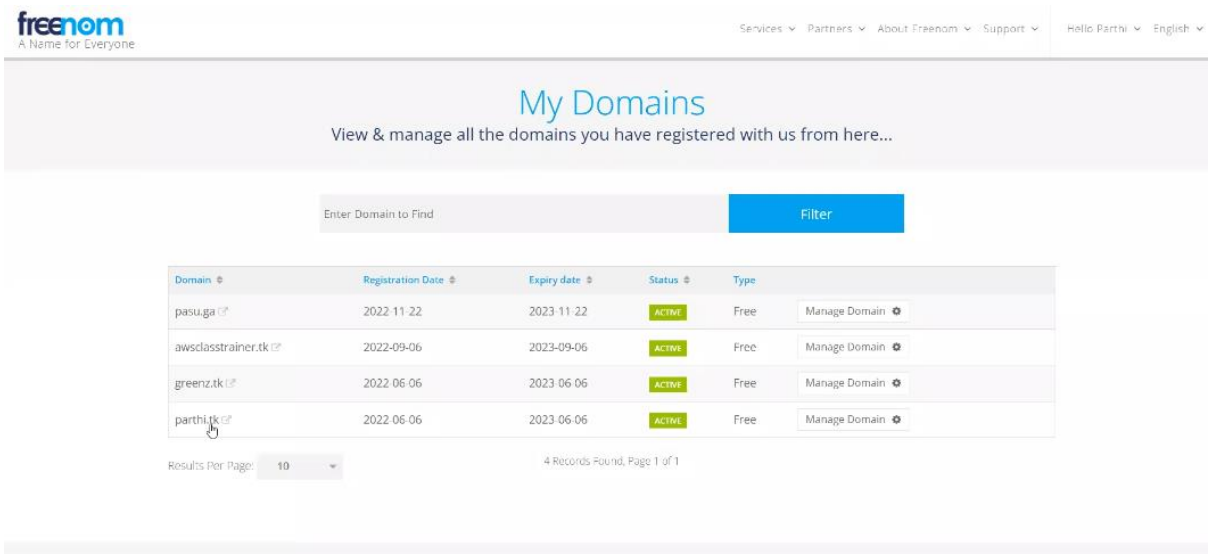
ROUTE 53 (global specific)

DNS-- Domain name server -- name provider

Route53—network engineering

(*) purchase a domain

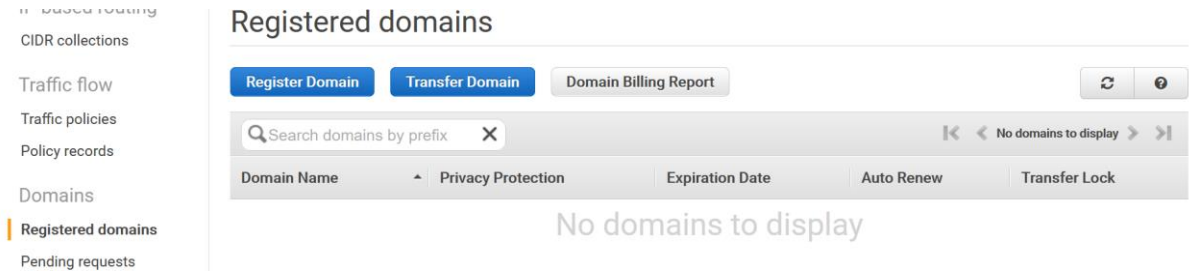


(*) login to AWS Account

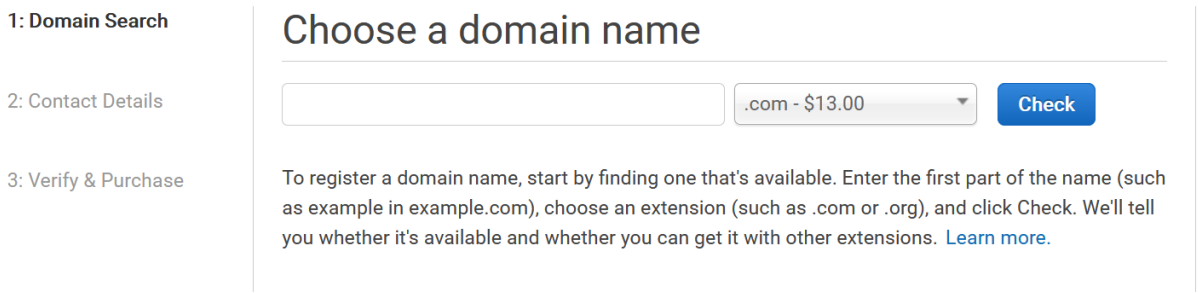
(*) search R53 – route 53 dashboard appears

(*) route 53 is a global specific service you cannot select any regions

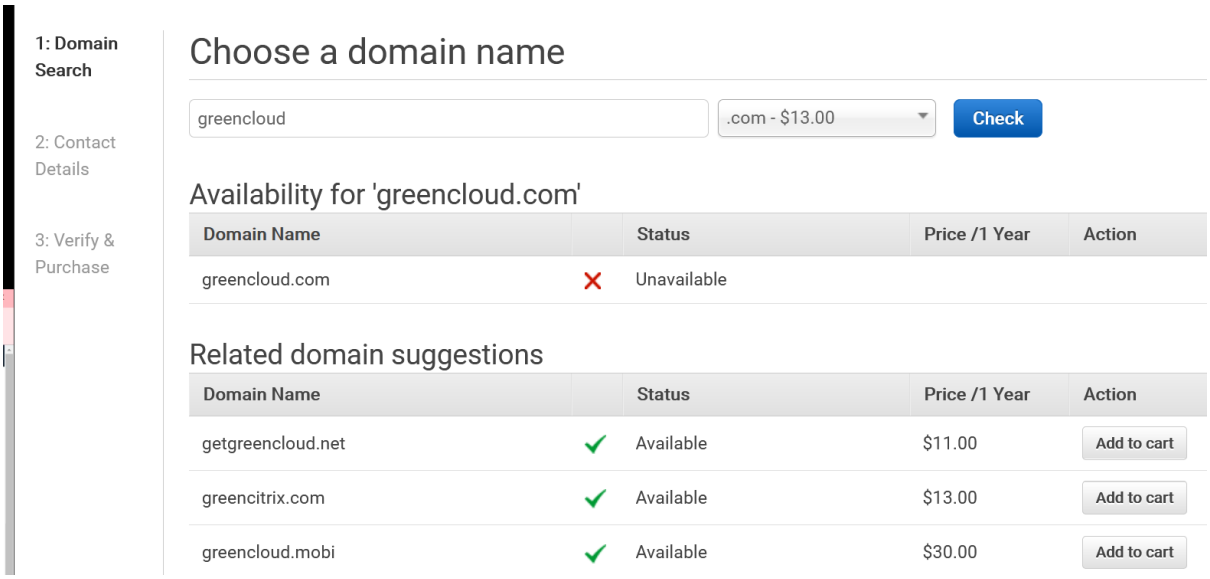
(*) go to domains -- registered domains



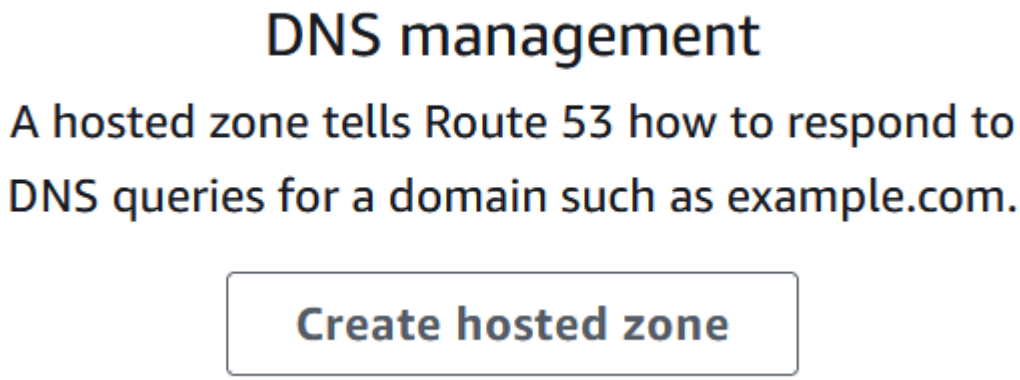
(*) press registered domain



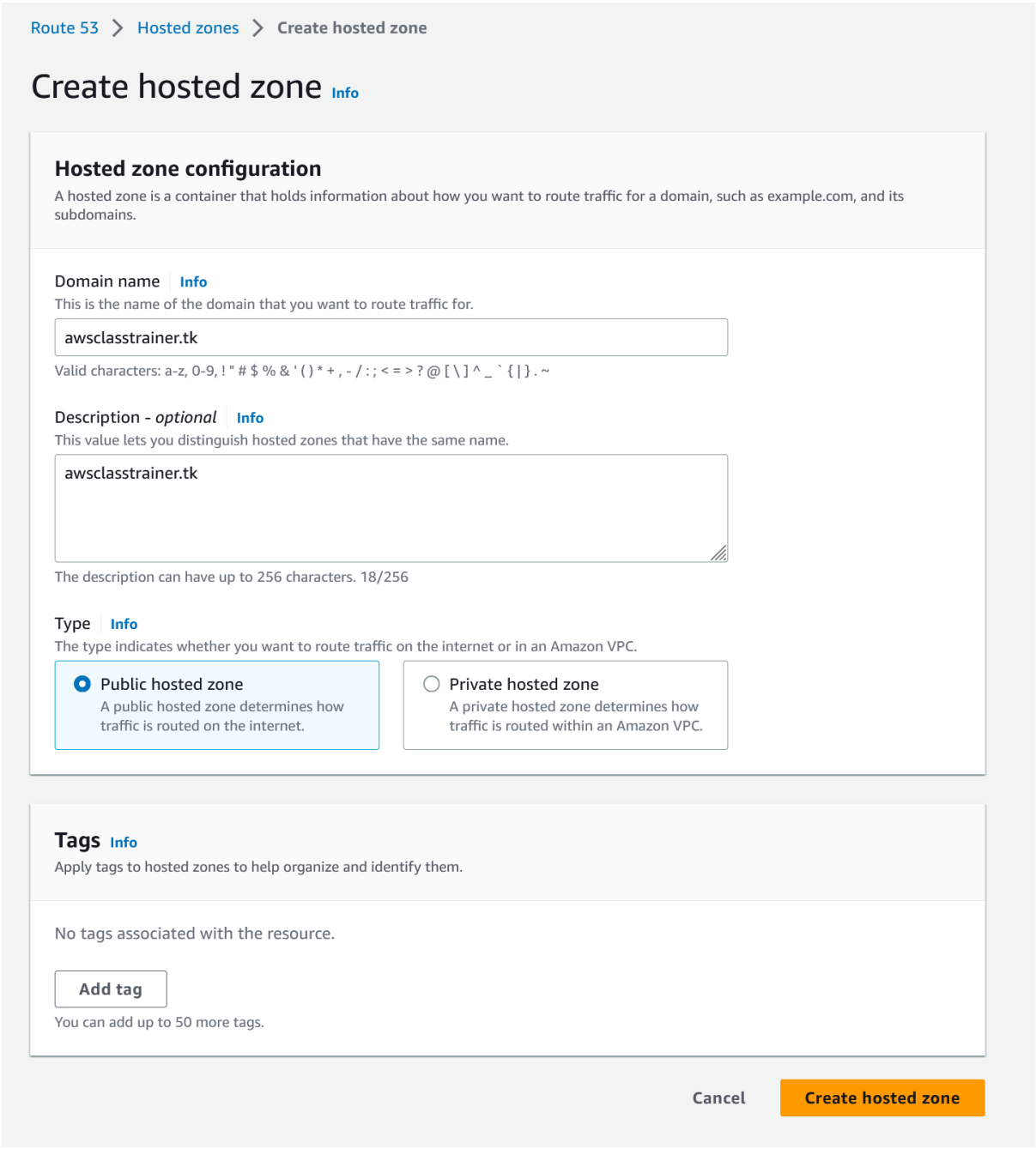
(*) availability of domain



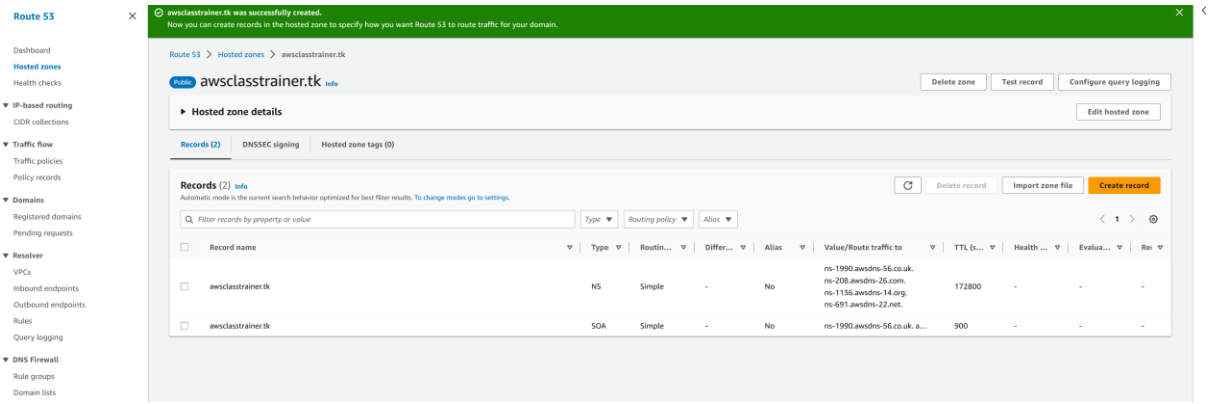
(*) to migrate a provider to cloud we need to create a hosted zone



