AWS

AWS: Amazon Web Services.

How to launch instances: (EC2)

Go to EC2 dashboard → Instances → launch instances

↓

choose Amazon machine Image (AMI)
(Amazon Linux, ubuntu, Redhat, windows)

↓

choose instance type
(t2. micro free tier eligible)

↓

Configure instance
(no. of instances & VPC settings as default)

↓

Add storage (u can add storages)

↓

Add tags (Name, ec2-server)

↓

Configure security groups

[SSH - In port 22
  httpd - In port 80]

↓

Review & launch instances

↓

then choose [existing or creating new pair
(pem format downloaded)]

→ wait a few minutes then instance coming to Running state

→ putty gen → pem - ppk

→ after conversion of ppk file then Goto PUTTY, Host the Ip address and then SSH under Auth click. then load the 'ppk file' key.

→ After it's going to connect linux server.

login as : ec2-user (Linux (AMI)) → (now you are in ubuntu) (ubuntu AMI)

↓

pwd (present working directory)

↓

sudo su (Become to root user)

↓

yum upgrade/update (for upgrating server)

↓

yum install httpd (Installing webserver)

↓

systemctl status httpd

↓

if inactive (stopped;)

↓

systemctl start httpd

↓

then check status (running)

↓

now we are in httpd server. then enter Ip address in URL open browser.

↓

u got a Apache welcome window

↓

In linux, cd /var/www/html

↓

pwd (/var/www/html) → (httpd, Apache, nginx are format)

↓

vi index.html → (if u want write any content GOTO write)

↓ (i-key after)

- Welcome to AWS devops

:wq! (save & exit from file)

↓

Then IP address browser check. u see the O/P:

Welcome AWS devops //

(same as Image link also u can paste in html tutorials formats.

sudo su

↓

apt update

↓

opt install nginx

↓

systemctl (status. nginx)

↓

status (running)

↓

u are in nginx server

↓

u got nginx welcome window.

↓

cd /var/www/html

↓

pwd (/var/www/html) ls -index.nginx-

↓

vi index.html //

paste - click right

→ for web servers (httpd, Appache, ngnix) - open port 80 means users connected to web server.

→ When I am connected to server - open port 22

→ for Jenkines - open port 8080

Save - { Esc then
shift plus :

***

Putty, mobiextrume & Git Bash

* These apps are used for interfaling the linux machines which are present in AWS cloud to from my local machine (windows10)

* used for login to remote Operating system using SSH.

***

if you are using ubuntu, centos operating systems you don't need putty, mobientrne & Git bash for conecting to linux servers which are present in aws cloud. you having default app terminal option (In ubuntu)...

** EC2 instances - Dynamic Ip's - means when u are stop & starting instances the Ip address will always changes (this is caller down time to users)

** Elastic IP: statie Ip's - means when you are stop & starting instances the IP address remains Constant

Elastic Ip's :-

1) By default you can create 5 elastic Ip's per region.

2) Creating :-

Goto Elastic Ip's → Allocate Elastic Ip address → choose regions → then its created.

3) when we allocate this elastic Ip to EC2 instances the public Ip remains constant even if stop & start the instances.

3) If you want more than 5 elastic Ip's in region. then you can request for the increase in quota of elastic Ip address (Elastic Ip Quata increase)

4) Elastic Ip pricing:

* there is no charge incurred if elastic Ip is attached to an EC2 instance.

* If you are created then not attached to the instance so u may pay charged per hour basis.

* If you don't need an elastic Ip, you can release the elastic Ip to stop charges appear on your account.

→ userdata :

→ Boot strap our EC2 server.

→ you can pass user data while the creation of EC2 instance, the instance is boostraped with scripts commands that are passed to user data.

* If you want php, maridab, httpd web servers to be readialy available night after instance creation all these can installed in user data section.
* user data can be configured only during when the instance is created for the first time, if you want make any changes in user data, the instance must be stopped & access the userdata to make any changes.

```
#!/bin/bash
sudo su
yum update -y
yum install httpd -y
system start httpd
System enable httpd
echo "this content is from EC2 user data" -> -
/var/www/html/index.html
```

---

for Ubuntu:-

```
#!bin/bash
=
=
apt update -y
apt install nginx -y
=
systemctl start nginx
=
=
echo "welcome to devops" => /var/www/html/index.html
```

*) we can Edit the userdata when instance be in Particular 'Stop' position.
Goto actions -> Instance Settings -> Edit userdata.

AWS cloud

Before cloud: we having data centers. its requires space, networking .... etc.

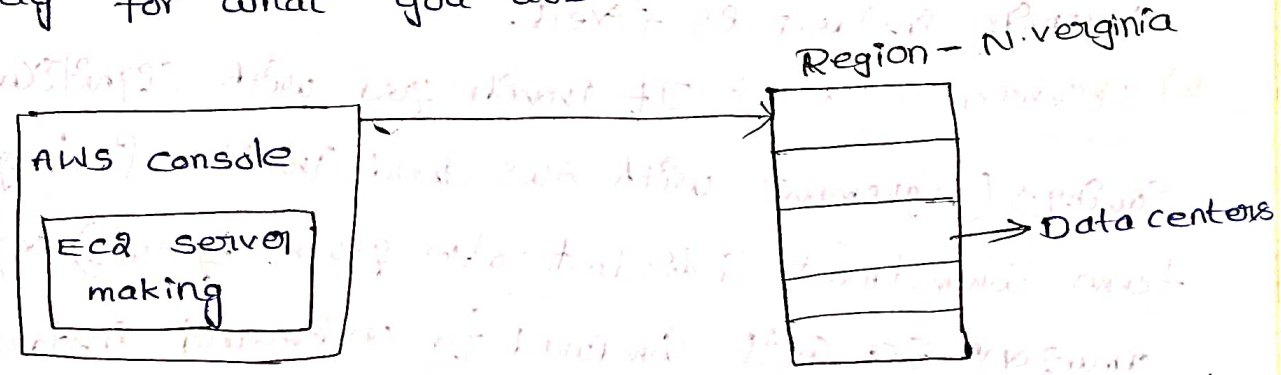
→ If we are using 10% of usage also you should pay money of 100%.

Cloud: It is delivering of computer services like services, storage, database, networking, software and analytics.

Amazon has decided to start & supplying data centers.

Aws – selling It services (servers, storage .... etc)

- pay for what you use.



when you creating server the notification will send to data base (Ram capacites details)

***

for instance details checking on aws linux server (through putty)

→ Curl http://169.254.169.254/latest/meta-data

All (Install Id, public-IP's ..... etc)

→ Instance type also we can change when instance in Stop Position.

→ If you want only Particular Public-IPV4 Address, then

Curl http://169.254.169.254/latest/meta-data/PublicIPV4

Termination Protection:- It helps Prevent EC2 server from accidental termination. you will never delete the instance when you are using termination Protection (Enable click) → Click actions → Instance settings → change termination Protection.

EC2 Purchasing options:-

1) on Demand instance:- there is no long term commitment required when you Purchase on demand instance. you Pay only for seconds/hr that your on demand instance are on the running state. the Price Per second Running instance is fixed.

2) Reserved instance:- It Provide you with significant Savings (Agreement with Aws cloud Provider for long term commitment & discount also given by Aws) on amazon EC2 costs Compared to on demand instance Pricing.

3) Spot instance:- this An instance that uses spare EC2 Capacity that is available for less then the on demand Price. because spot instance enable you to request unused EC2 instance at step discounts, you can lower your amazon EC2 cost significantly. the hourly Price of spot instance is called spot Price.

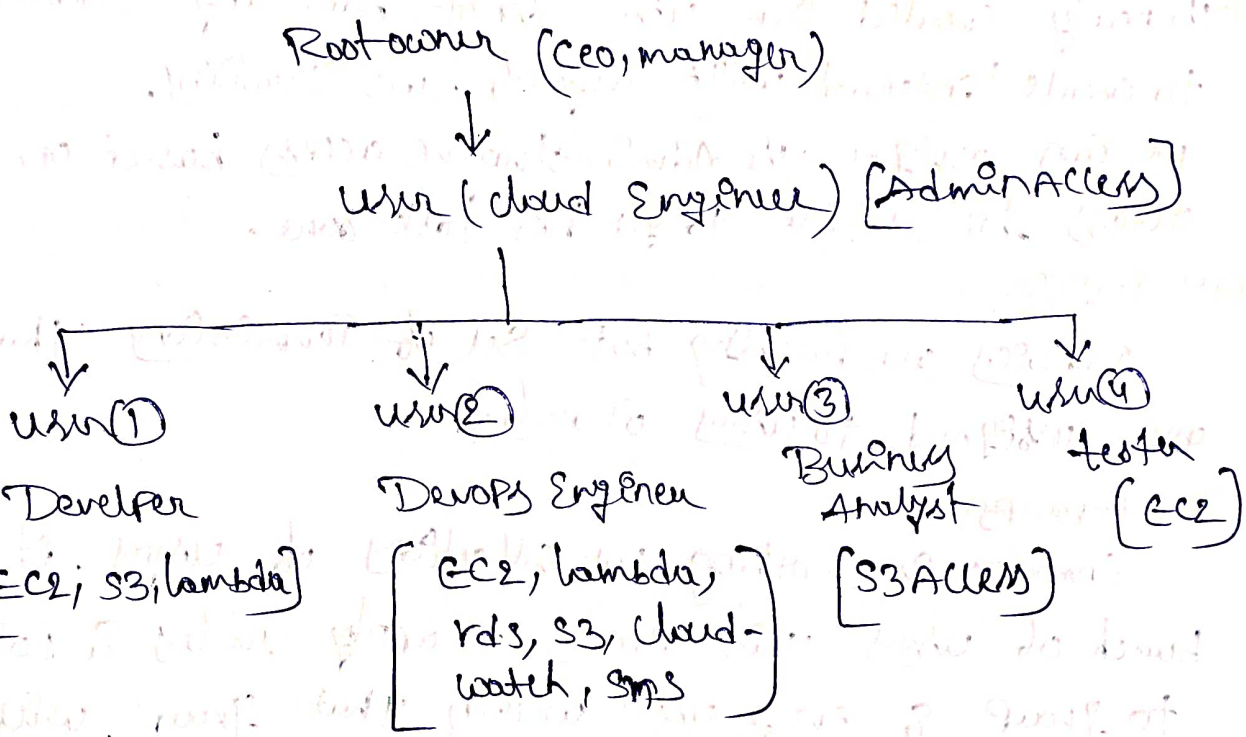
Spot Price to each instance type as each availability zone set by amazon EC2 & is adjusted Gradually based on long term supply.

IAM:- Identity Access Management

Root Account:- when you created AWS account you are like Root owner, Super user & Power user.

↳ It having All Previlages, resources & Services

↳ Generally Root account operates may be CEO, leadership, managers & Directors.



↳ To have a greater Security, MFA needs to be Enabled, you can use Google-AuthenticatorApp to set up this.

