CFA

7-8-2023

Docker
Kubernetes
Automation
Kubernetes Package Manager - Helm
GitOps.

DevOps : Automating SDLC
Apache Spark - Big Data Architecture
MLOps : pipeline where data go through manage, build, deploy.

Modules:
M1 : Cloud computing Overview
M2 : Cloud Economics and billing Read own
M3 : AWS Global Infrastructure Overview
M4 : AWS Cloud Security
M5 : Networking and content delivery
M6 : Compute EC2
M7 : Storage EBS, S3
M8 : Database
M9 : Cloud Architecture
M10. Automatic scaling and monitoring

Module 1 9-8-2023

- Cloud Computing
- Hardware or Traditional disadvantages
- Software Advantages

Cloud Service Models :
- IaaS, PaaS, SaaS
- XaaS - anything as a service

Each service/deployment models have diff. level of control, flexibility, management
- multi tenancy

Deployment models:
Cloud Hybrid Private (on-premises)

CFA LAB 1 11-8-2023

- sbin : holds executables that are executed by administrator
- var : holds log and spool data of complete system
- opt : used for installing third party applicat^n's which has its own configuration.
- mnt : interact with external devices
- dev : list of all usb, bluetooth devices
- etc : holds folders, subfolders and files req. to configure system and software

to change hostname:
- sudo vi /etc/hostname
  or
  sudo hostname new_host_name

- sudo vi /etc/hosts => edit old hostname
then restart
ip a -> ip address

lamp stact - linux apache mysql php
mean, mern.

- To deploy static website

- sudo apt-get install apache2 downloads .deb
  sudo apt-get update
  (sudo rm -rf /etc/apt/sources.list.d/jenkins.list)
  → to remove repolist

systemctl, service
⇒ utilities used to start, stop, restart, enable, disable a service
when change is done to particular servi

sudo systemctl start apache2
sudo systemctl status apache2 - to check whether apache2 is running/active
or
sudo service apache2 status
in browser : http://localhost:80 or
http://_____.
ip address

⇒ sudo vi /etc/apache2/sites-available/000-default.conf
in DocumentRoot _____ html file path
/var/www/html

download free html5
sudo cp Downloads/html5up-paradigm-shift.zip .
src dstn or current directory

to extract zip file
- sudo mkdir myapp
- sudo cp _____.zip myapp/. copy zip file to myapp folder
  cd myapp
- sudo unzip ____.zip
- sudo rm -rf ____.zip. outside myapp directory

Copy everything from myapp to /var/www/html
- sudo cp -rf myapp/ /var/www/html/.

cd /var/www/html

x =

change DocumentRoot to /var/www/html/myapp
sudo service apache2 restart
in browser restart
(sudo ufw allow port no) if firewall blocks

php based application :

- sudo apt-get install php libapache2-mod-php
  github.com/sreepathysois/basic-php-website => copy code file
- sudo git clone _____ url
- ls => "basic-php-website" folder is created
  book album
- sudo cp -rf basic-php-website/ /var/www/html/.
  sudo mv basic-php-website/ bookalbum

sudo vi /etc/apache2/sites-available/000-default.conf
sudo systemctl restart apache2

if there are multiple index files
sudo vi /etc/apache2/mods-enabled/dir.conf
=> DirectoryIndex index.php index.html
change priority

Restart

16-8-2023

Module 1

Section 2 : Advantages of Cloud Computing

CapEx : Capital required to acquire, upgrade & maintain physical assets

1. Trade capital expenditure for variable expense
2. Massive economies of scale
   - (We can get lower price for customers becoz of massive customers.) Because of aggregate usage from all customers, AWS can achieve higher economies of scale & pass savings to customers
3. Stop guessing capacity : ∵ it may lead to
   cloud helps to scale — overestimated server capacity
   On-demand — underestimated "
4. Increase speed and agility : in terms of organizat^{n} we can quickly deploy resources required within mins or by click of 3 buttons. (mins between wanting & having resources). It reduces time
5. Stop spending money on running and maintaining data centres.
   - focus on projects that differentiate your business instead of worrying about infrastructure, building, maintaining IT resources
6. Go global in minutes
   - data centres are set up near to customers which is achieved by AWS, who can quickly launch your applicat^{n} so that customers don't feel latency in accessing services

Section : 3 Introduction to AWS

What are web services ?