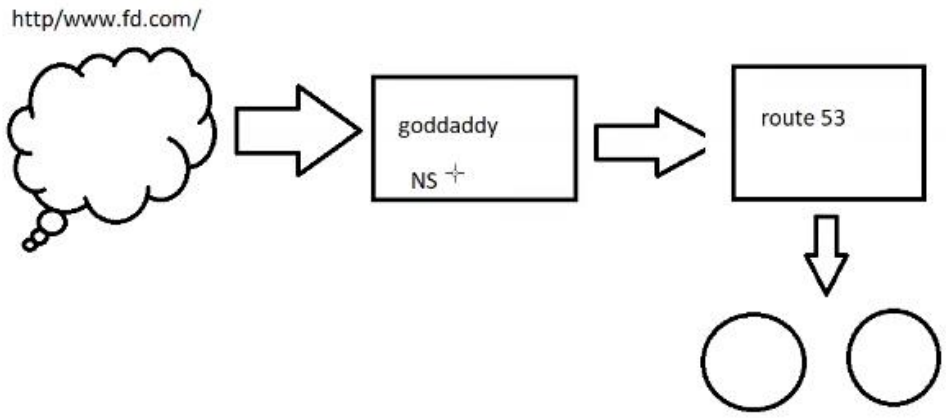
(*) create hosted zone – public hosted zone



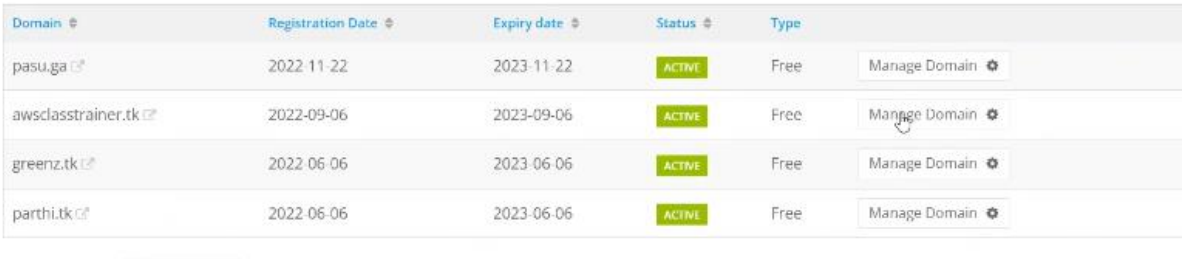
(*) integrated my domain with route 53 – route 53 provides 4 name server



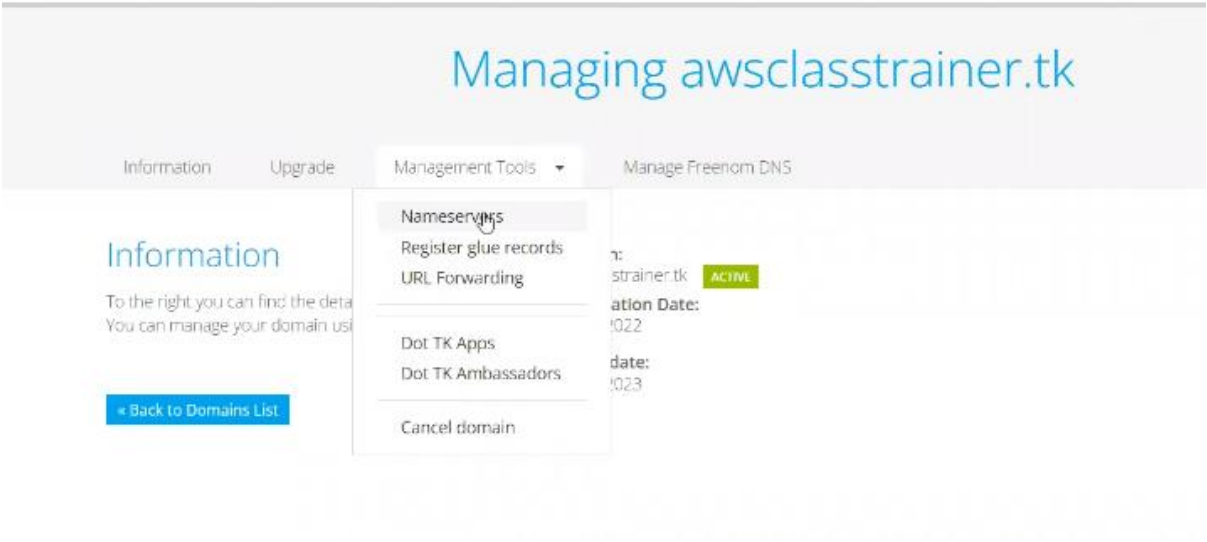
(*) we need to register these 4-name service with go daddy or freenom this is called 2-way handshake then the traffic gets routed



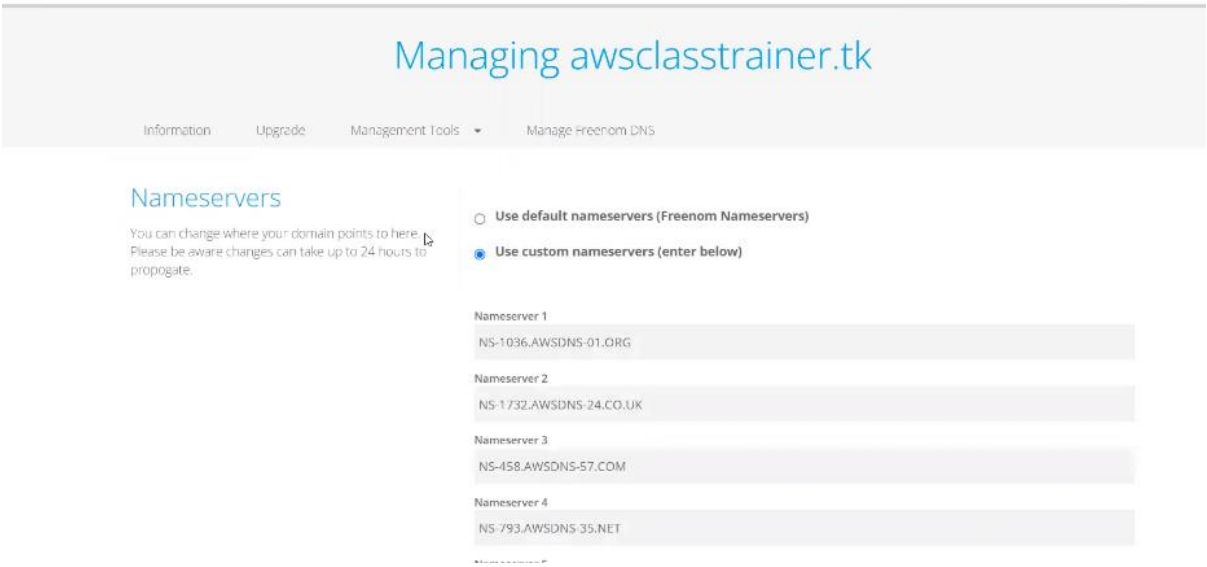
(*) then go to freenom and press manage domain



(*) then go to management tools --- name servers



(*) we need to delete the name server in freenom and we need to add the name servers from route 53 with no dots



(*) start of authority SOA for high availability route 53 provides 4 name servers

(*) check the name servers again in freenom

(*) global declaration of name takes 48 hrs.

