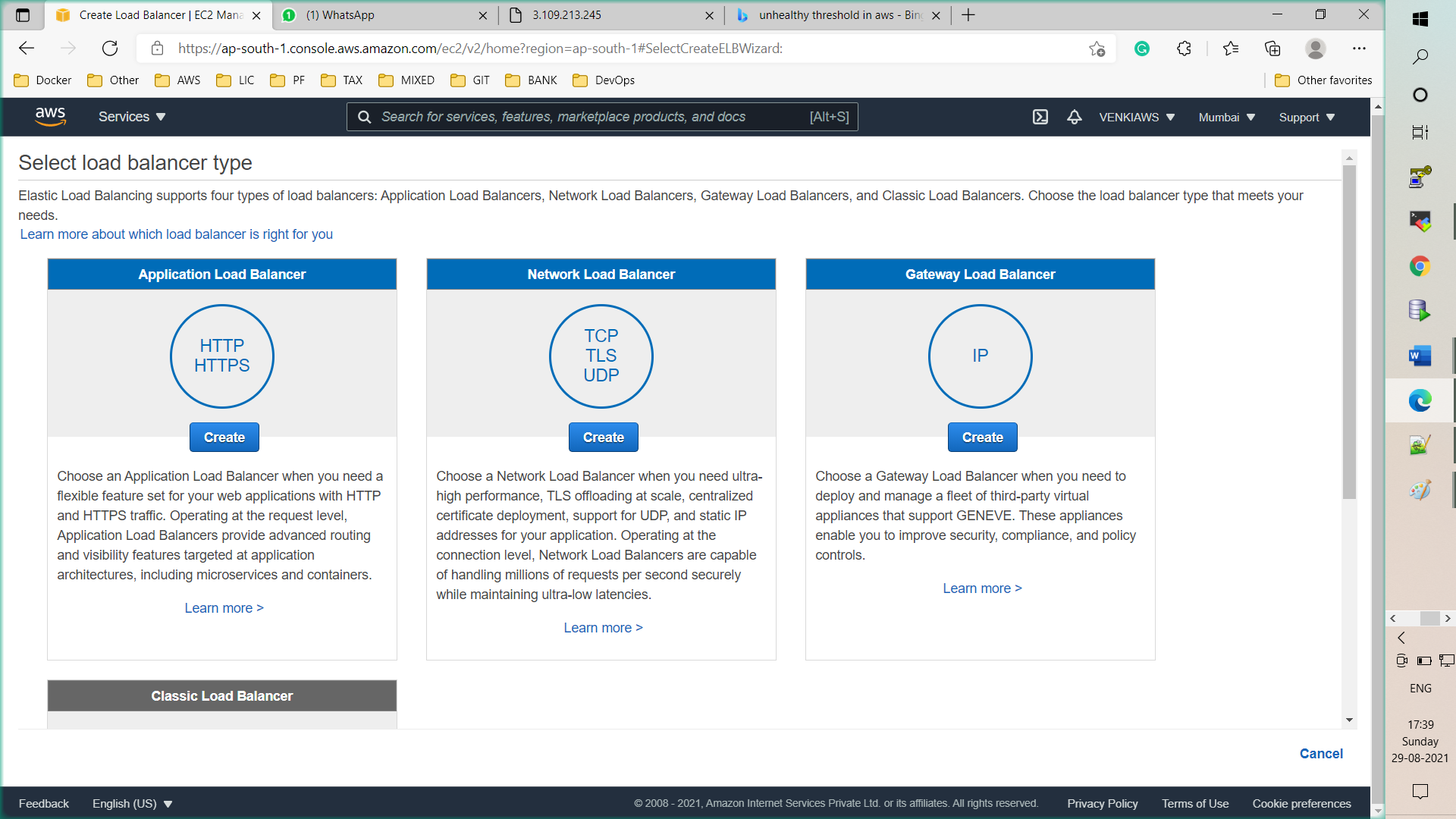




* Healthy threshold : Minimum 5 secs consecutive(continuous) successful response
* Unhealthy threshold : Maximum 3 secs
* Timeout : **Response time out in seconds**
* Interval : how frequently perform health check by default every 30 seconds
* Success codes : **200 is success response code**