Eg: Going to payment gateway from other applicat^{n} & giving result back

Interface definition file

What is AWS?

* AWS works together like building blocks

Services covered in this course

- Compute Services
- Storage
- Database

3 Ways to interact with AWS

- AWS Management Console
- AWS Command line Interface (CLI)
- Software Development Kit's (SDKs)

LAB 2 17-8-2023

Deploy dynamic website

lamp stack : linux apache2 mysql php

github.com/sreepathysois/phpmysql-app

Sudo apt-get install apache2 mysql-server mysql-client php libapache2-mod-php php-mysql

sudo git clone https://github.com/sreepathysois/phpmysql-app.git

cd ~~my~~ phpmysql-app.

Create folder

sudo mkdir /var/www/html/ecommerce

cd php/online-shoppingsystem

sudo cp -rf * /var/www/html/ecommerce/.

cd /var/www/html/ecommerce/
sudo vi /etc/apache2/sites-available/000-default.conf
→ DocumentRoot /var/www/html/ecommerce
sudo systemctl restart apache2

setting up database :
sudo systemctl status mysql
create db for ecommerce
create user and password, grant permission to ecommerce db.

- sudo mysql -u root -p ↵ (-u user root, -p pass) => to connect mysql db

mysql >> show databases;
> Create database ecommerce;
> show databases;

Create user who can access this db from local host
> CREATE USER 'msis'@'localhost' IDENTIFIED WITH mysql_native_password (remost host or %) BY 'Msis@123';
Grant permission only to ecommerce db (to all db * *)
> GRANT ALL PRIVILEGES ON ecommerce.* TO 'msis'@'localhost';
go into db
> use ecommerce; database changed
> show tables; empty
create tables and populate data req. for applicat^n
> exit
cd /var/www/html/ecommerce/database
ls
vi Onlineshop.sql
- sudo mysql -u msis -p (Msis@123) password
mysql > show databases;

> use ecommerce;
> show tables;
> source onlineshop.sql;
Queries in ↑ will be executed
> show tables;
> select * from admin_info; > select * from brands
>exit

Connect application and db
cd /var/www/html/ecommerce
- sudo vi db.php
server name = "localhost";
username = "msis";
password = "Msis@123";
db = "ecommerce";
- sudo systemctl restart apache2

Module 3 AWS Global Infrastructure Overview
Section 1: AWS Global Infrastructure

- AWS Regions: It is a geographical area
In order to achieve fault tolerance, scalability & stability there are diff regions set, which are isolated with each other.
- Each region provide full redundancy and connectivity to the network.
- each region has 2 or more availability zones
* Selecting a Region.
* Availability Zones ⇒ to make highly available, scalable, fault tolerance
Region
3-4 Availability zone ⇒ Each AZ is fully isolated Partitions of AWS infrastructure
↓
~~each AZ has 3-4 datacenter~~
↓
each has 100-1k severs

- AWS recommends replicating data and resources across availability zones for resiliency

* AWS datacentres
  We cannot know which data centre our resources are provisioned
  - Point of presence - 187
  - Edge locations - 176
    Regional edge caches - 11

AWS Infrastructure Features:
- Elasticity & Scalability
- Fault tolerance
- High availability

Create EC2 instance. Launch instance
Name & tags : lamp
Applicat^n & OS images : ubuntu
Instance type : t2.micro
key pair login : create new key pair
keypair name: CDCLabTest, RSA, .pem Create key pair
Network settings : check all
Configure storage : 8 GB gp2 => Launch

- cd downloads => sudo chmod 400 CDCLabTest.pem
  name of Private key
In EC2 instance
Connect => SSH copy: ssh -i _____
ubuntu@ ____ $ ip a
exit