IAM:- It is Advised to keep your root account Separately, Inorder to do this you can create an IAM user & give neccessary Permissions that you would require.

IAM helps you manage AWS users, Groups & give

Permissions based on the Principle of least Privileged only to the resources in aws that are needed for users.

least Privileges access means user should have only the Permissions require to perform the tasks.

IAM user:- An user is an entity that you create in aws account so that newly created iam user can interact with Aws Services.

when IAM user is created, you can give him the Aws Console access or Programatic access. By default the newly created iam user donot have any Permissions to create/interact with any of Aws Services.

we can assign the Administrative access based on what access he require to get his task done.

IAM Policies:-

Policies are nothing but Set of Permissions that are assigned to users or roles.

IAM Groups:-

Groups is nothing but collections of users. it refers bunch of users. this allows to apply roles & Policies to group & every user within that group will automatically have their Roles & Policies.

***:

↳ user(teja) is cloud Admin so the entire Aws account managed by iam user(teja). open the link, Crop iam user(teja) account then here you creates users like devops, then here you Creates users like devops, Developer, tester & business Analyst & Give

neccessary Permissions to users. there users will in under of IAM user(teja).

↳ If you create Groups [Devops Engineer, developer] simply you can add the user group when you creating user (Add user), that means whatever existing Permissions in Groups those Permissions will be applied to user.

whenver user leaving in organization then simply you can go in Groups & see user & select the user & remove it. Easily Edit or removes the Permissions in Group also.

Amazon S3:- It is an object storage. Best fit for pictures, videos & highly durable media storage. By default you can create 100 buckets in AWS account.

Goto 'S3'

↓

Create bucket

↓

choose 'bucketname & region'

↓

Select Create bucket.

Here whatever you created bucket that you can open & upload any documents, images & files from your local system.

IAM Role:- In AWS account every service is independent service. So inorder to interact with the resources & Services we must require Roles.

↳ create any one Role suppose [EC2 to S3] & attach to any one Instance then GoTo Putty & login,

↓

Sudo su

↓

Pwd, ls

↓

Apt update

↓

In order to upload any Content to S3 & download any Content to server you must have awscli utility.

AWSCLI :- It is command line interface tool using which you can interact & manage multiple aws services from the command line.

↓

apt install awscli

↓

aws s3 ls [list of buckets shows]

↓

for showing files in particular bucket aws s3 ls s3://s3nov9

↓ (amazon.txt)

aws s3 cp s3://s3nov9/amazon.txt ⊙ (current directory)

[from S3 to Server]

↓

ls [showing file]

amazon.txt

↓

seeing Content, Cat amazon.txt

↓

Creating file, touch testfile

↓

echo "welcome DevOps" >> testfile.

↓

aws s3 cp testfile s3://s3nov9

source destination

[from server to s3]

↓

then you can download these files from s3 to local system.

↓

aws s3 ls

s3nov9 → [amazon.txt, testfile]

↓

You can delete role also from server → Goto Instance → Actions → Security → modify Role → no roles applied → ok.

Programatic access for user:-

In IAM} Create Iam user & Give Programatic access section} after download csv file & getting access key & secret access key.

if you forget access keys then goto user & Security delete & Again Create.

After creating Ec2 server, Connect to Putty,

login (ubuntu)

↓

sudo su

↓

apt update

↓

apt install awscli

↓

↓

aws configure

↓

Give access & secret keys & output, region.

↓

aws s3 ls [view list of buckets]

↓

we have awscli commands for interacting & creating other services with the help of CLI commands if which service you want to create for that you use documentation & create it.

Load Balancer:- It takes request from users & distributes the load traffic Equally to across registred instances.

1) classic load Balancer (CLB)
2) Application " (ALB)
3) Networking " (NLB)
4) Gateway " (GLB)

1) CLB is no longer usage because becoming it by deprecated. once configured it distributes the load across all registered instances regardless of what is present on the servers. it doesn't supports features like host-base routing & path based routing. it can only be used to distributes traffic a single URL.

2) This loadbalancer specially design for web application with http & https traffic. this ALB works this application layer. it also Provides Advanced routing features such as host based & path based routing & also works with Containers & microservices.

↳ Host based Routing :- Suppose you have the two websites opencloud.com & Admin.opencloud.com. Each website hosted on two EC2 Instanes for high availability & you want to distribute the incoming web traffic b/w them.

if you were using CLB you need to use create two load balancer, one for each website But you can do same thing using single ALB. Hence you will saving money as well as you will pay for single ALB instead of 2 CLB.

↳ Path based Routing :- Suppose the website of your Company is opencloud.com & the Company's blog is hosted on opencloud.com/blog. the operations team has decided the host the main website & blog on diffrent instances

using ALB you can route traffic based on the path of the requested URL. Again single ALB enough for handle for this.

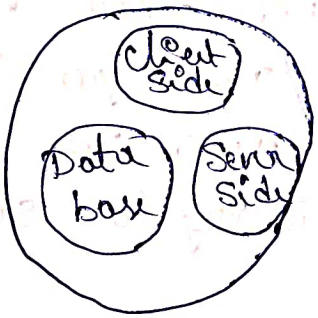
3) this load balancer operates at network layer of OSI model. Suppose you Company's website running on four m4-xlarge instanes & you are using an ALB to distribute the ~~distribute~~ traffic among them.

now your company launched new product today which got viral & your website starts to get millions of request per second.

In this case, ALB may not be able to handle the sudden spike in traffic.

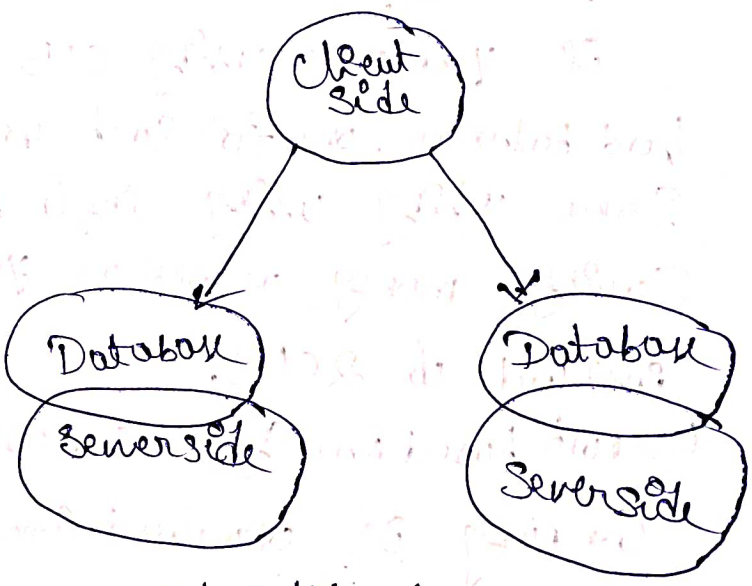
So this is where the NLB really shines. It has capability to handle a sudden spike in traffic since it works at the connection level.

Monolith

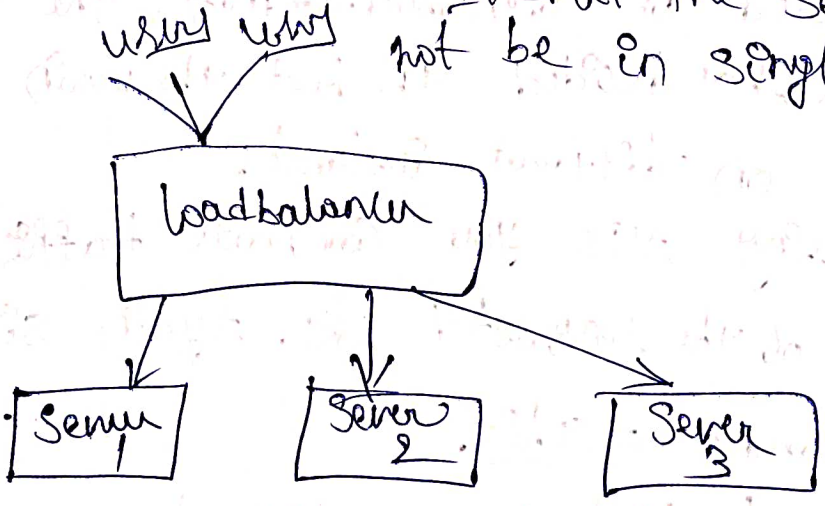


the entire application, all modules running in single unit.

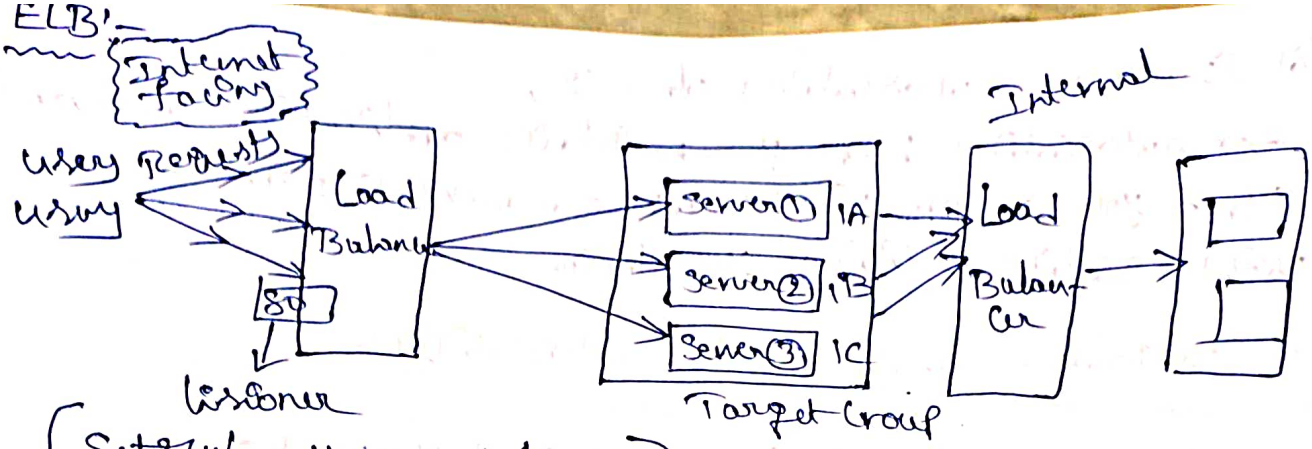
Microservices



It split the services, not be in single unit.

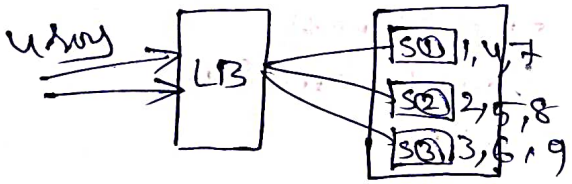


ELB:-



Security, you need to open Port 80 in listener for users traffic Receiving Purpose

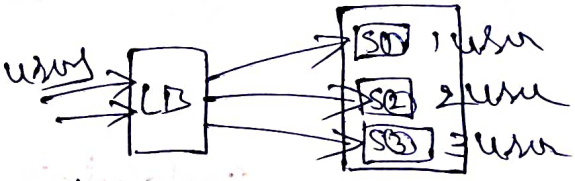
Roundrobin Algorithm In ALB:-



Round robin Alg. means the users can hit to servers then it Equally goes to trafic all servers.