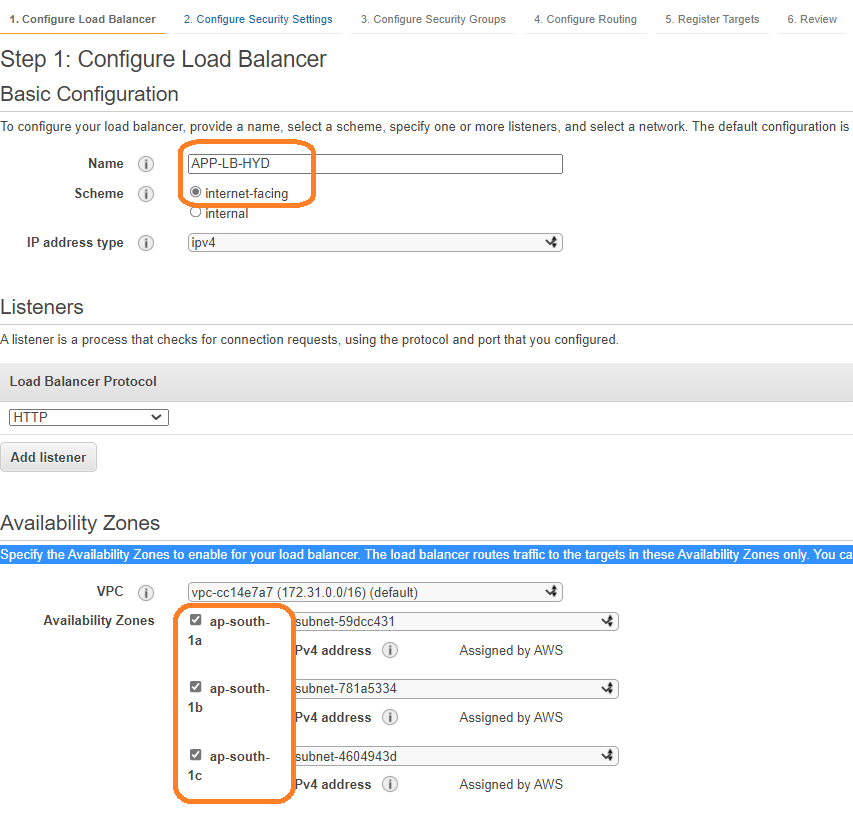
**Create Load Balancer**

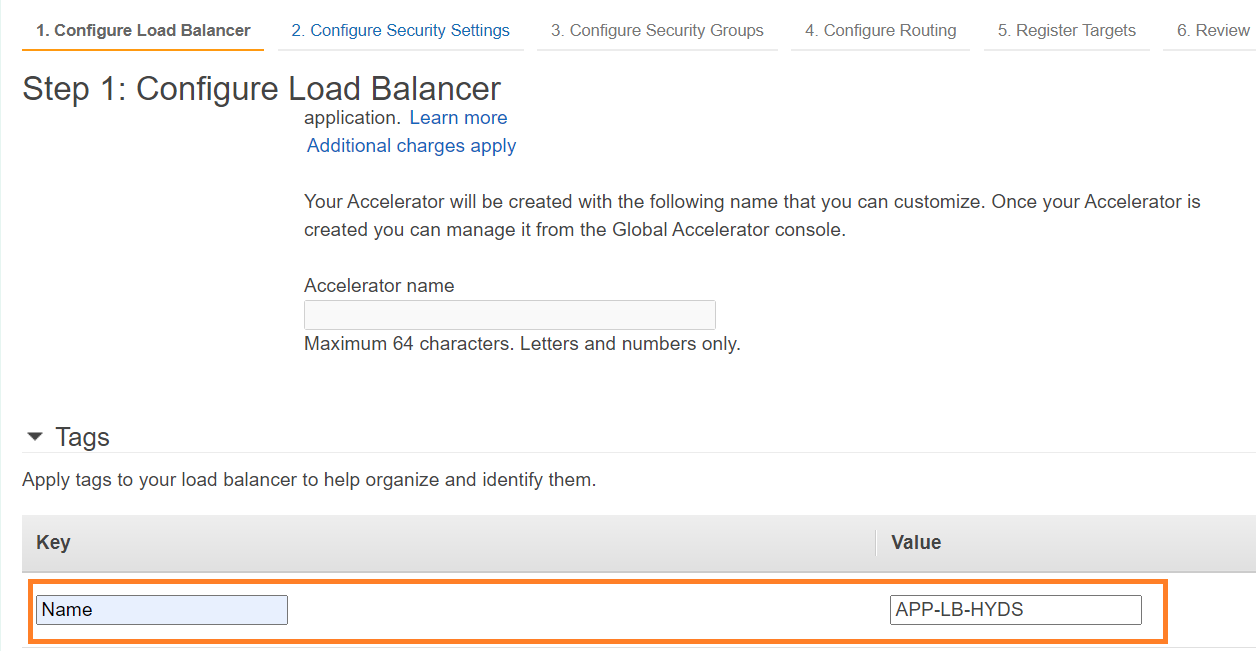
Click on Application Loan Balancer



Step 1: Configure Load Balancer

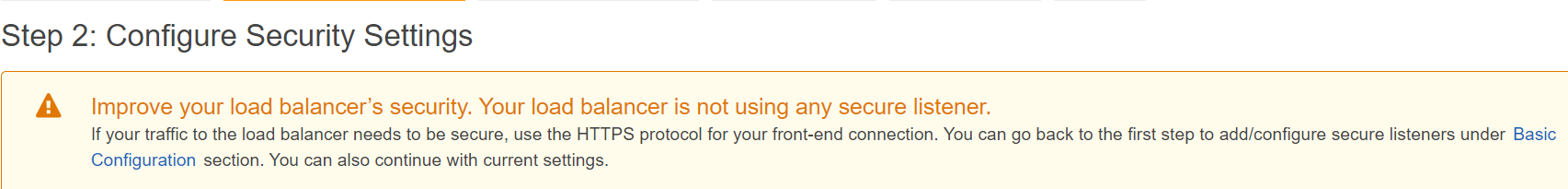
Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase the availability of your load balancer.