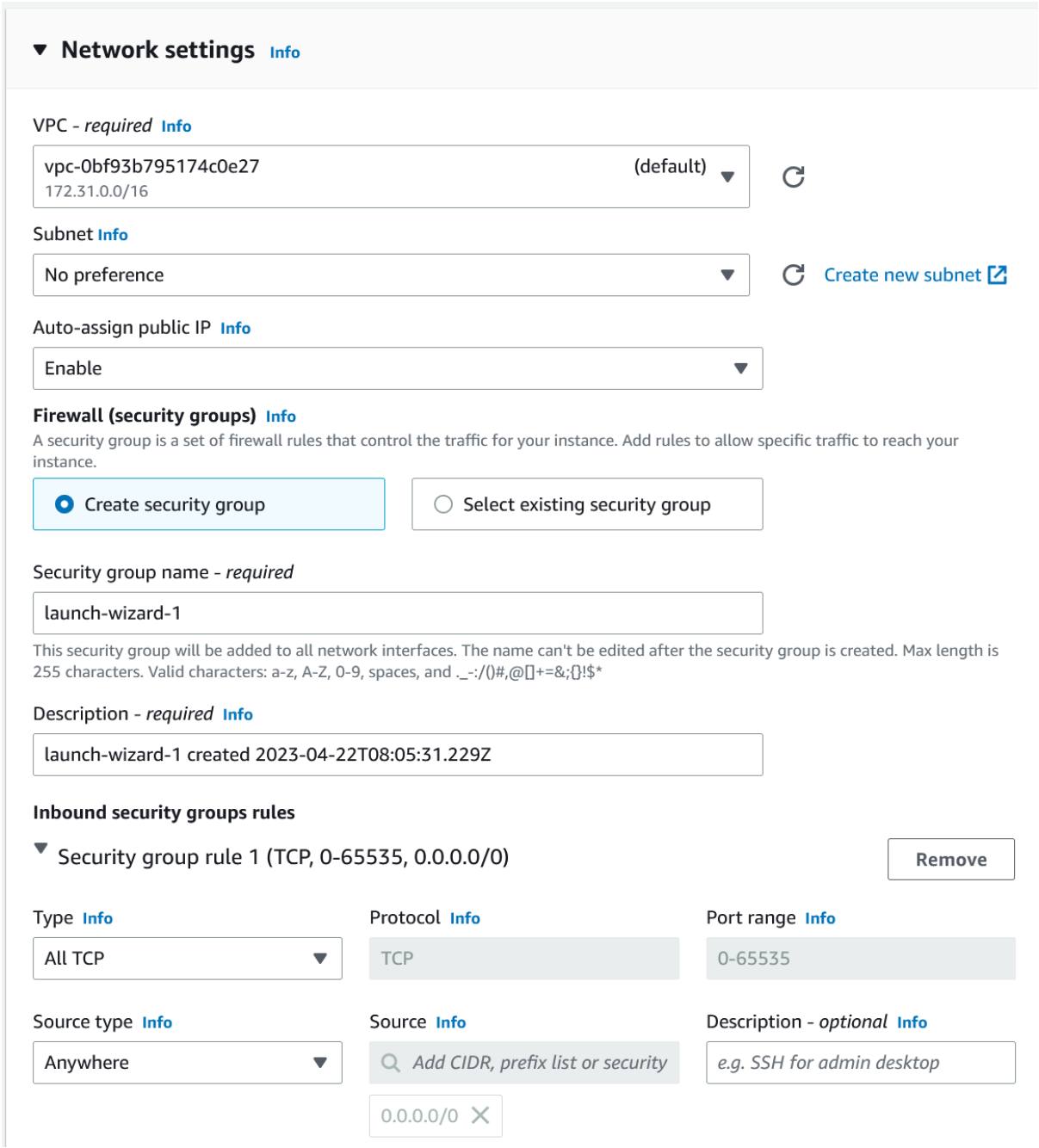
(*) next EC2 server creation

(*) Go to EC2 Dashboard go to instances

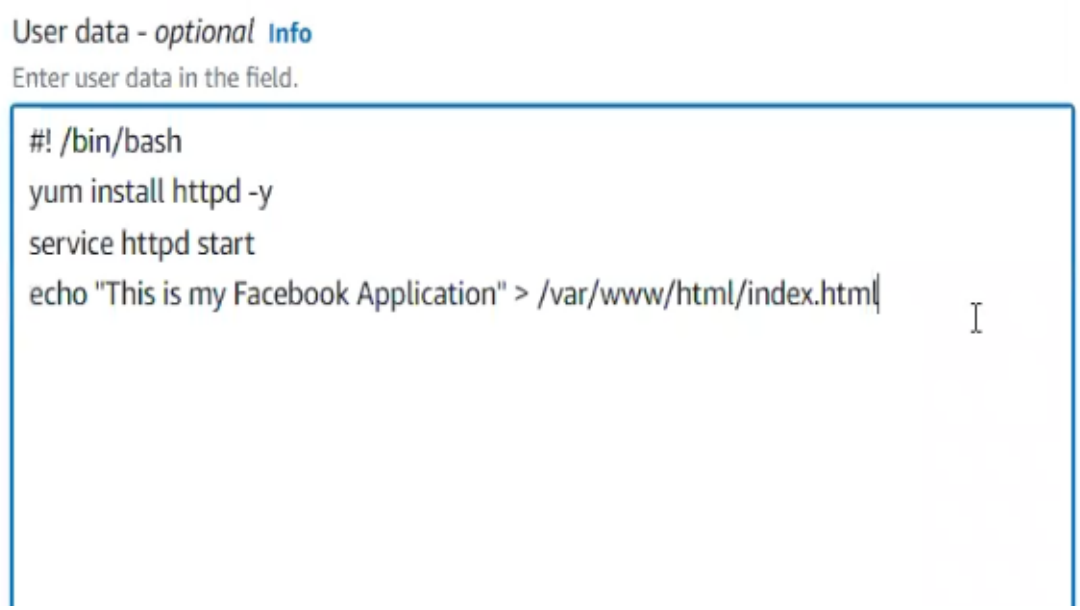
(*) then create servers in 2 different regions first Mumbai

(*) create Linux servers

(*) In network settings – edit -- inbound security group rules – select all TCP



(*) edit advance detail USER DATA options



(*) then create instance Mumbai instance created

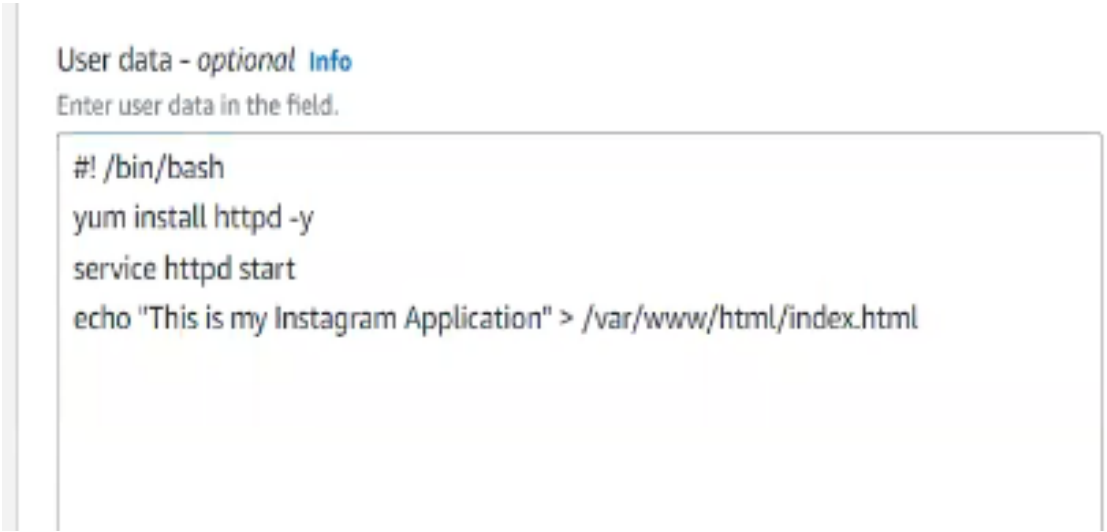
(*) paste the Mumbai machine Ip in browser the output is created this is my Facebook application

(*) then go to Singapore create instance

(*) create Linux machine

(*) In network settings – edit -- inbound security group rules – select all TCP

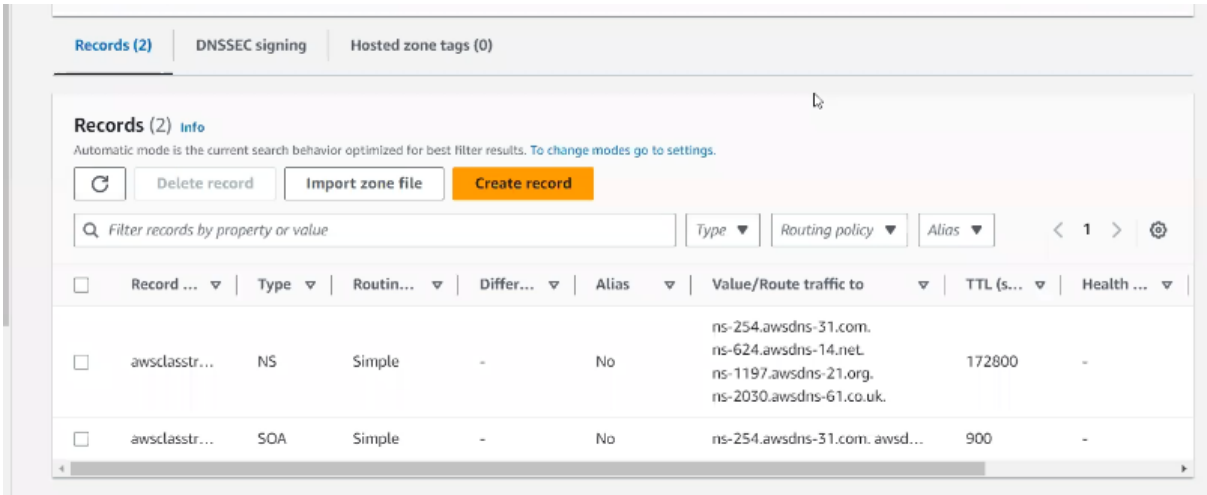
(*) edit advance detail USER DATA options



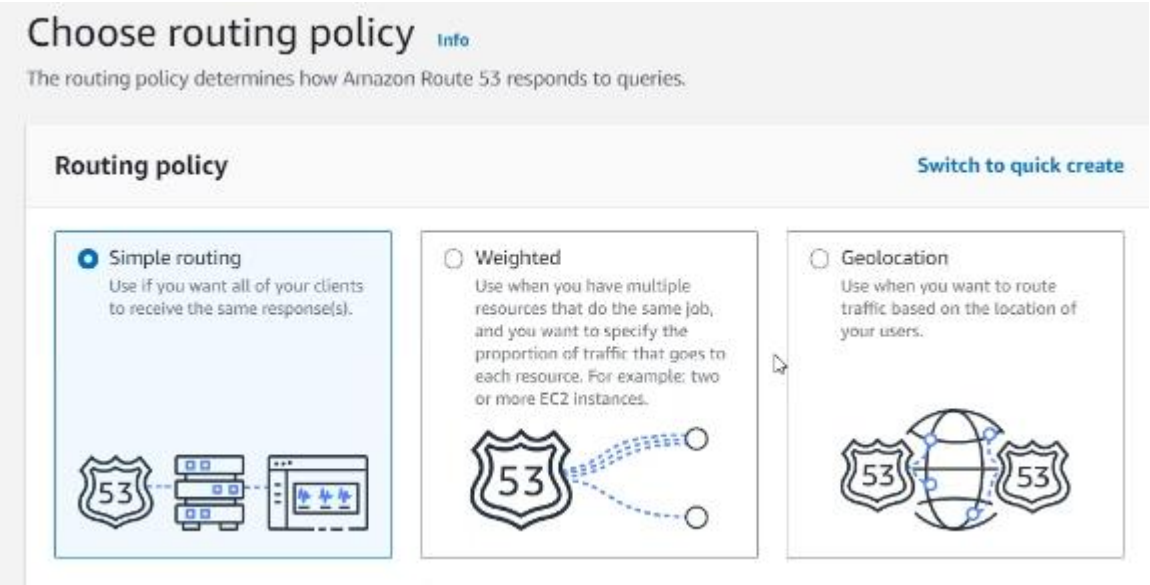
(*) Singapore instance created

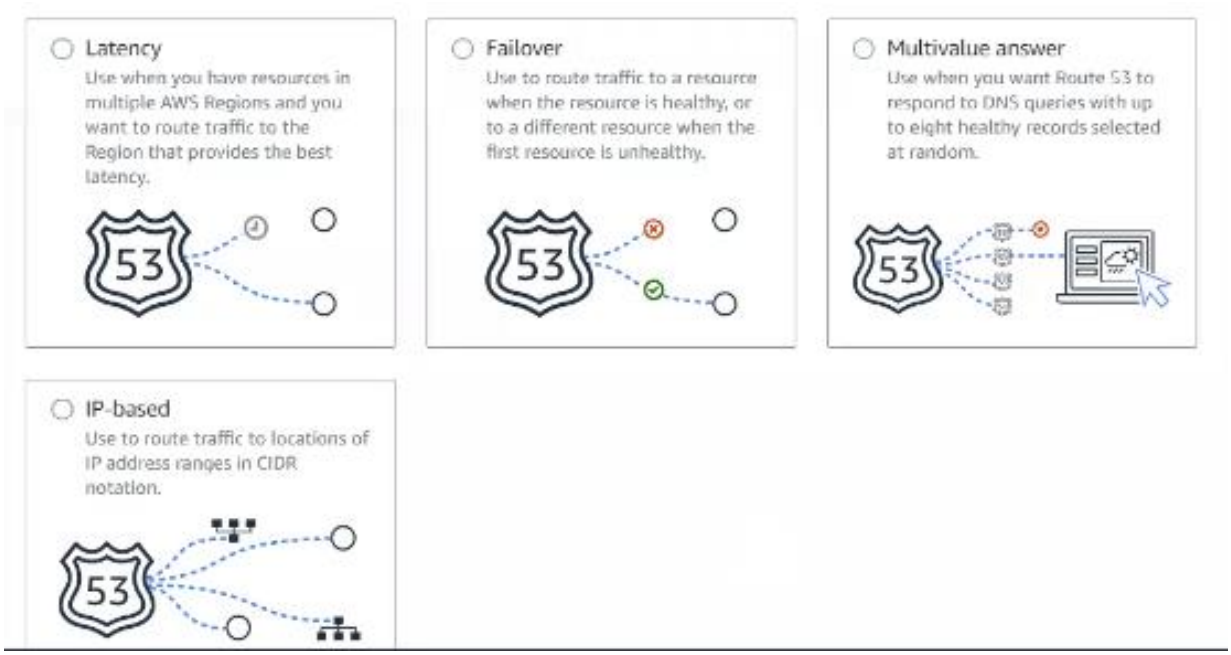
(*) paste the Singapore machine Ip in browser the output is created this is my Instagram application