Even if you have done Page refreshing also.

flowhash Alg in NLB:-



it means, the user① can hit the server, the traffic goes to 1st instance & 2nd user can hit the Server the trafic goes to 2nd instance. if 1st user refresh the page at the time trafic sends to 1st instance not sends 3rd instance.

Auto Scaling:-

It is monitors your application & automatically adjusts capacity to maintain steady, Predictable performance at the lowest possible cost.

for Example Autoscaling of EC2,

EC2 Autoscaling helps you maintain application availability & lets you automatically add & remove EC2 instances using scaling Policies you define.

Creat LoadBalancer & Attach to Autoscaling :-

first Creat two instance & install webserver in that.

I) After that Create Target groups

↓

1) Specify group details :-

Choose target type — Instance ✓
IP Address
Lambda function
ALB

Protocal → HTTP — 80

VPC default →

Health checks, HTTP (Protocal) & Path →

Advanced health check settings :-

Healthy threshold → 5 [if you get 5 then server working good]

Un " " → 2 [if you get 2 also then server unhealthy]

Timeout → 5 [5 times checks each time takes close]

Interval → 20 [Every 20 seconds one checking]

2) Register the instances here what we need.

II) Go TO load balancer →

load balancer name →

Internet facing ✓

IPV4

VPC as default:-

mappings:- Here you choose register instances subnets or you can select all

Security groups:- default & you must open Port 80.

Listeners & Routing:-

Protocol-80 & select target group

↓

Create load balancer.

III) Auto Scaling [Launch Configuration Auto Scaling Group]

After deregistering the instances in target then click Auto Scaling,

1) Goto Lunch Configuration,

↳ name:- , AMI:- , Instance type:-

↳ Everything as default, Sec. group Add Port 80 Protocol & existing Security also you can select.

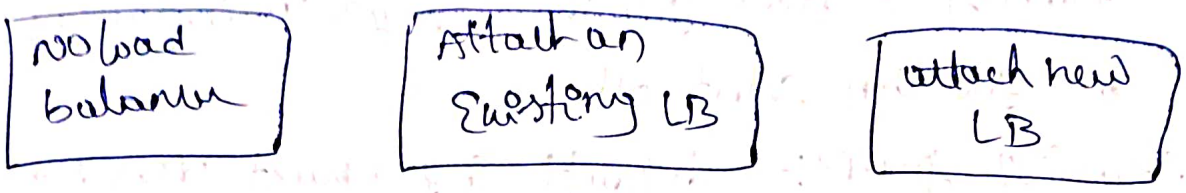
↳ choose key pair the Create lunch Configuration

2) Goto Auto Scaling Group,

↳ select launch Configuration

↳ choose instance lunch option (we must Contain instances in LB), VPC as default & select availability Zones.

↳ Configure Advance options,



Select target group! ——

↳ Configure group size & Scaling Policies

Desired capacity - 2

minimum " - 2

Scaling Policies - none

max capacity - 2

↳ Add notification, tags & review

↓

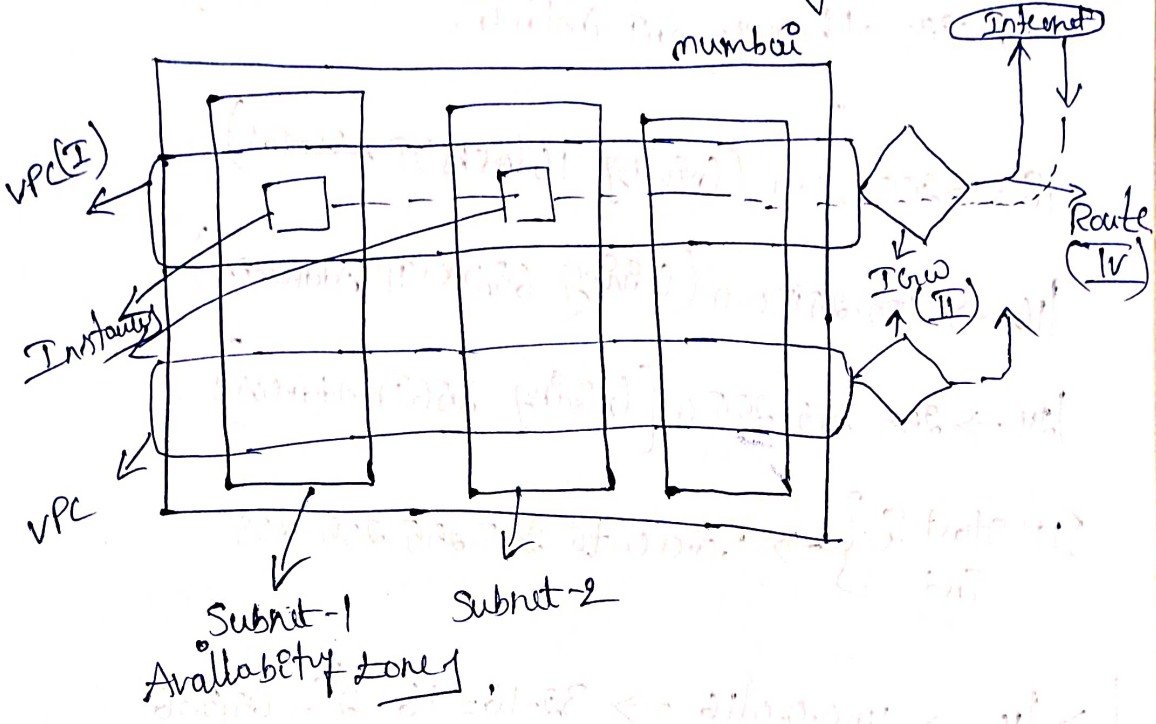
Create Autoscaling group

then open 'DNS' name in browser & search you get server① & refresh then server②

↓

Here if you delete & stop one instance then Automatically another one coming.

VPC (Virtual Private Cloud):- VPC allows us to create an Isolated network in the cloud. We can have Complete Control on our resources that are being hosted in VPC.



Route table:- A Route table Contains set of rules Called routes that are used to determine where network traffic is directed. Each route in table specifies a destination CIDR & target.

Public Subnet:- If the Subnet traffic is routed to an internet gateway, the subnet is known as Public Subnet.

if a subnet as a route the destination (0.0.0.0/0) & internet gateway as target, the subnet is known as Public Subnet.

Private Subnet:- Users can't see whatever present in Private Subnet.

if Subnet doesn't have route to IGW the subnet is known as Private Subnet.

↳ the Public subnet route internet trafic through the internet gateway.

↳ for Private subnet, the Rivate subnet routes internet traffic through NAT instance.

/8 → 255.0.0.0 (Giving 16 lakhs IP Address)

/16 → 255.255.0.0 (Giving 65,000 IP Addresses)

/24 → 255.255.255.0 [Giving 256 IP Address]

IP Start & End } → 1.0.0.0 to 255.255.255.255

↳ /16 → 10.0.0.0/16 → 32-16 = 16 = 2^{16} = 65,536

Each digit 8 bytes
[total - 32 bytes]

Any 65,536 instance can created for Particular VPC

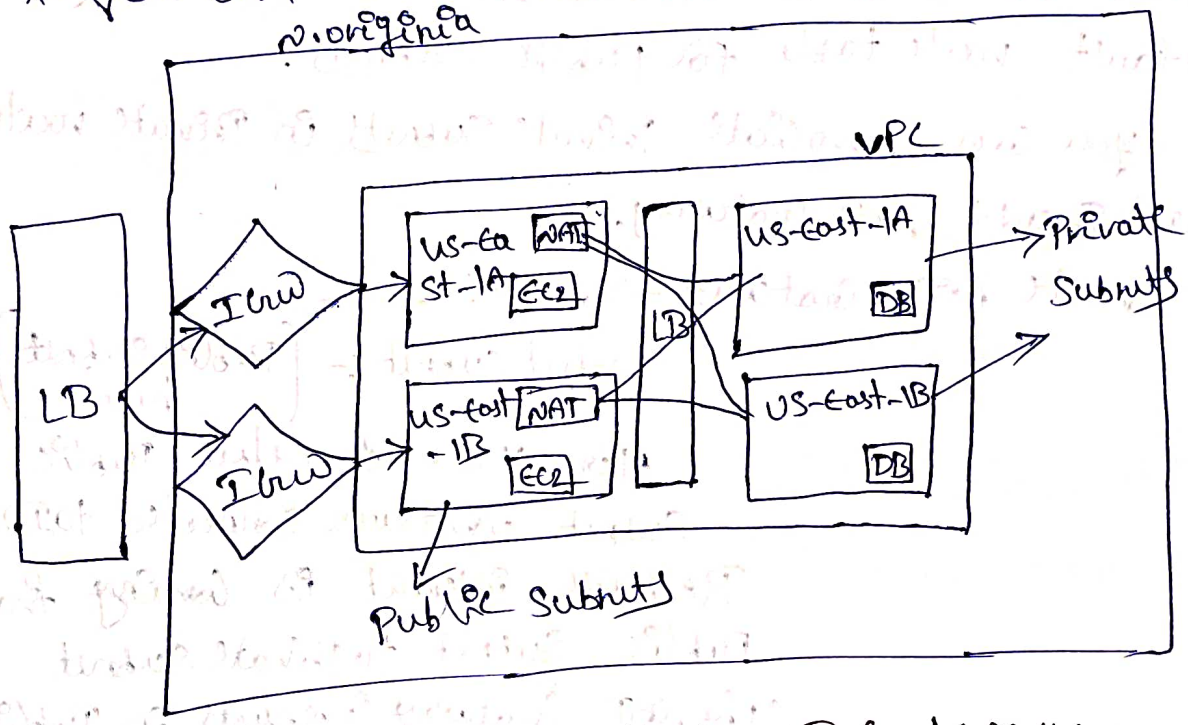
↳ 10.0.0.0/24 → 32-24 = 8 ⇒ 2^8 = 256

Any 251 instance can created for Particular Subnet

Remaining '5' instances AWS reserved.

NOTES!-

* for default VPC, all subnets having IGW.
* you can't attach more than 1 IGW to VPC
* you can create 5 Elastic IPs per region.
* you can create 200 subnets per VPC
* you can create 200 route table per VPC
* you can create 5 IGW per region
* you can create 500 security groups per VPC



Steps to create VPC, IGW & NAT Gateway:-

↳ 1st you create VPC & give name & enable here.

↳ you create IGW & attached to VPC

↳ you create 2 Public subnets, 2 Private subnets for select VPC ID (whatever u r created).

→ when you creation of subnets, VPC route table default can be added. So you can create another route table for removing of default VPC route table

↳ you create Public route table (add route IGW) & Private route table (add NAT Gateway).