Click on **Next Configure Security Settings**

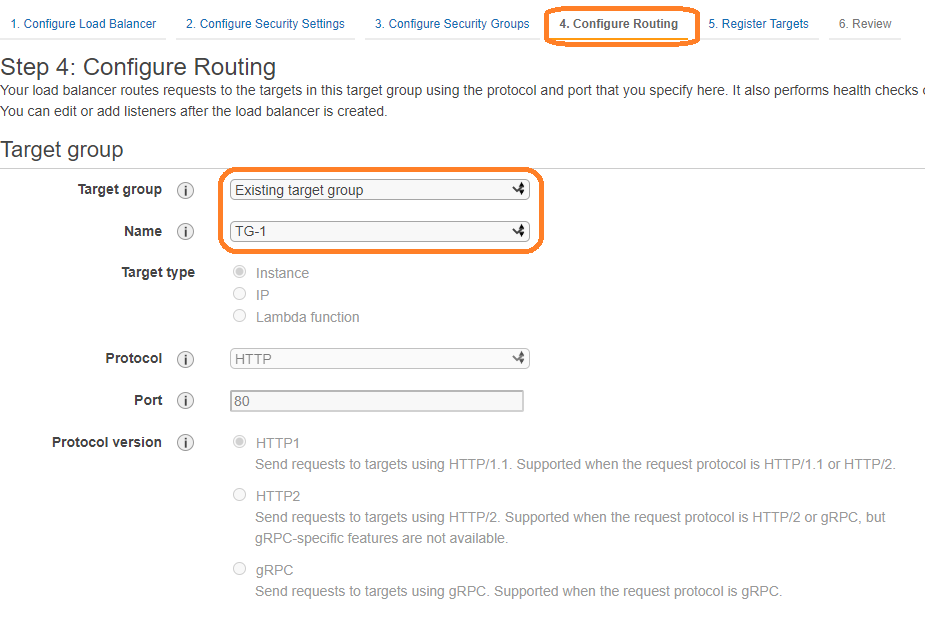
Step 2: Configure Security Settings

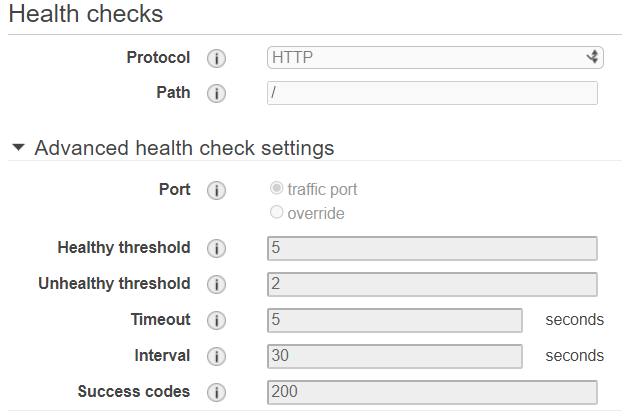


Click on **Next Configure Security Groups**

Step 3: Configure Security Groups

Step 4: Configure Routing