(*) create record



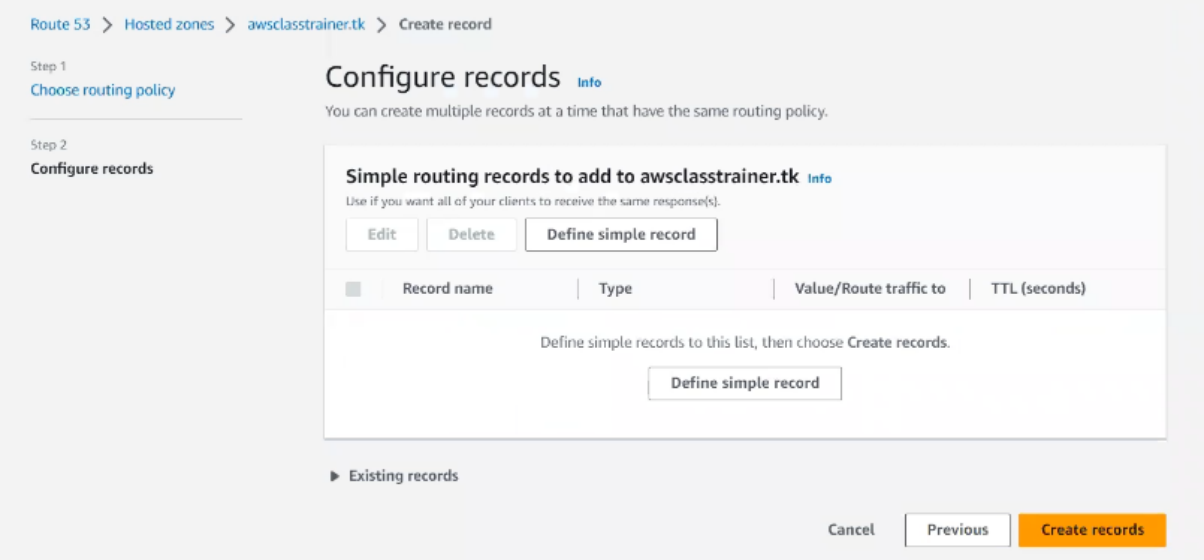
(*) choose routing policy 7 policies



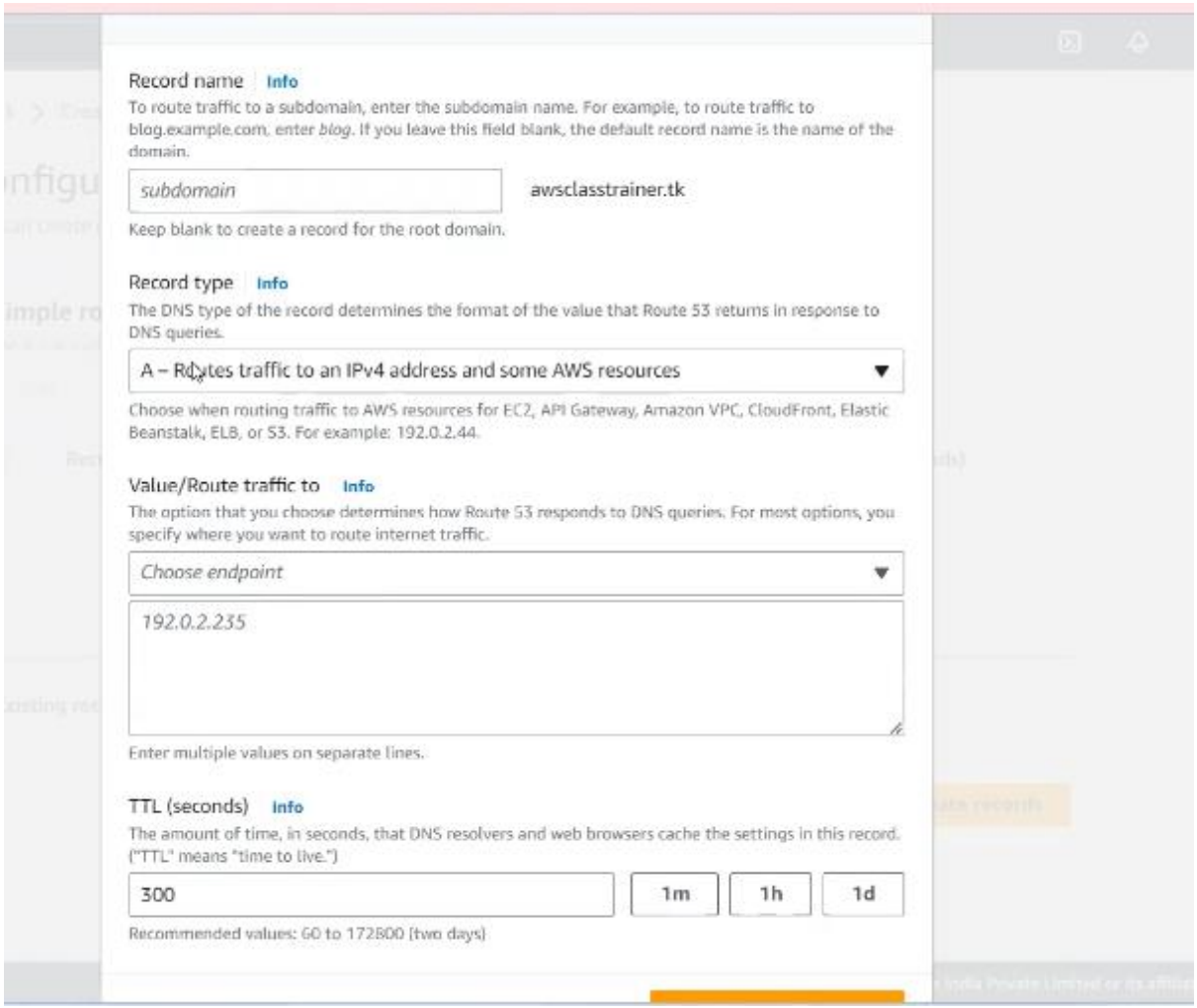


(*) we are using only 5 routing policies simple routing, weighted, geolocation, latency, failover

(*) we are taking simple routing policy --- next

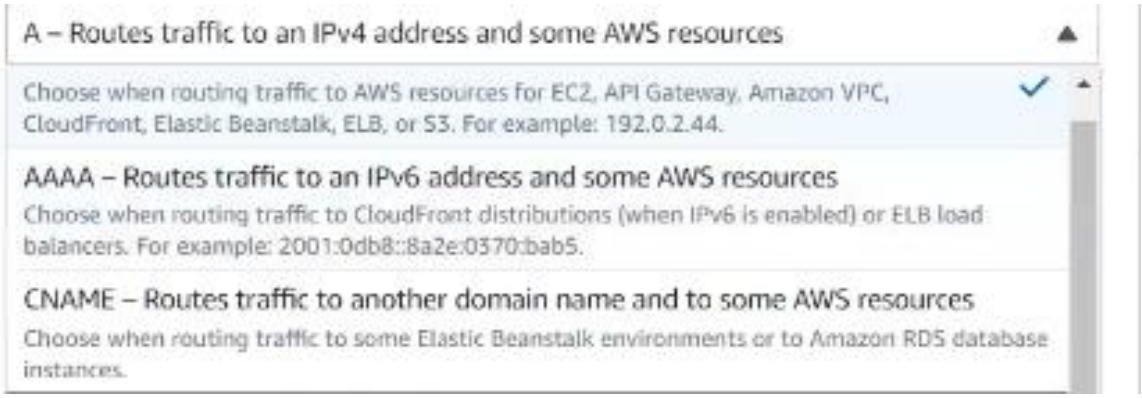


(*) define simple record

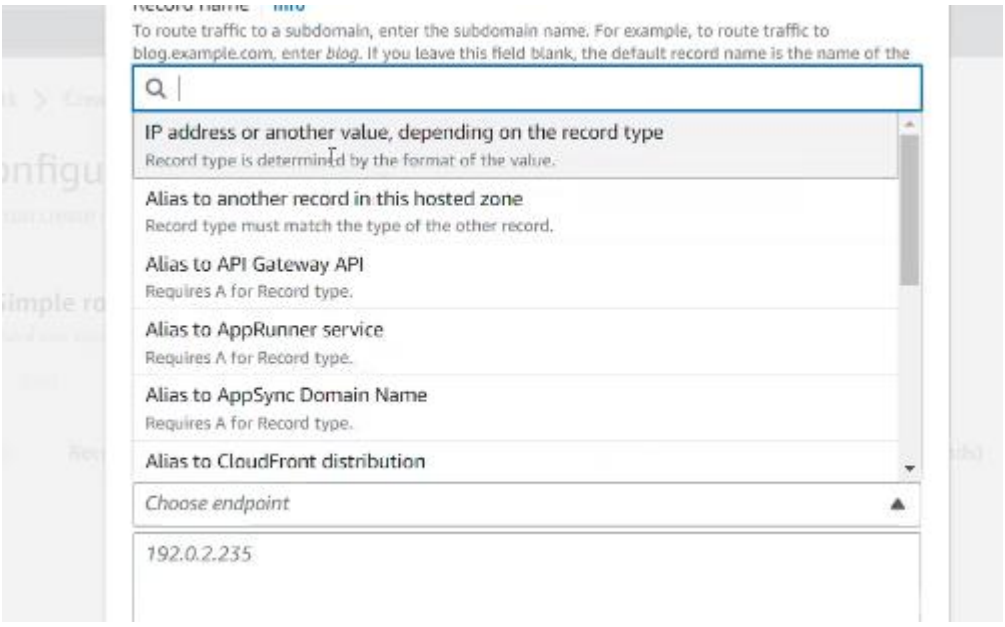


(*) record type

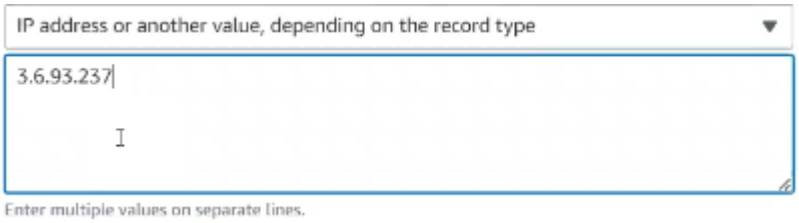
We are using 3 record types A, AAAA, CNAME



(*) ENDPOINT Ip address or another value depending on record type



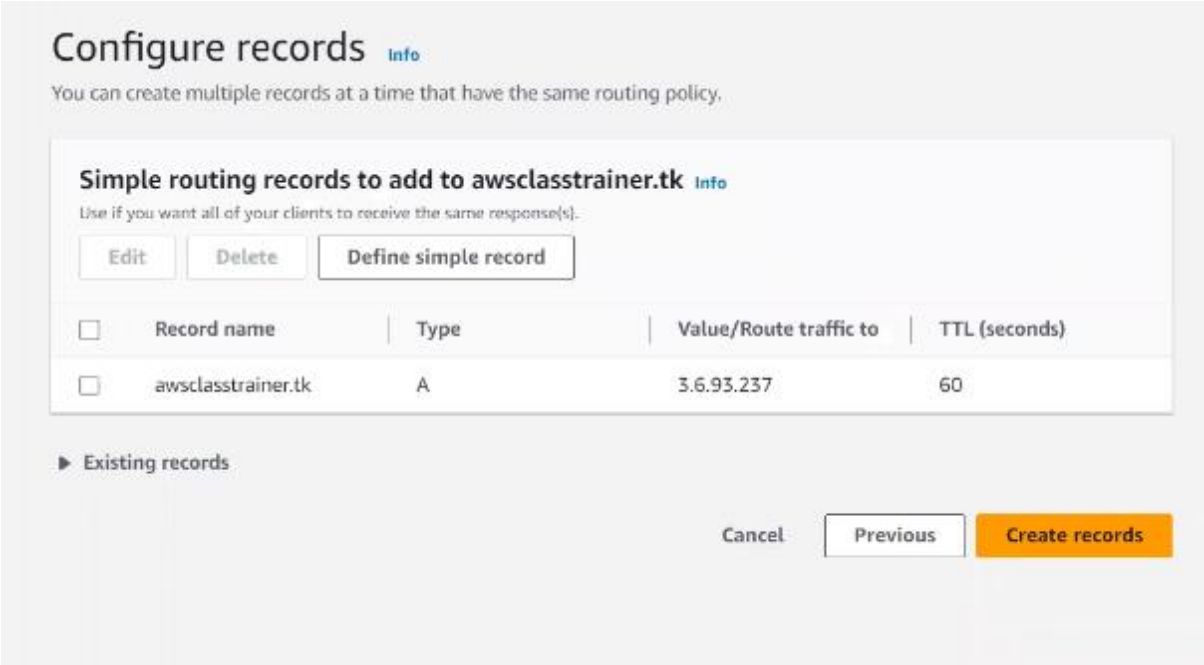
(*) copy paste Mumbai machine ip address