→ Here you can associate the Public Subnets in Public route table & add the IGW (0.0.0.0) route.

Routes	Subnet Association	Edge Association
↓		
Destination	Target	Status
10.0.0.0/16	local	Active
0.0.0.0/0	IGW (Select here)	Active

So now Public Subnet route table is Public route table & these routes can be applied also. Now not having VPC default route table for Public Subnets.

→ Why you can associate Private Subnets in Private route table & add NAT Gateway.

⇒ Create NAT Gateway → name :- ——

Select Subnet :- [Public Subnet here]

Here you must select Public Subnet whatever connected to IGW. Because internet is coming from Public Subnet to Private Subnet The NAT Gateway Presents in Public Subnet.

Elastic IP :- ——

↓

Create NAT Gateway.

Routes	Subnet association	Edge Association
↓		
Destination	Target	Status
10.0.0.0/16	local	Active
0.0.0.0/0	NAT (Select here)	Active

So now Private Subnet route table is Private route table & these routes can be applied also.

↳ Create EC2 instances (2) for Public & Private,

↓ for Public Subnet

no. Instances: ____

VPC: ____

Subnet :- Public Subnet

Autoassign: Enable

↓

After launching of EC2 then open Putty then easily Connected to Server

3rd step Configure details.

↓ for Private Subnet

no. of instance: ____

VPC: ____

Subnet :- Private Subnet

Autoassign: Enable

↓

After launching EC2, open Putty then here not able to connect to the Server because this is not connects to IGW

↳ for external Private Subnet Connection, open Putty [whatever Public Subnet is Connected] After

↓

Sudo su

↓

Ping IP [Private Subnet instance IP]

↓

ssh ec2-usr@IP(Pvt)

[Here Permission declined]

↓

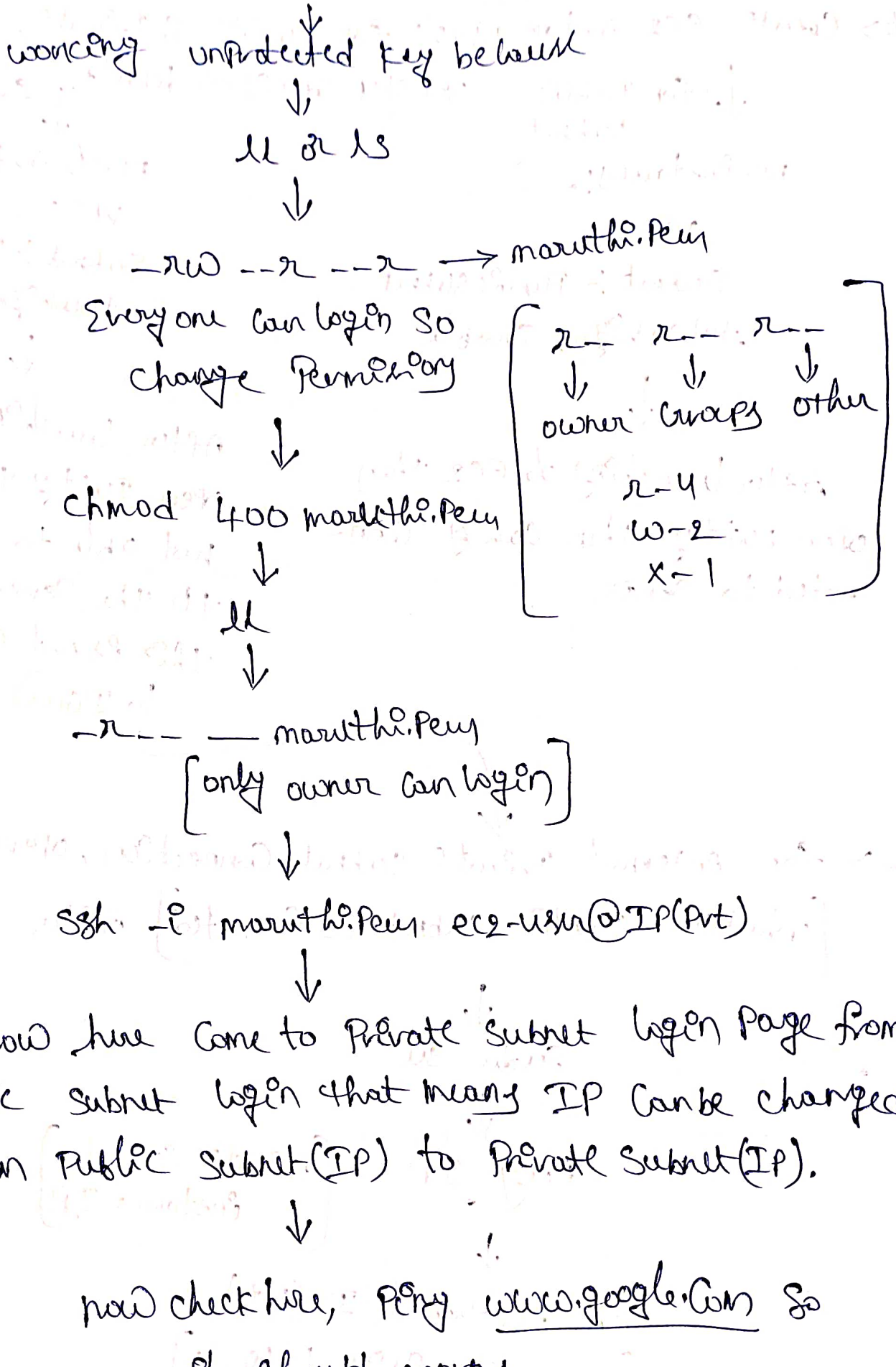
nano marutheri.pem

Paste the key & Come out

↓

ssh -i marutheri.pem ec2-usr@IP(Pvt)

[ctrl+O ↓ ctrl+S → save
ctrl+X → out]

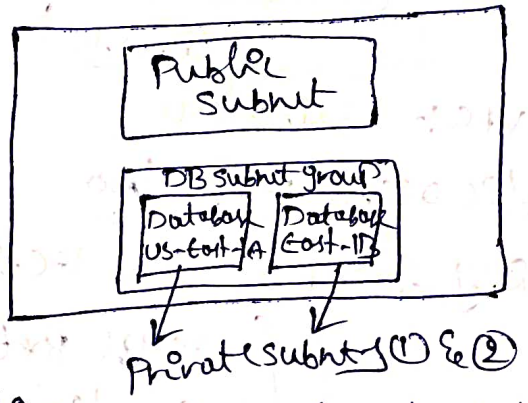

working unprotected key because
ll or ls
-rw- --r- --r- → marathi.Pem
Everyone can login so
change Permissions
chmod 400 marathi.Pem
ll
-r-- --- marathi.Pem
[only owner can login]
r-- r-- r--
owner Groups other
r-4
w-2
x-1
ssh -i marathi.Pem ec2-usr@IP(Pvt)
now here come to Private Subnet login page from
Public Subnet login that means IP can be changed
from Public Subnet(IP) to Private Subnet(IP).
now check here, ping www.google.com so
it should work.

RDS (Relational database service) :-

It makes easy to setup, operate & scale a relational database in cloud. It Provides cost-efficient & resizable capacity while automating time consuming administration tasks such as database setup, patching & backups.

The Amazon can be managed by databases. there diff DB's,

- ↳ mysql
- ↳ mariadb
- ↳ Postgres sql
- ↳ ms sql
- ↳ aurora
- ↳ oracle.



In Production enviornment the database is highly available that means Databases maintaing in minimum two subnets.

Database Creating :- Goto RDS

Step ① :- Create & select Subnet groups,

↓

Name, Description,
VPC: select the
Availability Zones :-
Subnets [you must select At Subnets]

↓

Create DB Subnet groups.

Step ② :- Goto & select Database,

↓

choose database :- Standard ✓
Easy create

Database → Aroa
mysql ✓
mariadb
Postgresql

templates :- Production, Dev/test, free tier
[for free we going to create only one Private Subnet DB]

Credential settings :- [user, password for DB]

multi AZ deployment :- In Production environmental multiple Pvt Subnets select but in free tier only one Pvt Subnet
Comes

DB Subnt Grp, VPC :- select this

Public access :- yes/no ✓

VPC Security Group :- new sec. group
[mysql Port 3306 add &
Create.
keep anywhere IPv4]

Availability Zones :- No Preference
us-east-1A
us-east-1B
[if not select Subnt Zone then it can choose anyone out of both two.]

↓

Creat Database.

Step ③ :- now Connect to the Database through Public Subnet Connection of Putty,

↓

Sudo su

↓

apt update

↓

apt install mysql (or) mysql-server

↓

mysql -h [Endpoint hostname] -P(Port) -u(user) -p(Password).

↓

Spaces

↓

You are in mysql [show databases;]

↓

[use mysql;]

↓

Database changed

[show tables;] → all table shows here

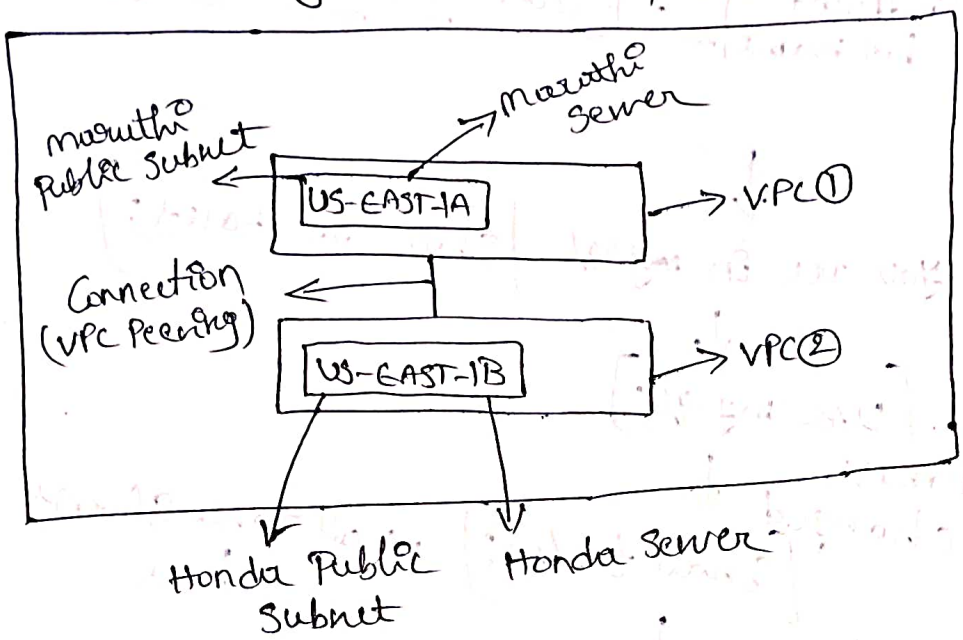
↓

select * from user; → if you want see data in user table

↓

exit

VPC Peering :- A VPC Peering Connection is a networking Connection b/w two VPCs that enable you to route traffic b/w them using Private IPV4 or IPV6 addresses.