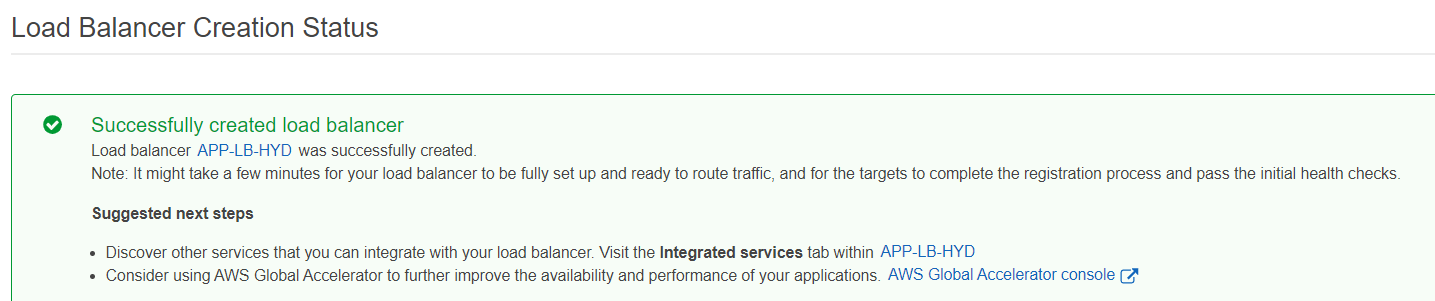
Perform the health on application level

Select the Target Group and Click on **Next Register Targets**

Step 5: Register Targets

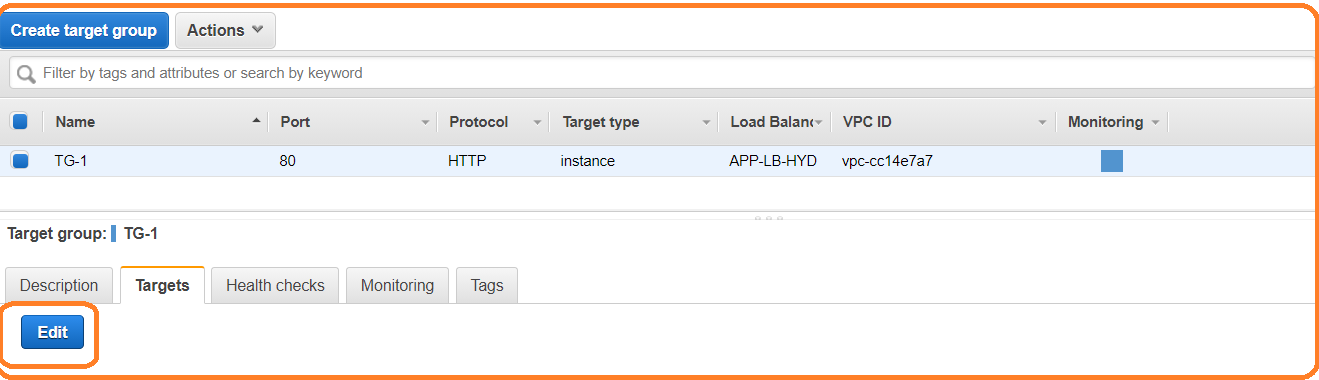
Step 6: Review

Finally

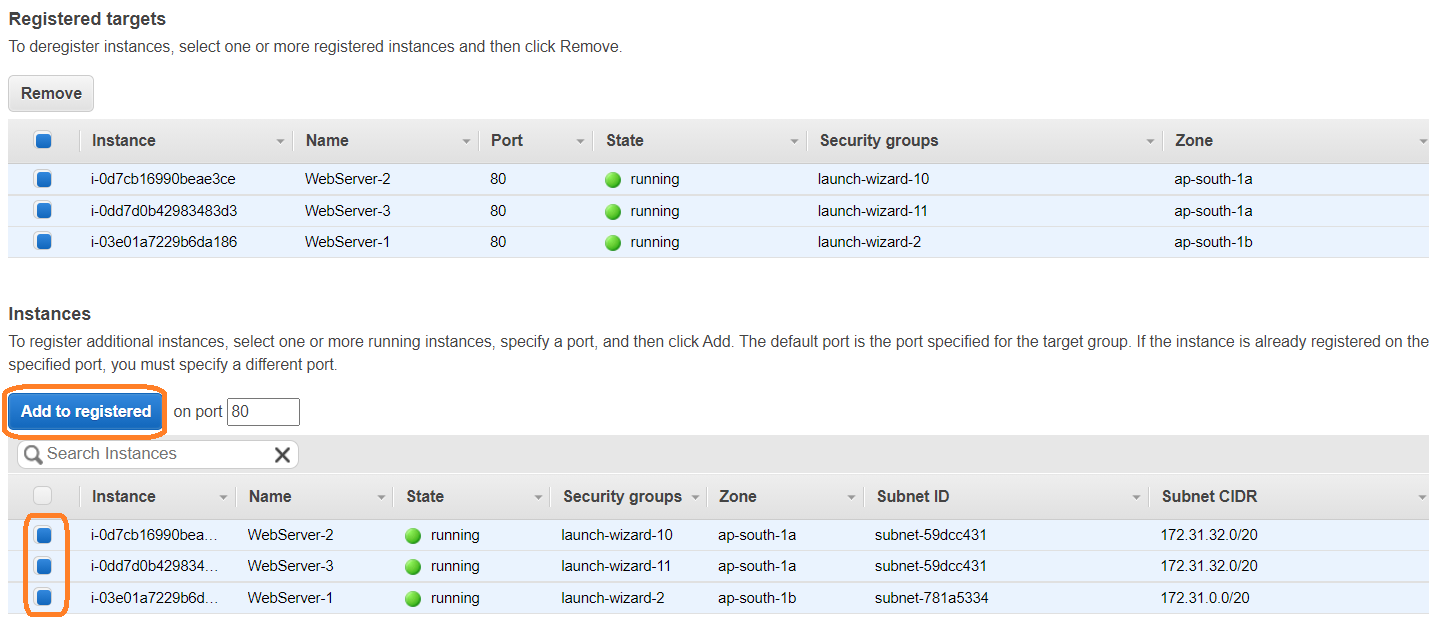