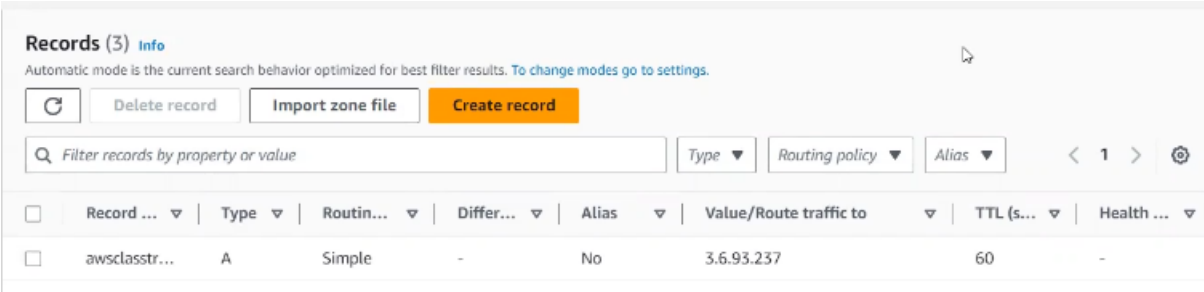
(*) TTL SECOND TO 1M

(*) PRESS DEFINE SIMPLE RECORD

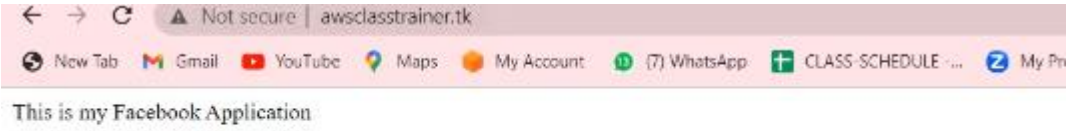
(*) create records



(*) record created



(*) this is Facebook application



GEOLOCATION POLICY

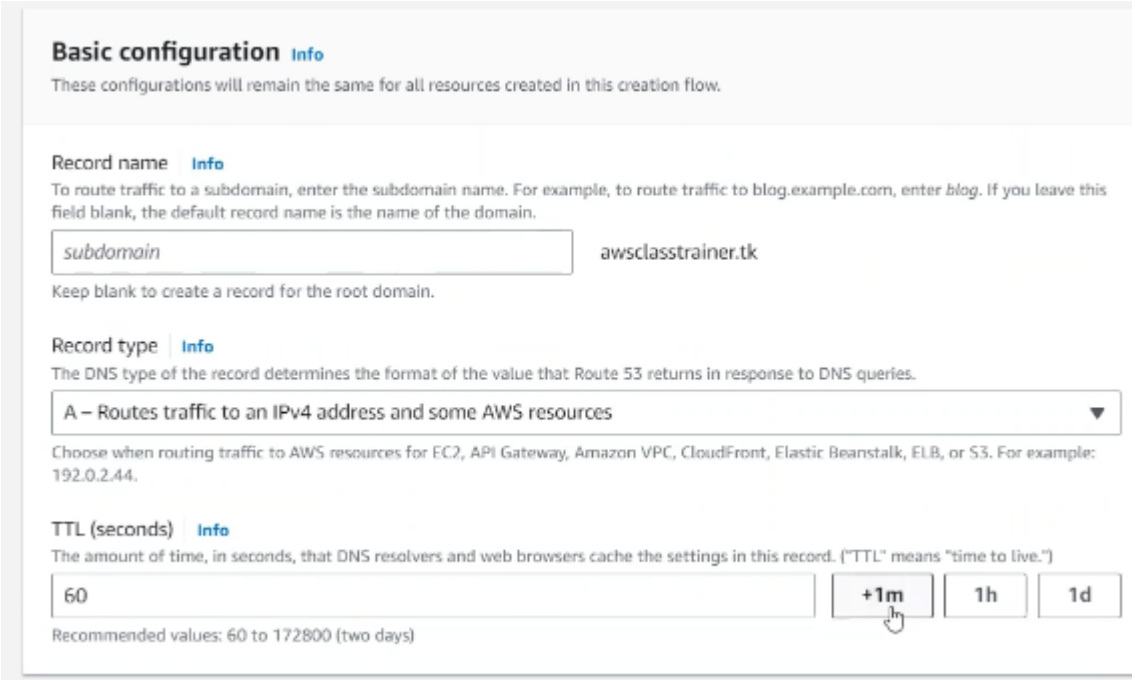
INDIA to India

Singapore to Singapore

Route users based on location accordingly

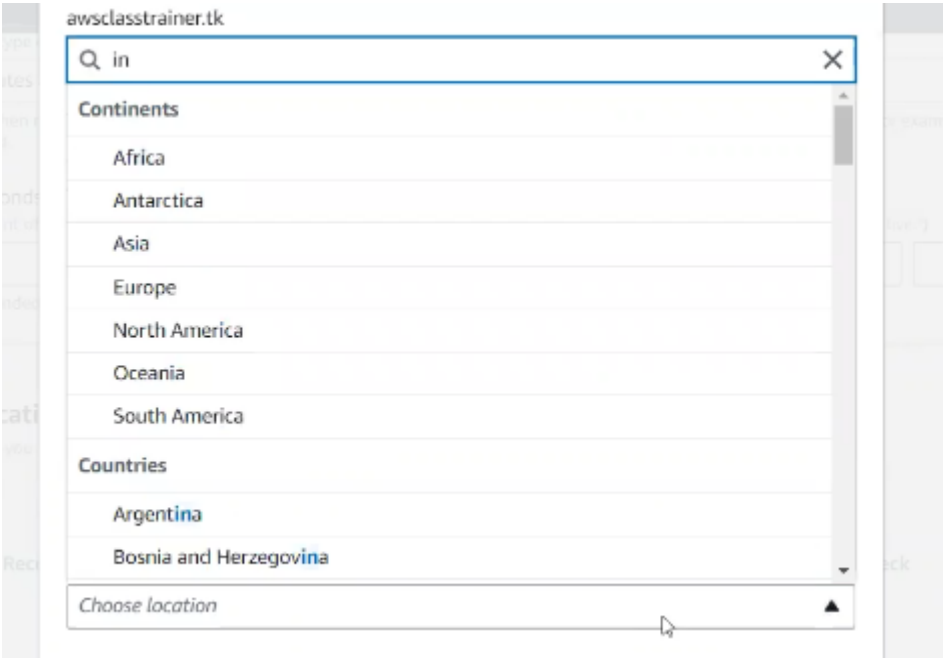
(*) select geolocation policy – next

(*) A type record

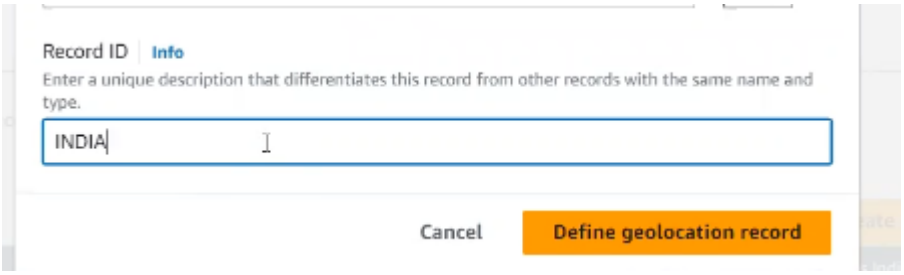


(*) first Mumbai machine ip address

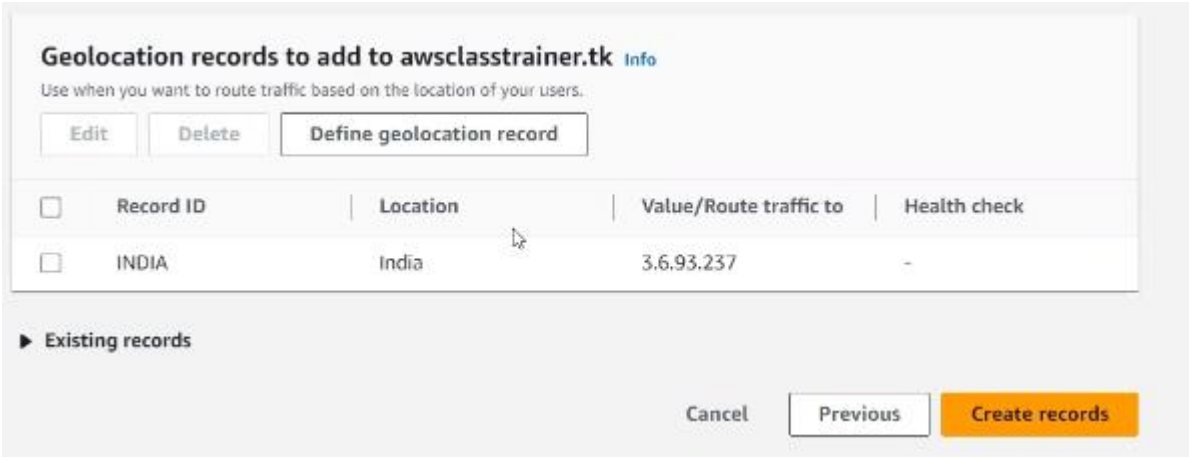
(*) select location India



(*) record id India



(*) SELECT DEFINE GEOLOCATION RECORD

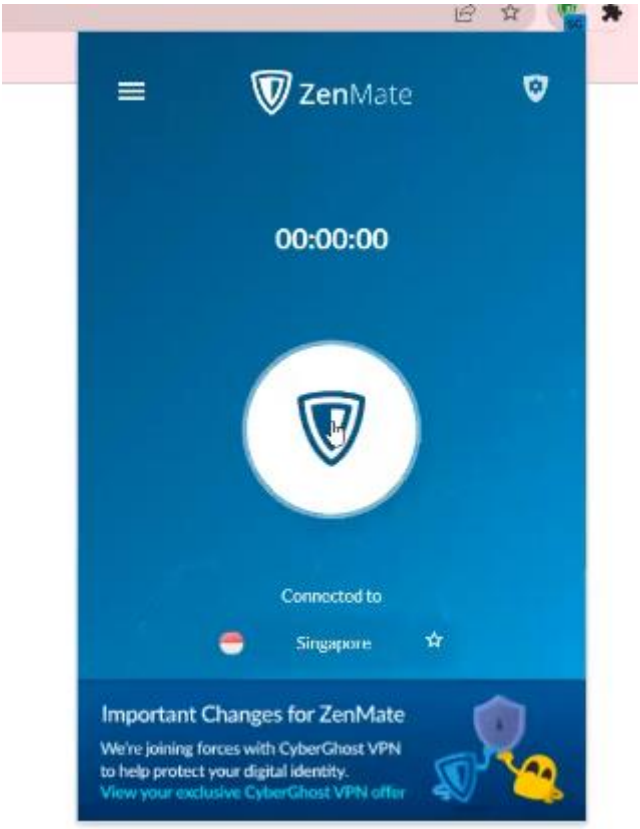


(*) Geo location record created

(*) follow this SAME procedure for Singapore machine

Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to	TTL (s...	Health ...
awsclasstr...	A	Geolocation	India	No	3.6.93.237	60	-
awsclasstr...	A	Geolocation	Singapore	No	54.169.67.38	60	-