LAB 3 23-08-2023

Bullet Proof Architecture for hosting static website

Object storage : which is accessible through url
↳ eg Amazon S3 (simple storage service)
- only host static website

Cloud front distribution:
- It distributes the website / content throughout the globe

US-east-1
Select S3 service
- Create bucket (with domain name)
Bucket name : msiscdc.com ⇒ no 2 buckets can have same domain name, it should be globally unique. (no capital letters, special characters)
region - us-East
Obj Ownership
(ACL - Access control list ⇒ policy that tells who can view, update, delete data)
ACL disabled
block all public access → uncheck
Bucket versioning → disable
default encryption
⇒ to tell bucket to act as a server
go to properties
- edit static website hosting ⇒ enable
- index document ⇒ index.html
- error document ⇒ optional

⇒ to make objects public so that normal customer can use
- select objects
in action tab → make public using ACL

- to enable ACL
  permissions → obj ownership (edit) ⇒ enable ACL

If availability zones is not there to select, then it is global service.

Cloud Front Distribution:
create " " "
- origin domain - it detects
- protocol - http only
origin path - to specify if website is in other folder
origin shield - No. (ddos attack)

default cache behaviour
invalidation ⇒ for everytime given edge location comes to origin server and takes the update and serve the users
- Http & Https
- Get & head
- Default root object ⇒ index.html
(cloud pin) to check from where we got the website

Route 53 service : It translates domain name to their corresponding IP addresses.
- create hosted zone
  domain name ⇒ cdcmsis.com
- create record ← what should be done when clicked
  alias → cloud front distribution
  paste : distribution domain name

for subdomain
create bucket : name ⇒ www.cdcmsis.com
go to bucket → properties → static website hosting → redirect requests for an object

host name → cdcmsis.com
protocol - http

Route 53
- Create record → www.cdcmsis.com
  Record type : C Name
  value : website link from bucket
  www.cdcmsis.com

LAB 4 Hosting dynamic website in aws 24-08-2023
Go to EC2 service
launch instance:
- name & tags (tag is local identifier & it is in webserver key value pair form)
- Applicat^n & OS images (AMI - Amazon Machine Image)
- ubuntu 22.04
- instance type → t2.micro, size
  family name generation
- key pair login : select vockey
  → download pem key for ubuntu, ppk for windows
- Network settings:
  security group → set of rules that determine what kind of traffic allowed into system and out of system
  inbound rule
  outbound rules → it is allowed to anywhere traffic ie 0.0.0.0/0
- allow SSH
- allow HTTP
- Configure storage (EBS - Elastic block storage service)
  - 8 - gp2

In instance connect and copy (ssh -i)

In terminal

cd Downloads

=> sudo chmod 400 labsuser.pem

=> ssh -i labsuser.pem (key) ubuntu@ _____ (Public dns from instance)

ubuntu@ _____

=> sudo apt-get update

=> sudo apt-get install apache2 libapache2-mod-php php php-mysql mysql-sever mysql-client

=> sudo systemctl status apache2

check in browser using public dns

to copy data from local system to cloud securely

in new terminal

=> cd /var/www/html => ls

=> scp -i /home/msis/Downloads/labsuser.pem -r (copy recursively) ecommerce (source file to copy) ubuntu@ _____ (cloud user name) :/home/ubuntu/. (destination, copy here)

(key pair)

- again in cloud vm

sudo cp -rf ecommerce /var/www/html/.

=> cd /var/www/html => ls (ecommerce)

=> sudo vi /etc/apache2/sites-available/000-default.conf

DocumentRoot /var/www/html/ecommerce

- sudo systemctl restart apache2

Setting up database:

=> similar to dynamic website hosting.