Click on **Target Groups** left panel under Loan Balancing

Register the instance in Target Groups

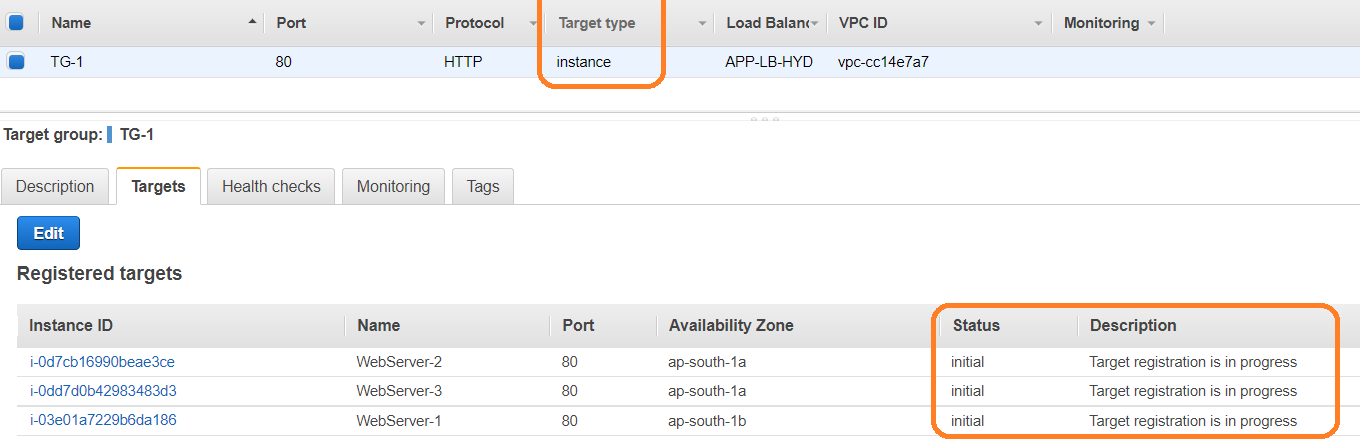


Click on Edit



Click on **Add to registered** and **save**

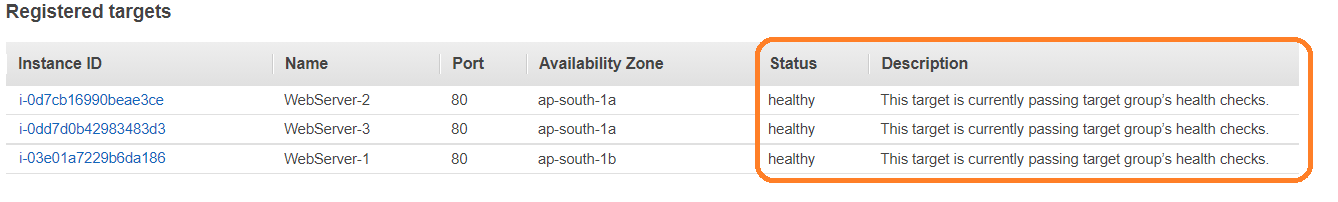
Initially Application status will be **initial**