(*) change Singapore location using vpn in browser

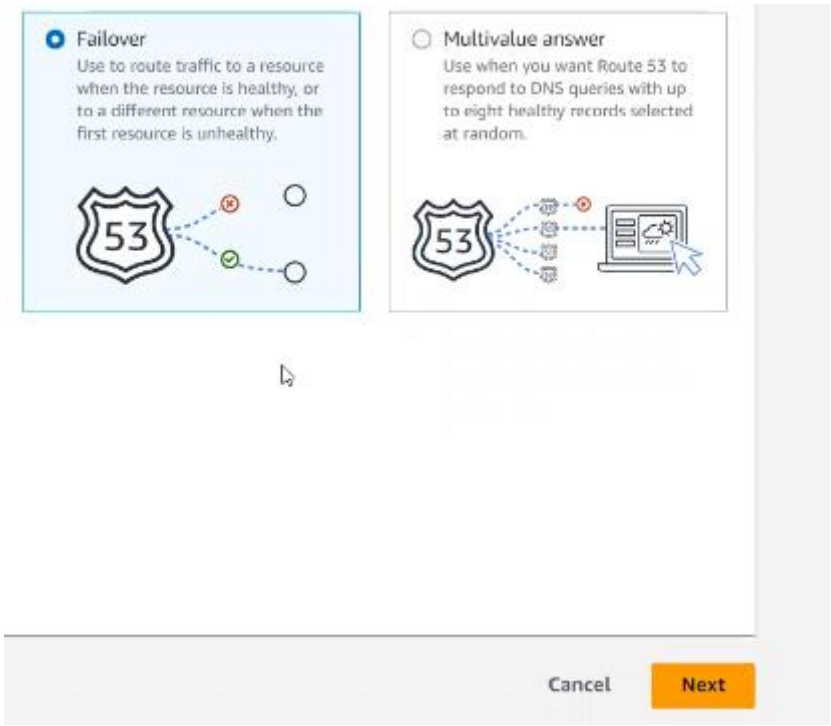


(*) output will be this is my Instagram application

FAILOVER POLICY

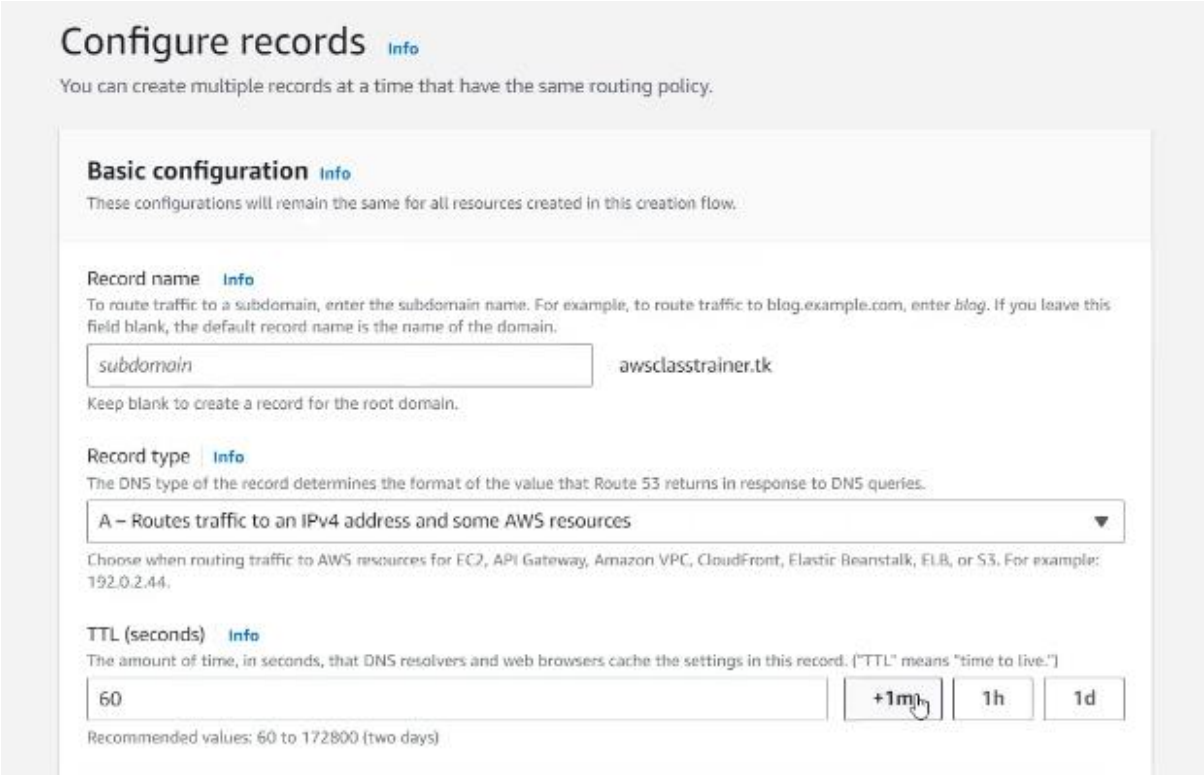
Disaster recover setup

main machine and DR machine if main machine fails DR machine starts to route traffic

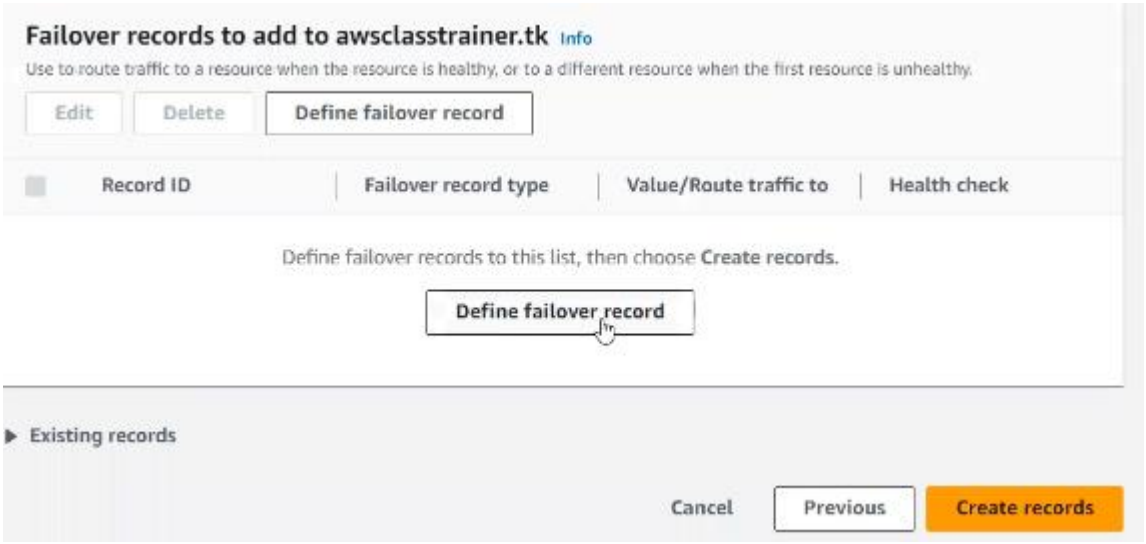


(*) CHOOSE failover policy – next

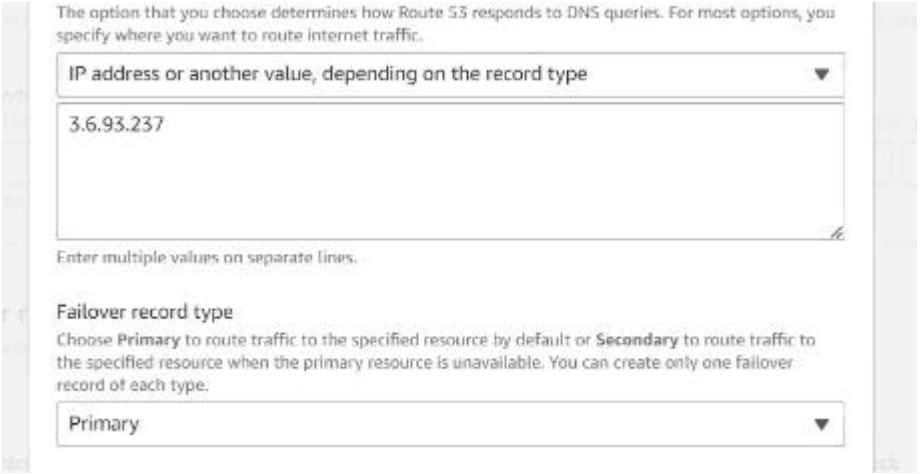
(*) configure records record type a TTL seconds 1m



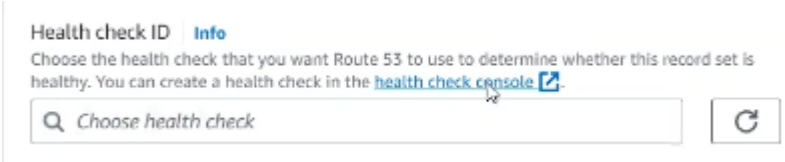
(*) define failover records



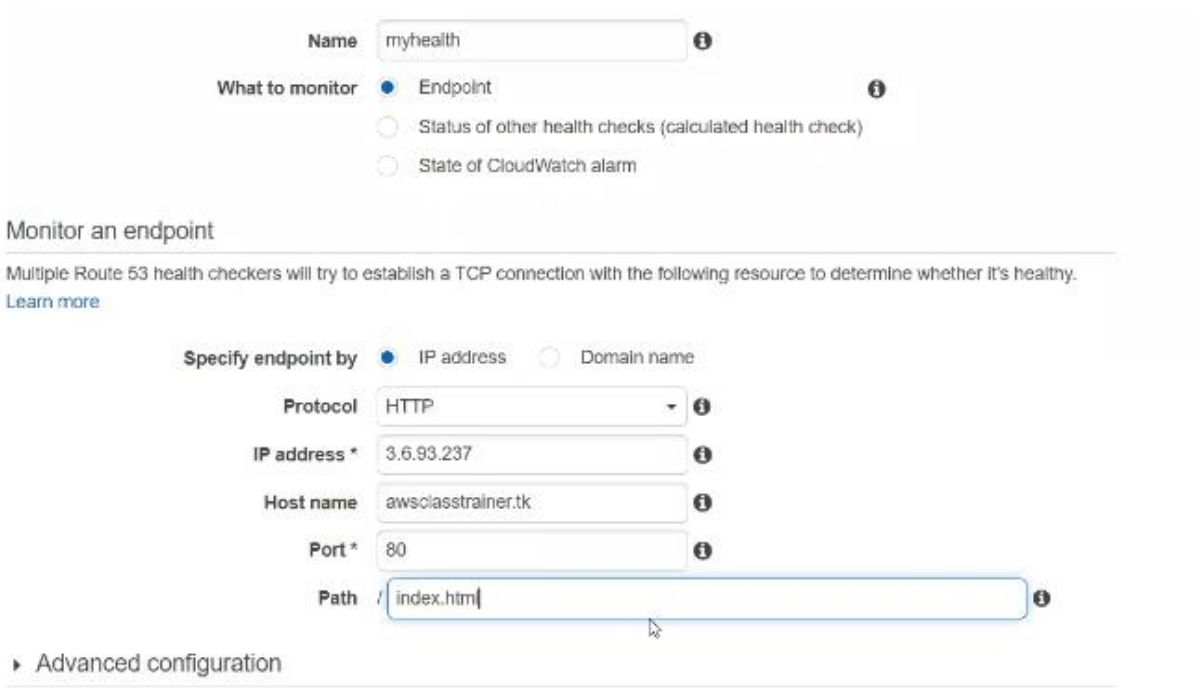
(*) Mumbai machine is primary machine copy Mumbai machine ip address



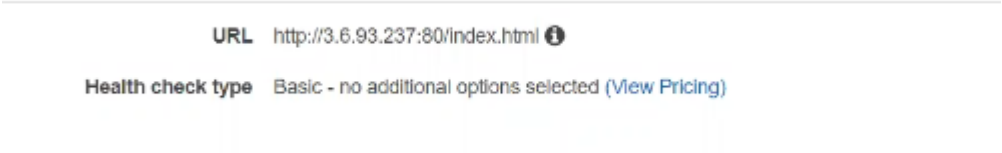
(*) choose health check id



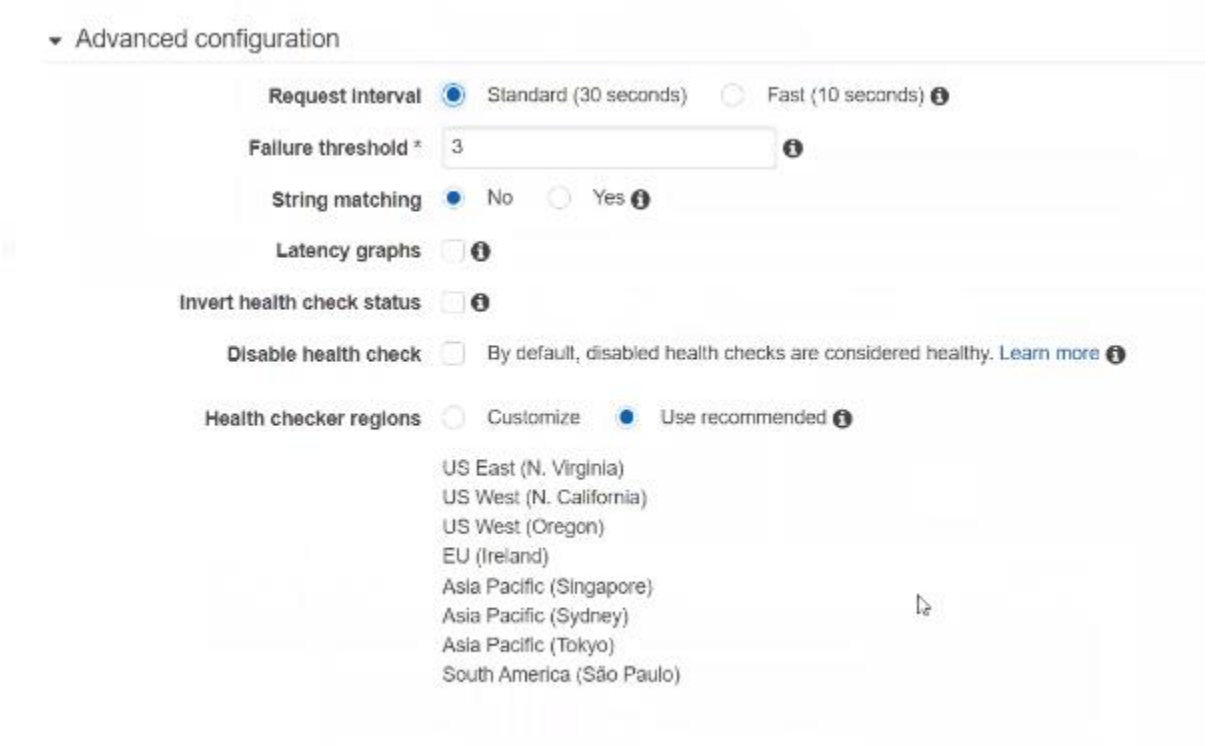
(*)