=> This process of hosting website is not good becoz it has unmanaged database. and both application and database is at same place

(not managed by cloud provider)

Module : 7 Storage 28-08-2023

- EBS
- S3
- EFS
- S3 Glacier - long time data archiving service

Section 1 : Elastic Block Storage

- It acts as virtual hard disk of a system
- Its detachable, durable and persistent block storage for EC2 volumes are replicated within some Availability zone (non-volatile) It holds data even if you turn off system & turn on after long time

AWS Storage Options :

Block storage	v/s	Object Storage
- change one block (piece of file) that contains the character		- entire file has to be updated - If huge data has to be stored stored

EBS : process data at low latency, high performance
- enables to create individual storage volume and attach them to Amazon EC2 instance
- To replicate volume, Snapshots can be used
- It can be backed up automatically to Amazon S3 through Snapshots, can be copied from one region to another

Uses include
- Boot volume (to store OS) and storage for Amazon EC2 instance
- Data storage for File System
- Database
- Enterprise application.

EBS Volume Types :
- Volumes can be reconfigured without stopping EC2
(SSD drives is mandatory for Boot volume)

EBS features
- Snapshots
- Encryption
- Elasticity ⇒ increase capacity & change types (ie HDD, SSD)

Volumes, IOPS and pricing
- General purpose SSD
- Magnetic
- Provisioned IOPS SSD

Data Transfer. (EC2 ⇒ unmanaged service)

Section 2 : Amazon Simple Storage Service (S3)
(It is abstract service : we don't have any knowledge of infrastructure or how it works)
- Object level storage.
- to access data globally with end point this service can be used
- bucket is resource which holds objects
- data is stored as objects in buckets.
- designed for 11 9's durability : data is replicated in 3 availability zones in a particular region
- It is a managed service (ie customer need not manage any infrastructure, storage, scaling)
- unlimited storage (single object limited to 5TB)

Data at rest Data in transit

30-8-2023

S3 storage classes

1. Standard : 4 9's
   => offers high availability, durability, performance
   applicat^n has min downtime
   redundancy or no failure
   => choose if you have frequently accessing data and to process
   => best fit for designing CDN
   => has rapid processing time.

2. Infrequent Access :
   - infrequently access data, 3 9's availability
   - offers high durability, low latency, high throughput
   - can be used for long term storage, backups, diaster recovery
   - low cost, high performance

3. One-zone Infrequent access
   - less frequently access, but requires rapid processing time
   - no need of high durability, even if we loose data no worries
   - used to store metadata of data
   - store only in one-zone

4. Intelligent Tiering :
   - don't know access pattern of data
   - automatically moves data from one storage class to another (i.e. standard or infrequent access) to optimize cost.
   - pricing : according to when data is accessed

5 Glacier
- low cost, secure, long time data archiving
- no worry of speed
3 types of retrieval methods.
- S3 bucket life cycle management → automatically move data on schedule (one method to upload data to glacier) first 30 days standard
next 6 months one zone infrequent
next move to glacier
- other method is directly upload to Glacier

6 Glacier Deep Archieve
- long term data preservation. storing 8-10 years
- compliance or regulatory data to be stored
- Both offers high availability, low cost.
(Glacier & deep archive

S3 bucket URLs
- to deploy static website → virtual hosted style url
bucketname regioncode
- to objects → path-style URL
regioncode bucketname

Access data anywhere:
1. Management console 2. AWS command line interface
3. SDKs

Common Usecases.
- Backup & storage
- application hosting
- Media hosting
- Software delivery

Storage Pricing:
depends on storage class requests, data transfers

Module 6 : Compute
Section 1 : AWS Compute Services

- EC2 : resizable compute resource
- EC2 Auto scaling : automatically scale EC2 instance

Categorizing compute services
scalability in terms of instances, servers
EC2 : Elastic Compute cloud
- replacement for on-premise traditional desktop servers
resources to compute (ie require CPU, memory

Nine key decisions to launch EC2
1. Select an AMI (Amazon Machine Image)
- it generally contains OS and few pre-installed softwares require to boot up the instance.
- AMI choices → Quick start
My AMI
Market place AMI
Community AMI

- Create my own AMI

2 Select an instance type
determines : memory (RAM)
Processing power (CPU)
Disk space and disk type (storage)
Network performance
Instance type categories
Instance type naming and size
Eg: t3.large
t → family name
3 → generation number
large is size