Create VPC Peering Connection :-

↳ Create maruthi Server in VPC① (Give route table for Public Subnet & add route IGW)

↳ Create Honda Server in VPC② (Give route table for Public Subnet & add route IGW).

↳ After lunching the instances then Connect maruthi Server to Putty, initially here check Connection from VPC① (maruthi Server) to VPC② (Honda Server) through Private IPs. So result is no Connection.

Step①:

↳ Goto VPC Dashboard then click Peering Connection

↓

Peering Connection name: _____

select local VPC to Peer with: _____

VPC (Requested): VPC①

Select another VPC to peer with:

Account → ① My account
② Another account

VPC (Accepter): VPC②

↓

Create VPC Peering Connection

↓

Here "Accept" the Peering Connection in Actions

Step②:- Goto Route table

↓

maruthi route table

↓

Edit routes & add Peer Connection with IP (Honda) [with local, IGW]

↓

Honda route table

↓

Edit routes & add Peer Connection with IP (maruthi) [with local, IGW]

So Connected is established

Step③:- Here check in maruthi Server.

Ping IP (Private) → Honda Server

(or)

login → ssh -i maruthi.pem ec2-user@IP (Private)

↓

Honda Server.

AMI (Amazon machine Image) :-

First you create one instance & connect to server through putty & gitbash & install webserver.

After creating actions (instance)

↓

click, image & templates

↓

Create Image

↳ so this image created in "AMI" option check also.

↳ so through this AMI image you can lunch the instance also. Here whatever instance you lunched through AMI Image, here already installed webserver coming for lunched instance.

↳ this image also can copies to another region & another accounts also.

[this is very helpful for whatever data (application) is present in server that can easily expandy to any place & recovery also]

Volumes & Snapshots :-

Volumes :- An Amazon EBS volume is a durable, block level storage device that you can attach to your instances.

Creating volume → Create volume (give memory details & Zones)

↓

Create volume

↓

After In action you can create the Snapshot

In Actions :-

1) Create Snapshot
2) modify volume
3) Delete "
4) Attach "
5) Detach "

Snapshot :- EBS Snapshot are a point in time copy of your data & can be used to enable disater recovery, migrate data across regions & accounts & improve backup compliences.

↳ if you delete the volume also you can recovery the volume data from Snapshot.

↳ Snapshot also copies to another region & another A/c.

Cloudwatch :- It is a monitoring & management service that Provides data & actionable insights for AWS.

↳ first initially you lunch one instance the check metric for instance.

↳ for creating cloud watch Alarms you need to Create first SNS topic.

Step 1 :- SNS (simple notification service) :-

It is a logical access point that acts as a communication channel. A topic let's you group multiple end points [such as Aws lambda, HTTP & emails].

Goto SNS → create topic

↓

Type : fifo, Standard ✓

↓

Create topic

then "Subscribe" this topic.

Goto Subscription

↓

Choose SNS topic: ——

Protocol : Email, SMS, HTTP

Endpoint : mail id give here

↓

Initially this is in Pending Confirmation so you open the mail & "Confirm" subscription later this subscription is "Confirmed state.

Step(2):- Goto cloudwatch

↓

first Create dashboard

↓

Select wedge type [line, Area, number, Bar]

↓

Select metrics or logs

↓

Configure & choose metrics

Application ELB, EBS, EC2, logs, NAT Gateway, network ELB, SNS, S3, RDS.

↓

By Autoscaling group

Pre-instance metrics

↓

Choose InstanceId & for instance Select

CPU utilization

metadata no token

Status check failed

CPU Credit Balance

↓

Create it & then Save dashboard.

Step(3):- Goto in Alarms & Create Alarm

↓

Select metric

[EC2 & Pre Instance]

↓

Why choose Instance id & Select Particular what do you want

(CPU utilization)

↓

Give time Period here [Avg - 5 mints]

threshold type: Static ✓

Anomaly detection

Whenever CPU utilization is: ✓

Greater >	Greater or Equal ≥	Lower <	Lower & Equal ≤

threshold value: 25

↓

Alarm stage triger:

In alarm ✓	OK	insufficient data

↓

Select "SNS topic here"

↓

Some option: Auto scaling & EC2

↓

Create Alarm.

Step④:- if you want to check CPU utilization increasing then use "stress Command" Connect to putty.

for Amazon linux AMI

↓

Sudo su

↓

sudo amazon-linux-extras install epel -y

↓

Sudo yum install stress

↓

Sudo stress --cpu 8 --timeout 20

↓

this command used for seeing spike in CPU utilization graph.

for Ubuntu AMI

↓

Sudo su

↓

apt install stress

↓

sudo stress --cpu 8 --timeout 2

then check in cloudwatch alarms you getting message & emails also when reaching of "threshold value".

* Steps when you getting cloudwatch alarm, Auto scaling is coming :—

→ Create VPC ①, public subnets (2), private subnets (2) Route tables (public & private), internet gate way attach to vpc & NAT gat way creates & attaches in private Route table, internet getway attaches in public route table.

*, launch the instances one or two related to public subnets for VPC ①, & instans any web seaver through Git bash or user data.

*. Creates target group & registed the instances

* Creates load balancer & select availability zones & select target group

↳ check here load distributing or not.

* Create launch configuction the create auto stations Group & attached load balansar & teaget group

↳ Check here auto scaling Cautomatically instances one coming or not) when instance having stoped & terminate position

When CPU utilization reaches to threshold value then get notification so at the time dynamic scaling gives instances so alarm has monitoring :-

↳ 1st Create SNS topic & subscribe it through Email or SMS.

↳ Create cloudwatch dashboard & in alarms for public subnet instance (giving threshold value & SNS topic select here).

↓

After Goto Autoscaling group

↓

Select Autoscaling

↓

Dynamic scaling Policy

↓

Select simple scaling

↓

Create here.

↓

In Autoscaling group

↓

Select Auto Scaling

↓

Scheduled Scaling

↓

Create ~~where~~ here

[if you know when you getting huge traffic at the time you give schedule scaling launch the instance at the time only].

↳ So By the creation of Dynamic scaling policy when you getting in alarm notification it's reaches to threshold value, the policy gives instances.

↳ Alarm can be comes to minimised stage position.

Lambda:- It is a serverless Compute Service that runs your code in response to events & automatically manages the underlying Compute resources for you.

You need not pay Entire running state of instance you will pay only when code is running time. But in EC2 you pay money for Entire instance running time.

↳ Create lambda function to tag all resources in aws ⇒ 1) Created lambda function Runtime-Python & attached role & kept one python code.

types:- Event Pattern & Schedule Pattern

2) Created Rule with cloudwatch Event Bridge rule, API call via cloud trail & Select lambda function.

3) once if instance is coming to running state Automatically tagged the instance. that means cloud Event rule is triggers the lambda function by via cloud trail API & automatically tags the resources.

↳ Create Lambda function to get an event from application & Post into chat tool.

⇒ ~~Chat to~~ chat tool = Slack tool & in this Install incoming webhook & integrate to the channel.

⇒ create one function & some changes you can do like inloming webhook URL.

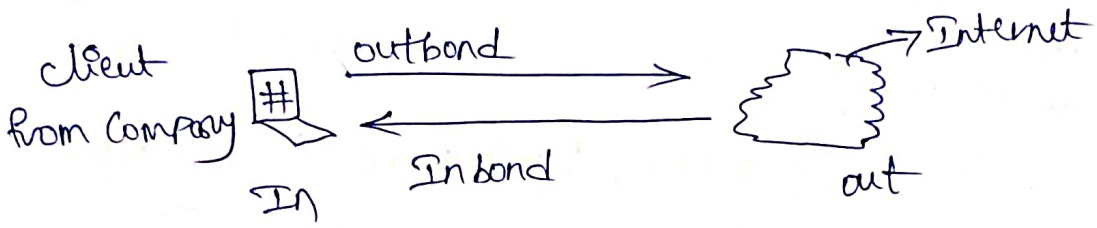
⇒ Create one bucket here lambda function trigger by using Aws S3 bucket kept.

⇒ Automatically messages post into chat tool.

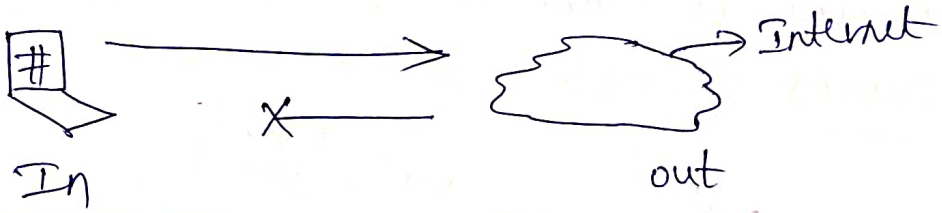
↳ Create lambda function to stop & start EC2 instances & RDS instances.

Security Groups & NACL:-

Stateful firewall:- this means any changes applied to an Incoming rule will be automatically applied to the outgoing rule.



Stateless firewall:- this means any changes applied to an Incoming rule will not be applied to the outgoing rule.



Security group	NACL
*) Stateful *) Instance level applying	*) Stateless *) Subnet level applying.

Security Groups:- If you are open Port 22 in Inbond Rule then easily Connected to server you need not to open Port in outbond Rule. because Port 22 is an existing Port Rule that is why reply is coming.

↳ if you ping www.google.com you are not able to Connect because there is no rule in outbond (in-out).

↳ if you open Port (all traffic) then it's start pinging.

NACL:- By default is NACL section, inbond & outbond rules are all trafic.

if you open Port 22 & 80 In Bond rule not easily Connected to Server because we need to use dynamic in outbond rule (for getting reply), you need not open Port 22 & 80 in outbond rule.