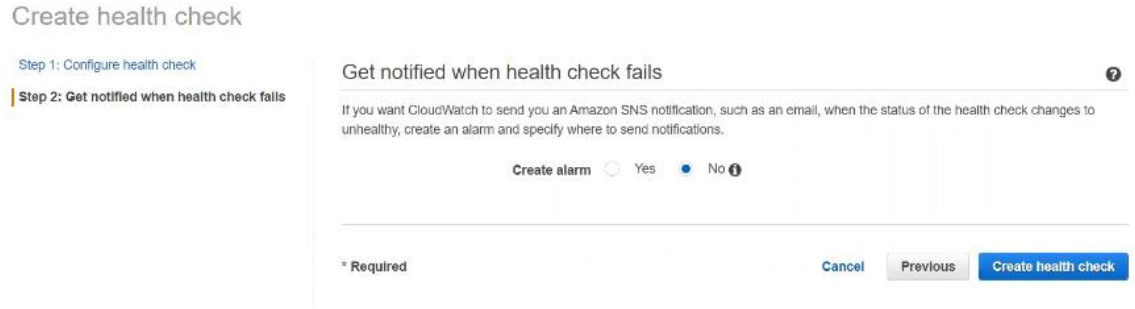
(*) an automatic URL is generated every 30s it will send request to main server the main server need to respond or else it will enter unhealthy state it will go to route 56 to reroute the traffic to another machine



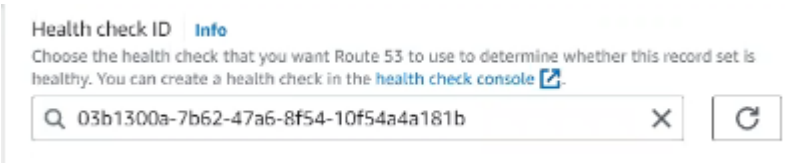
(*) advanced configuration – request interval 30seconds -- next



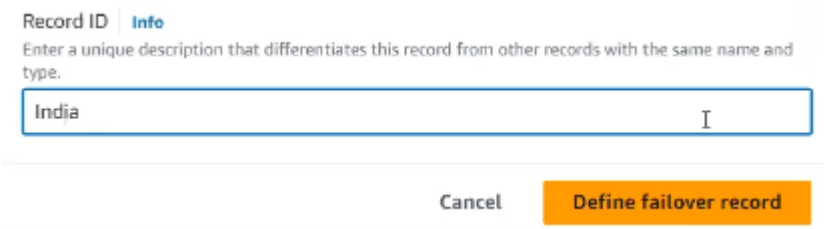
(*) get notified when health check fails notification service – create health check



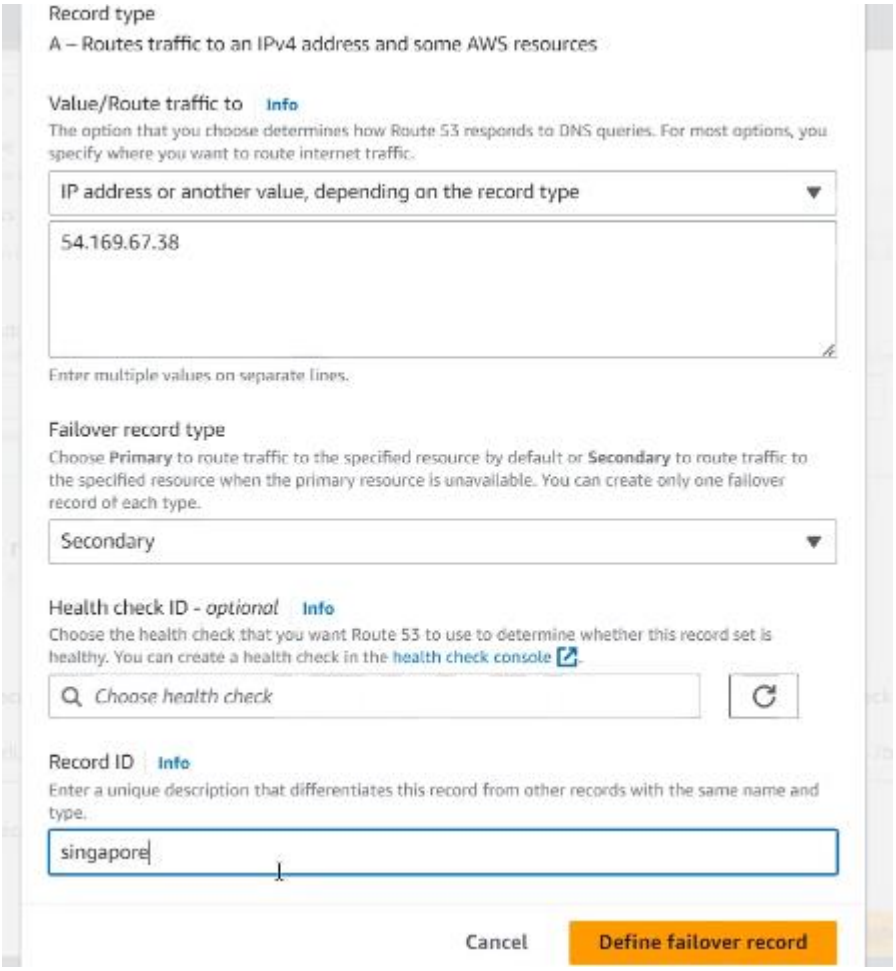
(*) this is our health check id



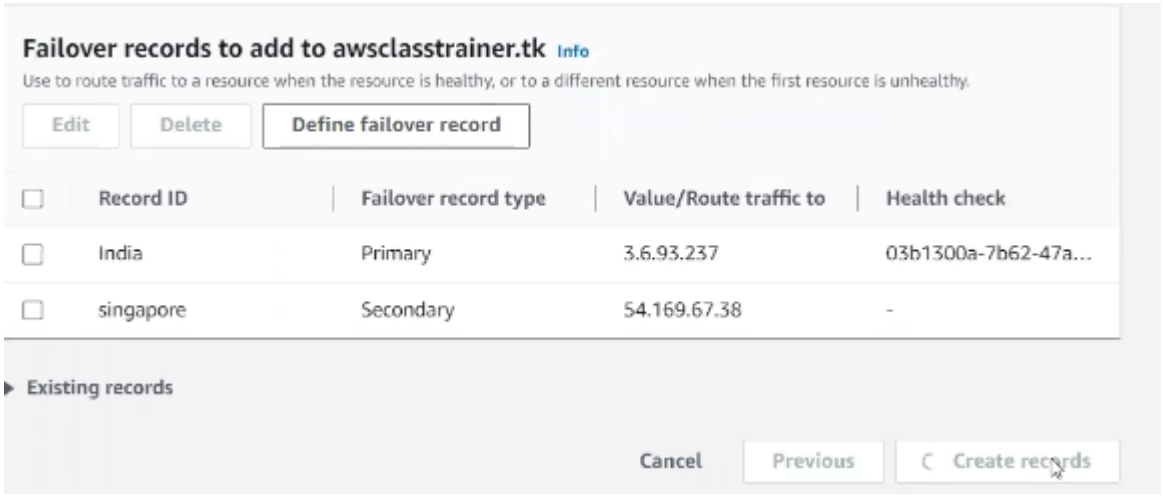
(*) Record Id – India – define failover record



(*) secondary machine Singapore – define failover record

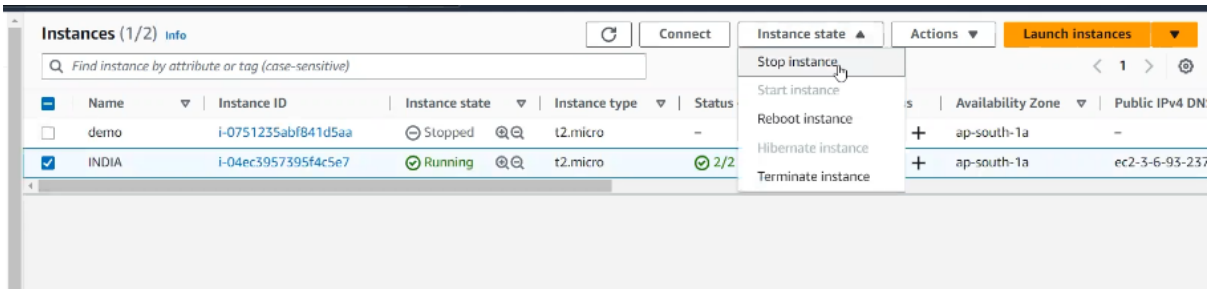


(*) create records



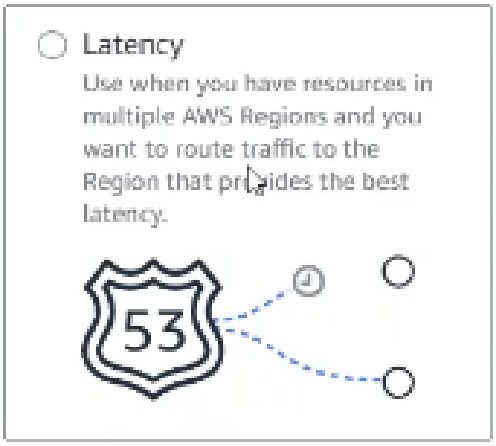
(*) THE TRAFFIC IS ROUTING FROM THE FIRST MACHINE

(*) NOW WE ARE GOING TO STOP THE FIRST MACHINE IN MUMBAI



(*) FOR EVERY 30S HEALTH CHECK WILL SEND REQUEST TO MAIN SERVER IF THE PRIMARY SERVER IS DOWN THEN IT TELLS ROUTE 53 TO REROUTE THE TRAFFIC

LATENCY POLICY



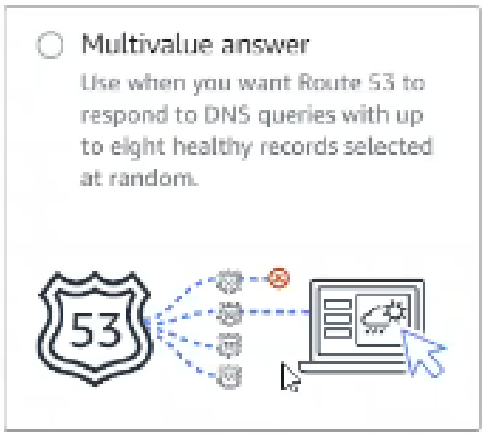
(*) Difference between latency and geolocation