NACL - Netw Access Control list
- It acts a security layer to your VPC
- It acts as a firewall that controls traffic in & out of one or more subnets

31-8-2023

LAB : 5

UseCase

1. Database should be kept private
2. Create isolated environment for diff. resources/services

It is a secure, isolated private cloud hosted within VPC - Virtual Private Cloud Public cloud

- You can create your own isolated space / resources & it cannot be accessed outside VPC unless it is permitted

NAT gateway : Internet launched in public subnet, and is accessed by private subnet whenever required through NAT gateway

public subnet : has access to internet gateway and internet outside VPC

private subnet - doesn't have access to internet gateway & internet outside VPC

Route Table : controls traffic / determines the path the resource follow to communicate with other

- If destination resource is within same VPC - local
- If destination resource is outside and it should go through i. internet gateway - public subnet
  ii. NAT gateway - private subnet

Multi AZ - to span resources across multiple availability zone

Creating VPC :

In management console

VPC → Create VPC, VPC and more

Name tag : myecomvpc , CIDR IPv4 : 10.0.0.0/16
Tenancy : select default
(dedicated - only by you)
No. of availability zones : 2
No. of public, private subnet : 2, 2
Customize subnets
public 1 : 10.0.0.0/24
2 : 10.0.2.0/24
private 1 : 10.0.1.0/24
2 : 10.0.3.0/24
NAT gateway : in one AZ
Create VPC

EC2 service — install mysql-client
Launch instance
ecomwebserver
→ network setting
VPC - myecomvpc
subnet - public 1
Auto assign public IP : enable
(whether to create IP address)
security type - ssh
Add security rule : type : HTTP source : 0.0.0.0/0
(Launch)

Launch instance
dBserver — install mysql-server, php-mysql
Network setting :
VPC - myecomvpc
subnet - private 1
Auto assign public IP : disable
Add security rule : type : mySQL Aurora
source : (ecomwebserver security group rule id)
is only allowed

how to do ssh to dataserver database server
-> by ecomwebserver

ecomwebserver connect
-> ssh -i labsuser.pem ubuntu@ _____
Public IP of ecomwebser

to copy labsuser.pem key from local to ecomwebserve
scp -i labsuser.pem labsuser.pem ubuntu@ _____
:/home/ubuntu/ (in local terminal)

before going to dBserver
Come to ubuntu@ _____ ↑ copy ecommerce install apache2 and host until before dB
ssh to database server and next connect to dBserver
ssh -i labsuser.pem ubuntu@ 10.0.1.24
Private IP of dbserver

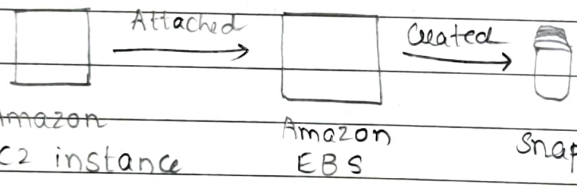
In dbserver
Create user and database, grant all permission to database.
change bind address, so that it listens from anywhere : cd /etc/mysql/mysql.conf.d
sudo vi mysqld.conf
bind address = 0.0.0.0
restart mysql

Come to ecomwebserver
terminal => sudo mysql -u msis -h 10.0.1.24 -P3306 -p
Enters mysql >
Here populate the database :
use ecommerce ;
source onlineshop.sql ;
Connect dB & applicatn :
cd /var/www/html/ecommerce
- sudo vi db.php
servername = "%" ;
username = "msis" ;
password = "Msis@123" ;
db = "ecommerce" ;
restart apache2

Usecase 6 02-09-2023

Change the region to cheaper one
- This can be done by taking snapshot and replicating it into another region

Launch EC2 instance
myinstance
network - default (no vpc)
Configure storage : 8 gp2
launch an instance
(In Windows : download putty for windows
download ppk key from AWS details
Connection : SSH → Auth → Credentials → Select downloaded key)