↳ dynamic Port → 32768 - 65535 (or) All trafic

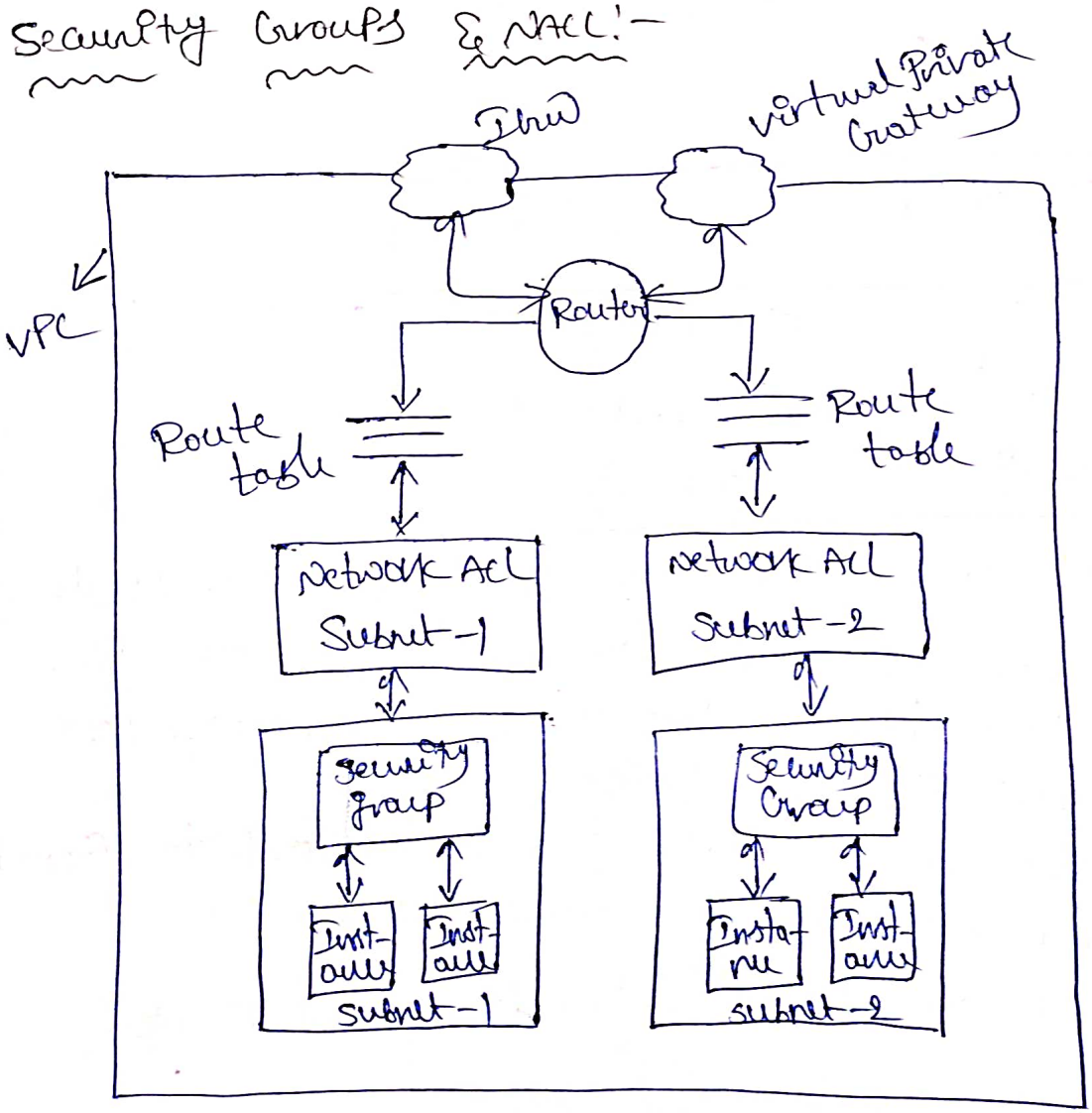
or

Ephemeral Port.

[for getting reply means login page shows on Server through using of Ephemeral Port]

NOTE:- if you are open Port 80 in outbond Rule the another webserver is Connected to this webserver.

Security Groups & NACL:-



↳ Consider two instance with security groups,

i) Subnet webServer instance — webServer Sec. group ①

ii) Subnet DB " " — DB " " ②

if you want access DB Server instance only from web-Server instance, then GOTO DB Server Sec. group ② & change here inbond rule.

mysql/Arora @ Anywhere All traffic All Source IP here [webServer instance Pvt IP or webServer Sec. grp]

That means here only connected from webServer instance [this DB Server instance not able to connect through Putty because here changed the rule].

After then try to Ping,

↳ Ping IP (DBServer Pvt IP)

↓

Starts pinging.

↓

ssh -i .Pem ec2-user@IP (Pvt IP of DB Server)
Ubuntu

↓

then now in DB Server Subnet

↳ if you want access only access from custom IP or Company IP in ssh then you add rule on DB Server Subnet Security.

Snow Ball!- Aws Snowball is a Suite-case sized data migration & edge computing device. It is transfer large amount of data b/w S3 & your ~~onside~~ onSite data Storage location at faster then internet Speed.

Storage classes Available in S3!-

i) S3 Standard
ii) " " -IA
iii) " Intelligent-Tiering
iv) " one Zone - IA
v) " Glacier
vi) " " Deep Archive
vii) " outPosts

Horizontal & vertical Scaling in Aws!-

Horizontal Scaling means that you Scale by adding more EC2 machines into your pool of resources where is vertical Scaling means that you Scale by adding more Power (CPU, RAM) to an existing EC2 instance.

RedShift in Aws!- Aws RedShift is a datawave housing Solution from Aws. RedShift shines in its ability to handle huge volumes of data. this enable you to use your data acquire new insights for you business & Customers.

Availability Zone!-

It is on a datacenter, where we can deploy EC2 instance In N.orginia we have '6' availability Zones highest availabl Zones only in N.oregua.

Region:- It is a geographical location where we cluster data centers.

Status checks in AWS:-

These status checks are "results of automated tests performed by EC2 on every running instance that detects hardware & software issues.

i) System status check:-

which conform that AWS is able to get the network packets to the user's instance.

ii) Instance status check:-

which detects a problem within the EC2 instance.

Tomcat Setup & apps deployed in tomcat server:-

Tomcat installation Procedure file in Devops Demos. (Github)

Step 1:- Create EC2 linux install

↓

cd /opt

↓

amazon-linux-extras [shows all repos]

↓

amazon-linux-extras install java-openjdk-11

(java -version)

↓

download tomcat file → wget "Copy link Address of tomcat"

↓

ls [apache-tomcat-9.0.56.tar.gz]

↓

extract the tomcat file & Create one more directory of tomcat → tar -xvzf /opt/tomcatfile

apache-tomcat-9.0.56.tar.gz

↓

ls [apache.tomcat-9.0.56.tar.gz
apache.tomcat-9.0.56]

↓

mv apache.tomcat-9.0.56 tomcat

↓

cd tomcat

↓

ll [bin, Building.txt, conf, lib, logs, license, temp, webapps, work]

↓

cd bin

ll → startup.sh, shutdown.sh → for linux
shutdown.bat, shutup.bat → for windows

means starts the tomcat in current location → ./startup.sh

Paste PublicIPv4:8090 (open in sec. groups)

Able to see tomcat page.

if you want access in any location (start & stop) the tomcat you need to give linked file for that you need to add path.

echo $PATH

/usr/local/sbin: /usr/local/bin: /usr/bin: /root/bin

ln -s /opt/tomcat/bin/startup.sh /usr/local/bin/tomcatup

'startup' command link with another directory. this is can accessable to in any location

Here you can't login in managed apps on tomcat. So you need to do comment or delete in context.xml.

Search context.xml file in tomcat → find / -name context.xml

shows here.

file1 → opt/tomcat/webapps/host-manager/META-INF/context.xml
file2 → /opt/tomcat/webapps/manager/META-INF/context.xml

↓

change here.

[vi file1 & file2

<!-- valveclassNAME - - - -

- - - - - - - - ->

↓

tomcat down (or) tomcatup

↓

for login credentials in managed apps.

[update details in cd /conf/tomcat-users.xml

directory, lastsee tomcat-details script cricub.

↓

[if ~~any~~ change done u need to done & do tomcat up → tomcatup

↓

GOTO managed apps & login

↓

you are login tomcat application manager.

↓

cd webapps

↓

ll (Show all apps)

↓

you can copy jar or .war file to opt/tomcat/webapps → cp spring.jar /opt/tomcat/webapps

↓

Refresh tomcat Page & spring app is coming.

ECR (Elastic Container Registry :- ECR is a managed container image registry service that is secure, scalable & reliable.

Docker image Send to ECR :-

vi Dockerfile

```
FROM nginx:alpine
COPY . /usr/share/nginx/html
```

↓

:wq!

↓

docker build -t <imagename> .

↓

docker images

Prerequest :-

i) apt install awscli

ii) aws configure (or) Give & attach IAM Role.
to instance because EC2 send image to ECR

iii) Create one Private repos in ECR & see view Push Commands using this command you can Push image from local server to AWS ECR registry

In view Push Commands :-

i) Retrive authentication token & authenticate your Docker client to you registry. → login to ECR from Server

ii) tag your image

iii) Push image from server to ECR.

↓

Refresh the ECR repos in aws & image can

↓

you can pull also image from ECR to Server.

↳ docker Pull repositoryURI .

ECS (Elastic Container Service):- It is highly scalable & fast container management service. You can use it to run, stop & manage containers on a cluster.

There are two types of ~~ECS~~ ECS clusters. Those are

i) Aws ECS fargate

ii) ECS EC2

i) Aws ECS fargate:- It is a compute engine for deploying containers without configuring any server. ~~Here~~ Here you don't need to provision servers, storage & other infrastructure.

The only thing you need to do is provide Aws fargate with a container image & deploying it as a server or single task to ECS.

ii) ECS EC2:- you need to provision EC2 instance that will run Docker act as your container host, & manage storage, firewalls & networking.

The setup time/effort to get an ECS EC2 cluster up & running is about ten times more then running ECS fargate.

Create ECS cluster:-

i) ECS clusters → fargate
→ ECS EC2

ii) Task definitions

iii) Tasks

iv) ECS Services.

I) Creating fargate cluster:-

i) Goto ECS in Console → select Aws ECS

↓

Create cluster

↓

Select { Aws fargate ✓
ec2 linux + networking
ec2 windows + networking

↓

Give clustername:
Create vpc:

↓

Create

Here fargate cluster is created it means at backend cloudformation stack starts.

ii) Task definition :- To Run Docker Container in ECS, you need task definition that defines.

→ which image to use in container
→ launch type (fargate or ec2)
→ networking option (Subnet & Sec group).
→ logging option
→ Command to run
→ & values to use.

↓

Goto Task definition

↓

Select { fargate (Server manage d by Aws)
ec2 (we manage the servers)

↓

Task definition name:
Task Role:
Network mode:

Task size — memory & CPU configuration of container give here.

Add Container here →
- Container name!
- Image — URL give here
- → Port mapping –
- → Health checks –
- → Resource limits –
- → Docker labels –

↓

Create Task definition

iii) Tasks!–

→ if you need to run a single container without load balancing & more advance features, ECS tasks can get the job done.

→ Tasks are designed to help you deploy single container.

→ if you need a more advanced features such as multiple containers, you use the next page type of service which is ECS service.

↓

Select 'task definition' & custo actions → Select 'Run Task'

↓

Launch type → Fargate ✓ / EC2

Task definition: "Select here"

cluster :– vasu cluster "Select here"

no. of task = 1

cluster VPC :

Subnets :-

Sec. groups :- "Add ports here"

Auto assign Public IP :- Enable

↓ Run task

→ Here paste PublicIPv4 of task & see application page.

→ if you want give version for image, then GOTO "create new version" option & give version for image.

iv) ECS service :- To Run a group of containers in ECS with multiple instances & load balancing, you create a service from a task definition & run your container.