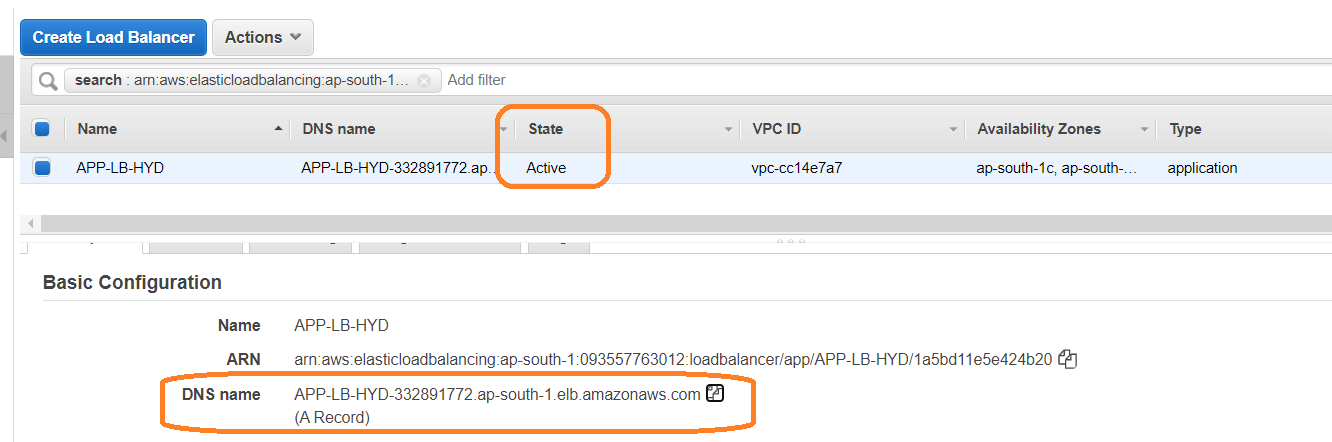
Later it will change to **health** or **unhealth** based on health check

What ever status is unhealthy load balancer will not send the request

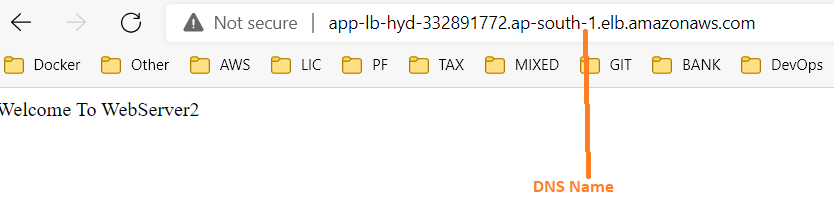
What ever status is healthy laod balancer will sent the request