Geolocation – routes traffic according to location

Latency - routes traffic according to region

(Eg) it will route traffic to better latency better perform

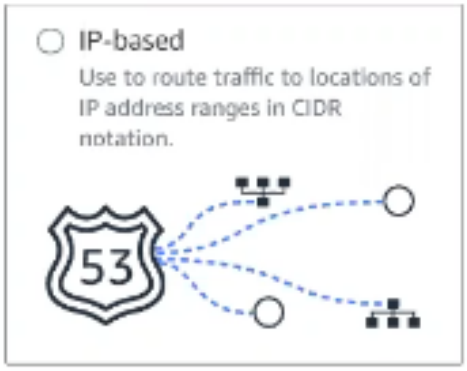
MULTIVALUE POLICY



(*) based on health check

4 name servers having health check ACCORDING TO HEALTH CHECK IT ROUTES TRAFFIC

IP BASED POLICY



based on Ip address it will route the traffic

ELASTIC BEANSTALK EBS (PAAS)

PAAS

|

APPLICATION

DATA

(*) used for developers

(*) end to end web application management

(*) .net, php, nodejs, python, java, ruby, go

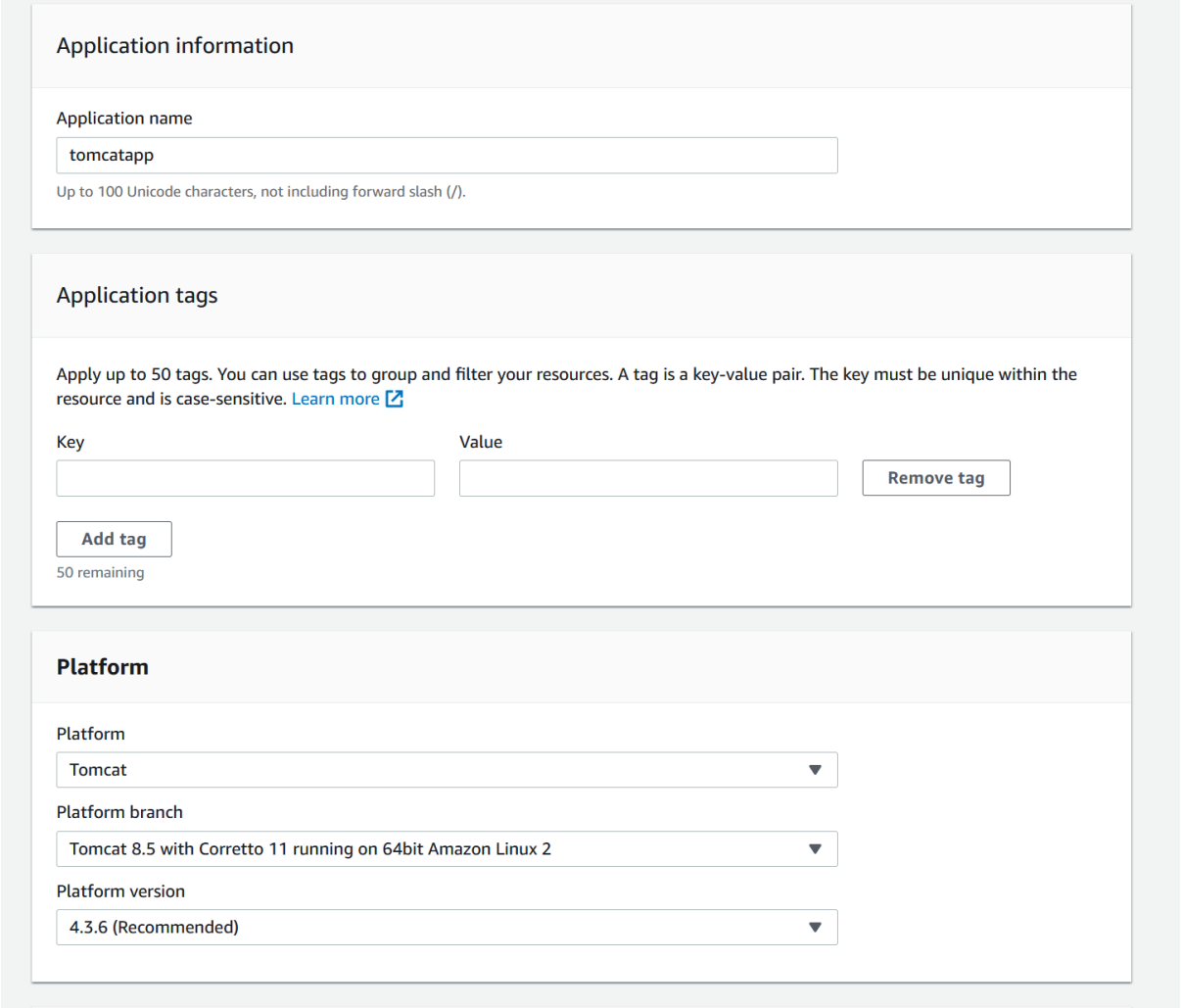
(*) apache, nginx, passenger middleware services

(*) Run and Manage Web Apps

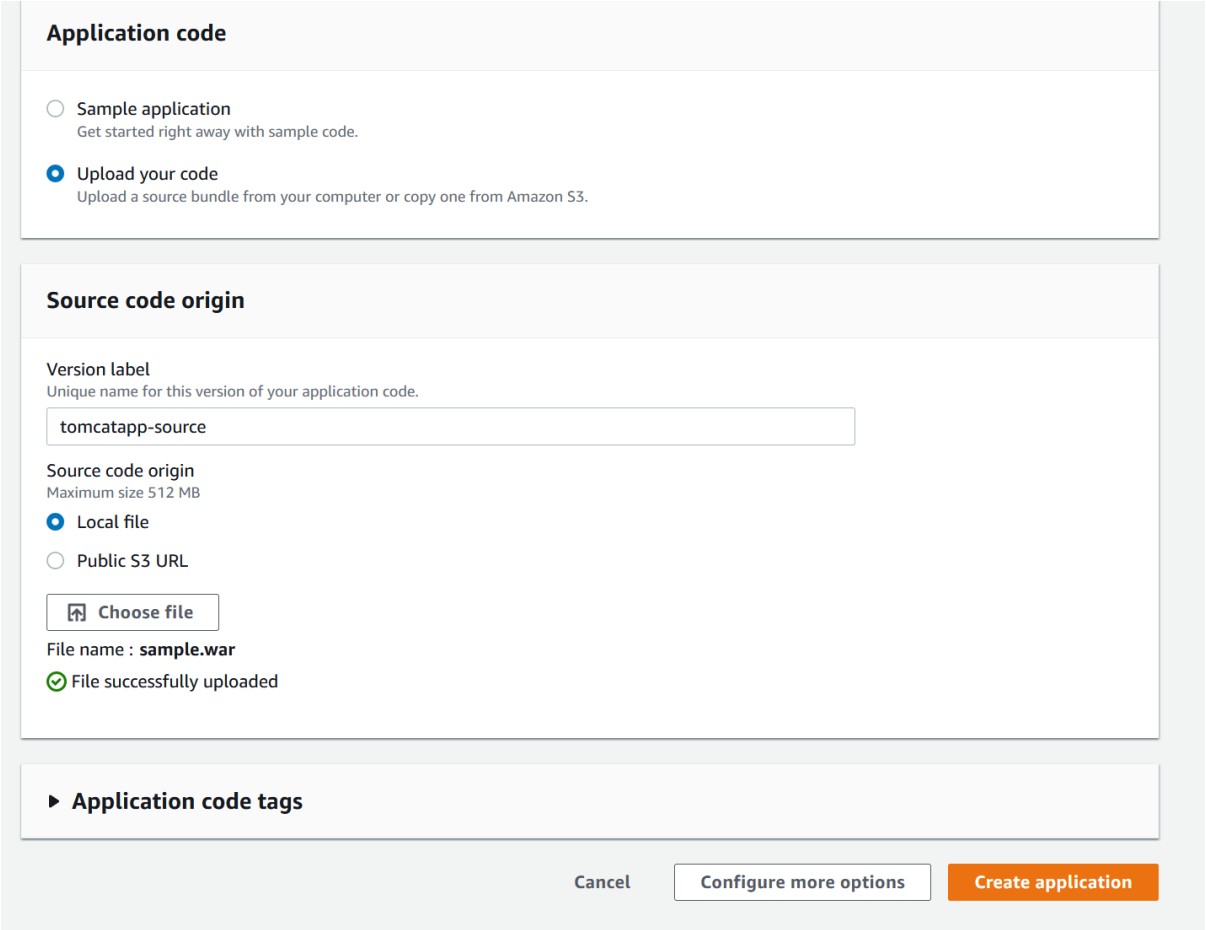
(*) search elastic beanstalk

(*) create an application

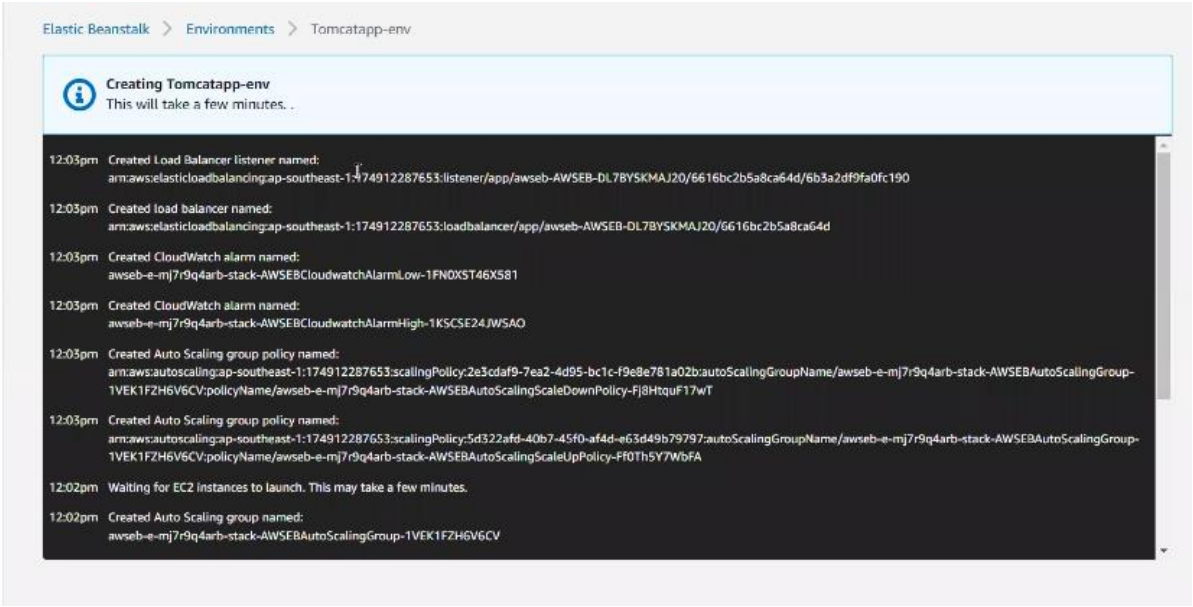
(*)



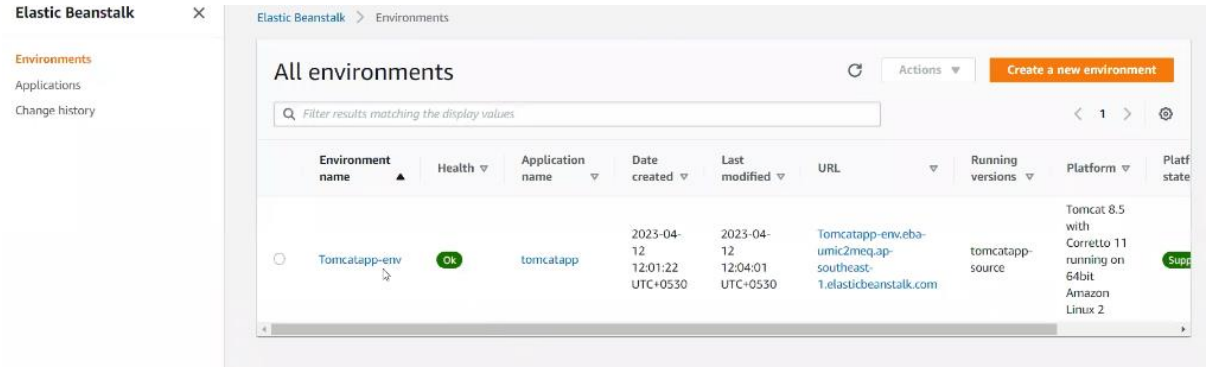
(*) upload smple.war



(*) creating tomcatapp



(*) ok health check created application



(*) click url to get output