GOTO select cluster → under cluster select "service" →

↓

launch type { fargate ✓ / EC2

Task definition :

cluster :- vasu cluster [select here]

Replica :-

no. of tasks :- 2

min. health % :-

max % :-

Deployment :-

cluster VPC :-

Subnets :-

Sec groups :-

load balancer :- "Select here"

Auto scaling :- " "

↓

Create service.

Here Paste the "DNS name" of load balancer & check instance & Autoscaling also.

II) Creating ECS EC2 cluster:-

i) GoTo ESC in Console → Create cluster

↓

Select | Fargate
EC2 (Linux + Network)
EC2 windows + network

↓

Cluster name:-

Instance Configuration | Instance type:- on demand / spot
EC2 instance type:- ____
no. of instance:- 2
EC2 AMI Id:-
EBS Storage
key pair

Networking | VPC:-
Subnet:- only Pvt Subnets select here because Container run only in Pvt. Subnets
Sec. group:-

Container instance IAM Role:-
Cloud watch Container Insights:- Enable / Disable

↓

Create

Here two instance are UP & running in 'ECS Instance' under in cluster & also shows in aws EC2 option.

ii) Task definition:-

Goto 'Task definition'

↓

Select { Fargate / EC2 ✓

↓

Task definition name:-

Task Role:-

Network mode:- Bridge / AWSVPC ✓

Task size:-

Task memory:-

Task CPU:-

Add Container

Container name:-

Image:-

memory limits Hard - 400

" " Soft - 200

[If Container memory occupies more than Hard limit then Container kills similar Soft limit also]

Port mapping — Host Port → EC2 Port; Container Port

Health Checks:-

↓

Create Task definition.

iii) Task:-

Select 'Run task'

↓

lunch type { Fargate / EC2 ✓

Task definition :- "Selecthire"

no. of Tasks = 2

Task group :-

placement templates :-

↓

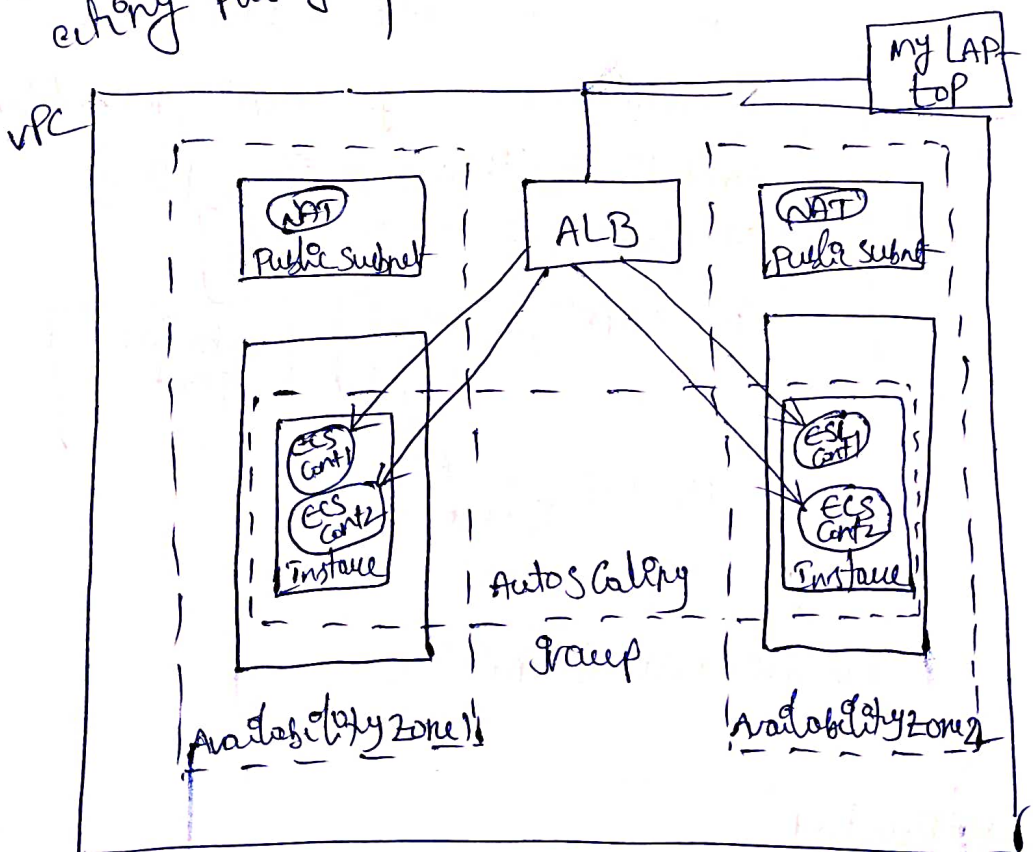
Run task.

Under cluster we are able to see tasks in 'Task' option & EC2 instances will shown in 'ECS instance'

↳ Paste the PublicIPV4 in URL then not able to see application page because these are in Pvt subnets. So inorder to access these we login through bastion host (Public Subnet).

↓

In bastion host after connecting putty | curl "Pvt IP of Pvt-Subnet Instance"



Aws Systems manager(SSM):-

It is Aws servile that you can use to view & Control your infrastructure on Aws.

Run Command :- It allows to automate common administrative tasks & Perform one-time Configuration changes at scale.

Task:- I have 1000 servers & I need to install Patch on 780 server (780 Prod & remaining 220 Test).

By using ASM servile, we can Create Install & Updating anything on different no. of servers at time.

Step①:- Create one role EC2 to amazon SSM full access

Step②:- Creat '3' instances (Linux) & attach this role to '3' instances. if you are not attach the role then not able to see instance in SSM manager.

↳ while creation of '3' instance you should give tags

name:- Server1, Server2, Server3
env - Prod

↳ ps -ef Shows all running servies on server, if you want restart,

→ /usr/bin/amazon-SSM-agent restart

after attaching the role to instance then able to See in 'SSM'

Step③:- I want to install nginx web server on three instance at time only.

Script:-

```
#!/bin/bash
yum update -y
yum install nginx -y
service nginx start
yum install tree -y
```

GOTO Run Command

↓

Select Command document :- all shows,

Aws Run shell script,

Docker

Ansible Playbooks.

↓

Select target :

manually selecting instances

specifying a tag ✓

Tag NAME	Tag value
env	Prod

whatever instance are registerd for particular tag. those all instances comes here.

Commands :- "Script Posthere"

↓

Run

↓

now paste the Public IPv4 of server ① & ② & ③, thapable to nginx server.

Stop & start RDS instance using AWS lambda:-

↳ Create database & while creation database instance you must give tag names.

Tag key	value
DEV-TEST	Auto-Shutdown

↳ Creat IAM policy for 'rdsstopstart' & here give JSon format script.

↳ Create a Role for lambda-rdsstopstart

↓

name → rdsLambda

↳ Create "2" AWS lambda functions for stop & start the RDS instances.

→ first Create one function for stop instances,

function name :- StopRds

Runtime - Python 3.7

execution role :-

↓

Create function

under the function, we have option Configuration in this Configuration add the key & value of RDS instance.

Environment variable.
- key - DEVTEST
- Region - US-East-1
- value - Auto-Shutdown

→ Illy Create function for start instance.

↳ for both stop & start RDS instance. here we need to delete sample code & Paste the Python boto code for stop & start RDS.

↓

Choose "Test"

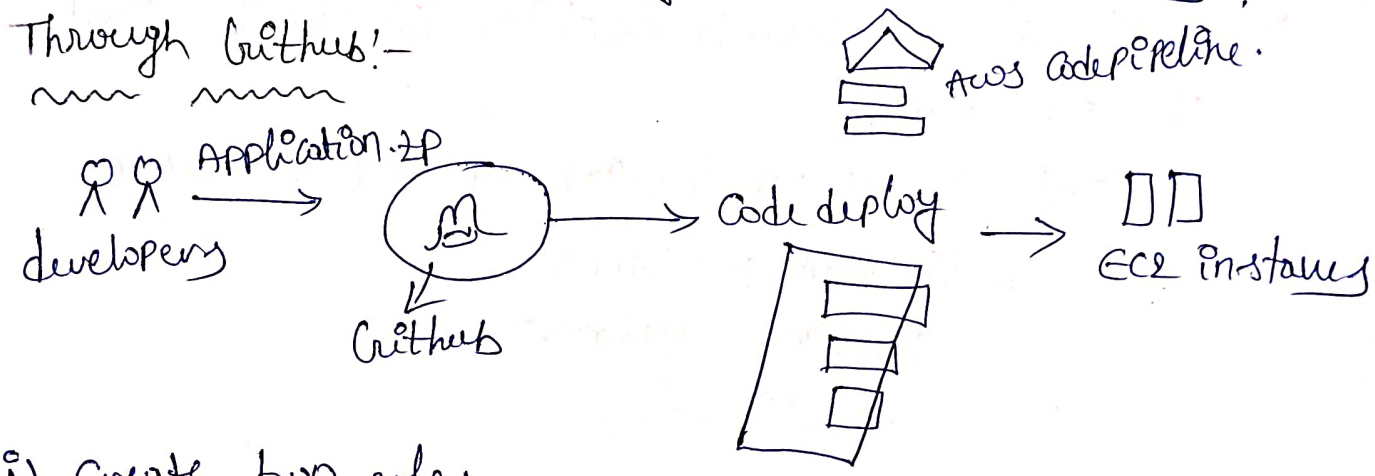
↓

See execution results

↳ Create amazon Event bridge pattern for stop & start RDS instance based on scheduling.

Aws Codedeploy :- It is a fully managed deployment service that automates software deployment to variety of Computer services such as EC2, Awsfargate, on-Premises, it makes easily rapidly releases new features.

Through Github:-



i) Create two roles

→ EC2 — S3, AWSEC2 & Codedeploy Access

→ Codedeploy role, Codedeploy — Aws Codedeploy access

ii) Create 4 instances & install any web server through userdata & Give tags.

iii) Goto Codedeploy option,

↳ Create "application"

↓

application name:

Compute plateform: EC2 | on-Premises ✓

Aws ECS

Aws lambda

↓

Create

iv) Create "deployment Group"

↓

deployment group name:

Service role : "select 2nd role"

deployment type :- Inplace ✓

Blue/green

Inplace :- if something is installed in EC2 instances then updation Github new version is coming. that new version installed in that instance only.

Blue/green :- if something is installed in ec2 instance after if new version is coming then the new version is installing in new instances & previous instance will deletes.

Environment configuration:

Amazon EC2 Autoscaling group

" " Instances ✓ — specify & Give tags here for EC2 instance

onPremises "

Deployment settings:

one at atime

Half At "

All " ✓

Load balancer: ——

↓

Create deployment group

v) Goto Codepipeline

↓

pipeline name: ✓

Service role: new / Existing

Artifact stores: Github

[Entry repository & branch]

Build Provider:- Code build / Jenkins (Skip) step

deploy Provider:- Code deploy

application name:- "selecthire"

deployment name:- " "

↓

Create

vi) Paste 4 instances IPv4 in URL then able to see "Homepage of application for Github code application". if something modification is done in github then build triggers & modification page coming.