Go to Loan Balancer and verify state



Copy DNS name and verity status in browser



**Host Based Routing**

JavaWebapp.techinfo.co.in/java-web-app

JavaWebapp.techinfo.co.in/maven-web-app

**Path Based Routing**

techinfo.co.in/java-web-app

techinfo.co.in/maven-web-app

**Refresh and verify content**

**Path based routing steps**

#!/bin/bash

yum install httpd -y

service httpd start

chkconfig httpd on

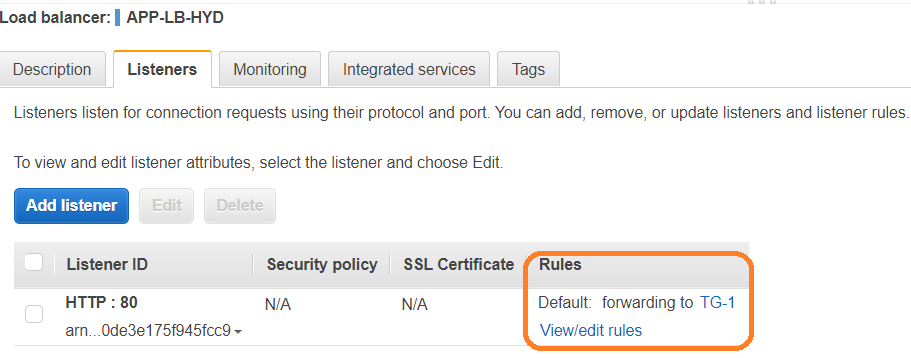
cd /var/www/html

echo "**Welcome To WebServer3**"> **admin**

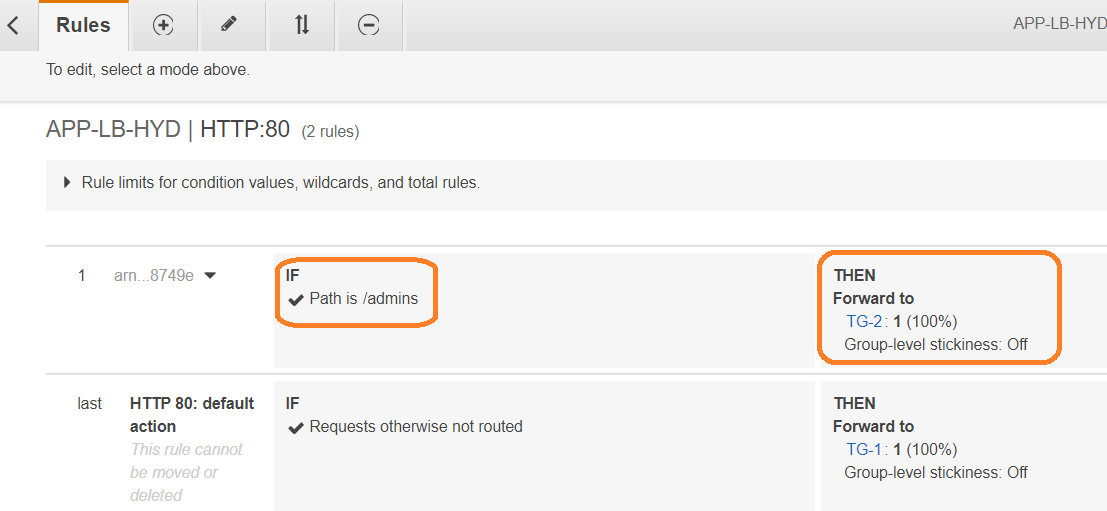
service httpd restart

Create one more target group and register WebServer3 instance

Click on **Loan Balancer 🡪 Listeners 🡪 Click on View/edit rules**



Click on (+) and click on insert rules 🡪 **path(drop down)[left side]** and right side select **Forward to <Target Group name>**



Click on Load Balancer 🡪 Description 🡪 DNS name