Through S3:-

i) Roles

ii) Create one ec2 instance & install webserver through userdata & attach IST role to this instance.

iii) Create one bucket in S3, & enable the option "keep all version of an object same bucket"

iv) Create application & deployment group

v) Create Code-pipeline

↓

Here in this, Source Provider - Amazon S3

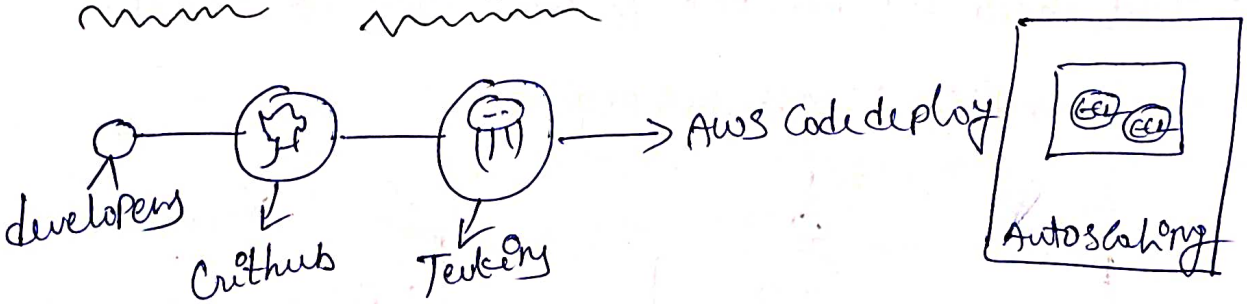
Give bucket = "bucketnamehere"

S3 object key = application.zip

vi) upload any application.zip file in S3 bucket from Server. then see homepage application through IPv4.

if you want any modification then
↳ unzip application.zip
↳ ls (shows three).
delet zip file, you can modify in index.html file again after modifying this you can zip the application.
↳ zip -r application.zip .
vii) Again you can upload new zip file to s3 then see changes in application.

Aws Code deployment using Code deploy service - In place Deployment using Jenkins :-



i) Create '2' roles
ii) Create 4 instances & install web server & attach IAM role
iii) Create application & deployment group(Inplace).
iv) Create load balancer & Autoscaling for '4' Ec2 instances
v) Create Job freestyle Project & Give Paste Repos URL of Git. In Post build Action.
Deply an application to Awscodedeploy.
Aws code deploy name:
" deployment group:
" " Config: one time / Half time / all time
" region:
s3 Bucket: "Bucketname"

S3 Prefix: "foldername in bucket"

Accesskey:

Secretkey:

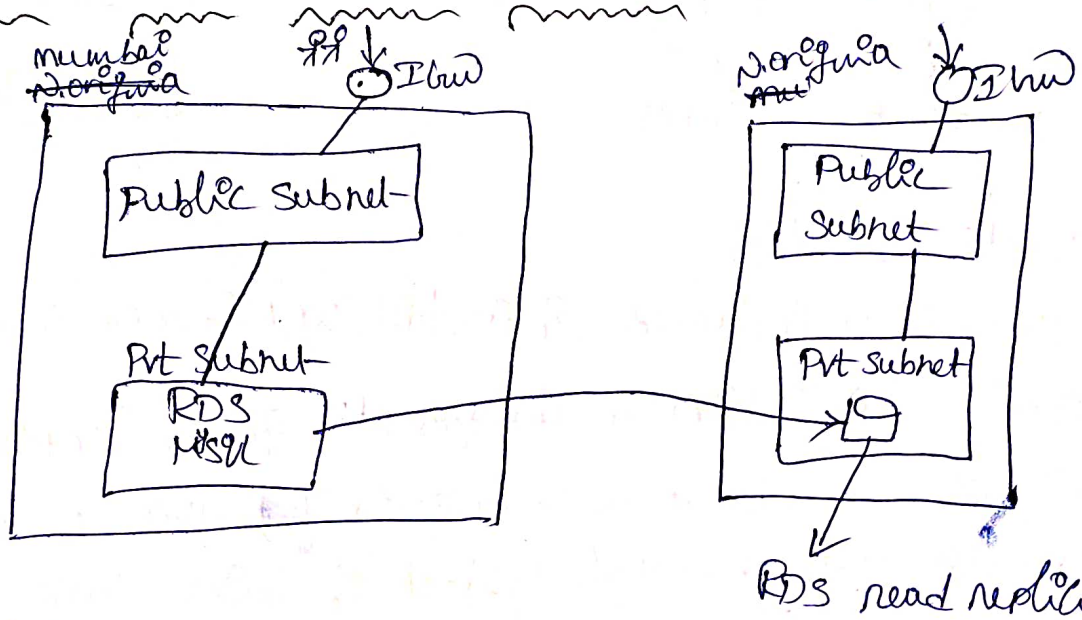
vi) check in AWS Code-deploy option in console check & observe Inplace deployment how there.

Blue-green deployment:-

Illy Inplace deployment way, but select Blue-green deployment type.

↳ finally check in AWS Code deploy option in console & observe ~~bl~~ blue-green deployment how there. Initial instances removed & new instances added.

RDS Read Replica Across AWS region:-



↳ Create a DB subnet group in mumbai & n.virginia.

↳ Create a RDS instance through console in mumbai.

↳ Goto action in RDS select,

↓

Create read replica DB instance

↓

Region name:
Availability Zone: "Select here"
Subnet group: " "
Source - RDS [mydbinstance]
DB instance identifier - read-replica db instance

↓

Create read replica

↳ Here in mumbai region, you can create DB & tables in mysql after installing db mysql in Public Subnet instance.

↳ In N.virginia create Sec. group for mysql & add that Sec. group in "read replica db instance" in Sec. group, IP you can keep Public Subnet IP because you connecting from Public Subnet.

↳ Here in N.virginia login to mysql db after installing db mysql in Public Subnet instance

↳ check here in Server all updated data coming in N.virginia whatever data you modified in mumbai region.

↓

mysql -h endpoint -P port -u master -p

↓

mysql> Show databases;

[mydb
mysql]

↓

mysql> use mydb;

[Database changed]

mysql > CREATE DATABASE vasu;

↓

mysql > Show databases;

[mydb, mysql, vasu]

↓

mysql > SHOW TABLES;

(tables shows here)

↓

mysql > CREATE TABLE vasu1 (name VARCHAR(20), owner

↙

tablename

VARCHAR(20), species VARCHAR(20), sex CHAR(1), birth DATE,

death DATE);

↓

mysql > SHOW TABLES;

Tables in VASU
vasu1

↓

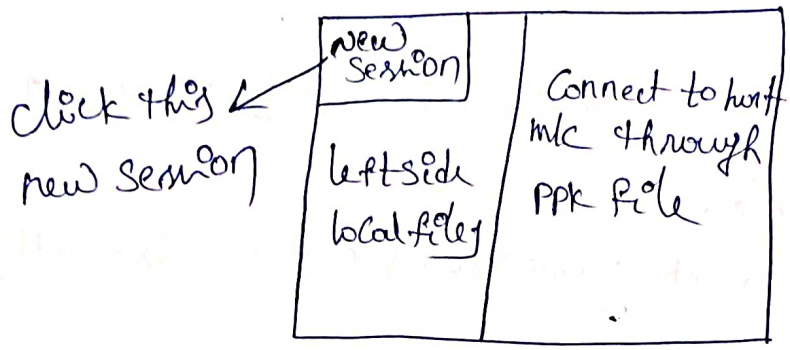
mysql > DESCRIBE vasu;

field	Type	NULL	key	default	Extra
name	varchar(20)	YES		NULL	
owner	"	YES		NULL	
species	"	YES		NULL	
sex	char(1)	YES		NULL	
birth	date	YES		NULL	
death	date	YES			

WinSCP:-

The main function of this, transfer the files b/w local system to remote system.

Open the winscp,



Here leftside local files drag to right side after Connecting of host m/c. then see in Server local files Coming or not for that Server.

---

if system reboot the mount disk not need to unmount, for that update in file,

vi /etc/fstab

↓

At last line,

/dw/xvdf1 /tmp/disk2 xfs defaults 0 0

↓

:wq!

for unmount disk,

↳ umount /tmp/disk2

here check mounting Path is removed

for Again mount disk,

↳ mount -a

files /dw/xvdf
mefs + xfs /dw/xvdf

Allocat Some marg to Particular file,

↳ fallocate -l 3G test.txt

Rout53:- Rout53 is a DNS Servile Provided. ~~by Aws~~ It's cost effective Servile for managing & maintaining domains.

↳ Create one EC2 instance & install nginx webserver.

1st way:-

i) Create one domain in Godaddy when it's means after Creating Godaddy A/c.

ii) Here you getting default nameservers (NS) in Godaddy domain.

iii) Create Hosted Zone for Particular domain name in Public type in Aws Route53 (Public or Private) & add these nameservers (NS) in Godaddy domain instead of default nameservers (NS).

iv) 'Create a record' in hosted & Give name, A type (IPV4) or Cname or AA IPV6 type Record.

Cname ↙ [redirect to A type record]

2nd way:-

i) Create one domain in Aws under 'domain registration' [.com, .xyz, .net, .org...]

ii) Goto Hosted Zone & shows here domain registration

iii) Goto that domain registration & 'Create records'.

iv) Here using Alies with help of ~~Cnames~~ Cnames.

ACM!-

↳ Create a certificate for particular domain name through DNS validation.

↳ here Add route53 record to particular domain or

↳ Goto route53 & create record → add the certificate in C-name record

↳ create one instance & Add to the load balancer.

↳ Select Add listener https & select ACM certificate while creation of load balancer.

↳ Copy the DNS loadbalancer & Goto the route53 & add in the 'A type record' & select alies & paste the 'DNS' load balancer in record.

↳ see Domain name in URL check with http & https Connection start word of domain.

↳ if http is not require, then Again Goto listeners in loadbalancer option & select http option & select redirects to the https.

↳ See & check in URL the only https action works.

SSL Certificate Providers!-

1) ACM
2) cloud-flare.com
3) letsEncrift

How to launch windows Server in AWS!-

Goto AWS Console → Instances → launch instances

↓

1) Choose AMI
2) Choose Instance type
3) Configure details & Add storage
4) Configure Sec. group

[RDP – TCP – 3389 – Anywhere]

↓

Review & launch.

whatever Pemkey you launching that is Encrypting Password so you need to decrypting the Password.

Goto "Connect" & under 'RDP client', [username, IPv4 shown].

1) Download remote desktop file(rdf) for connecting to window Server.
2) for getting decrypting Password, under RDP client Select 'Get Password' & then Paste Pemkey & Select the decrypting option then Password to be generated.
3) without having rdf file you can having a another way also. that is 'Remote Desktop Connection' in windows OS.

Connection!-

i) Open RDP in windows → Enter 'Public IPv4 & username' → Enter 'Password' → you are get in to window server.

ii) open 'double click that file → Enter Passwd (whatever you generate decrypted format) → ok → you in window Server.