06-09-2023

to add additional volume / storage to above instance

Volumes tab in EC2 instances
create volume
gp2 , size : 1 GB
Availability zone : Similar to above zone
Add tag : Name ← key
Myadditional Vol ← value
create

we can't mount volume if both are in diff. Az.

to use this volume attach to EC2 instance
in Actions → Attach Volume.
select instance
device name : /dev/sdf
Attach volume

Formatting and mounting : check whether OS detects the newly created volume

in terminal
cd Downloads
ssh -i labsuser.pem ubuntu@ ____ (public ip address)

to know what diff external and internal storage to system => df -h

to detect additional volume format
to see format of
sudo file -s /dev/x
sudo file -s /dev/xvdf
if it is : data => we have to format it to ext

to format : sudo mkfs -t ext3 (type of format) /dev/xvdf (on what storage)

sudo file -s /dev/xvdf
df -h still it will be not detected
Now we have to mount it
sudo mkdir /mnt/mydata in mnt directory create mydata folder & mount volume there

sudo mount /dev/xvdf (source) /mnt/mydata (dest)

(df -h => /dev/xvdf mounted on /mnt/mydata
cd /mnt/mydata
create a folder and create a file in that

To take snapshots
select volume => actions → create snapshot
descript^n : MyVolsnapshot
Tags : key : name
value - my volsnapshot

In Snapshots tab
select volume -> Actions → Create volume
gp2 ⇒ 1 ⇒ AZ
Tag: Key: Name
Value: Restored volume

Attach this volume to EC2 instance

in teminal:
df -h
sudo file -s /dev/xvdg see in attach volume what is it
We need not format ∵ we have snapshot of formatted storage to mount
sudo mkdir /mnt/restoreddata
sudo mount /dev/xvdg /mnt/restoreddata

Check whether folder created in first additional volume is copied in this restored volume

13-9-2023

(Try Elastic BeanStalk)

Select an instance type: based on usecase
- General purpose
- Compute optimized
- Memory
- Accelerated computing
- Storage

Instance types : Networking & features
cluster placement group
interdependent instances has to placed closer to each other or in some server.

3. Specify network settings
4. Attach IAM role (Optional) owner to speak to
gives temporary privileges to the services and also what actions can be performed without sharing credentials
(ie to read/access data from those service)

5. User data script (optional)
Write shell script to host website
ie ssh, install, git clone, copy to var/www/, restart apache2) Assignment

6. Specify Storage if we restart instance Public IP address changes, also the server but if we reboot there is no change
- configure root volume
- attach additional storage volumes.

Storage options :
- Amazon Elastic Block storage (EBS) Virtual storage
- Amazon EC2 Instance Store physical storage of server where instance is running
Other : Amazon EFS, S3

7. Add tags.
8. Security group :
- determines the traffic allowed inside and outside the instance.

9. Identify or create the key pair
remote desktop protocol

We use managed database 25-9-2023

Cloud provider manages everything
you only need to focus on your application
and managing it
you just need to tell cloud provider
what database, version, storage you want
Example for managed service is S3, RDS

VPC
EC2 - only for webserver
RDS - for database

Subnet group - group private subnets
create VPC similar to previous

EC2
name : mywebserver
edit n/w settings & choose VPC you created
public subnet.
Similar to previous.

do ssh in terminal

sudo apt-get update && sudo apt-get install
apache2 libapache2-mod-php php php-mysql
mysql-client.

sudo git clone https://github.com/srepathysois/
phpmysql-app.git

php mysq-app/php/online -

sudo cp -rf * /var/www/html/.
sudo rm -rf index.html
restart apache2

Create managed database server using RDS.

Create db.
- standard create
engine type : mysql Aurora - AWS implemented of Mysql & postgres
version :
Templates : free tier. only 1 version is created

Settings instance identifier : dbserver
manage master username : admin
master password : Msois1234
He will have all privileges.

instance configuration
burstable
db.t3.micro.

Storage : type : GP SSD gp2
storage : 20 GB.

VPC : choose your vpc
DB subnet group : Create subnet group
public access : no
VPC securit group
Create new name : dbsecurityGroup
uncheck enable backup.

Create

How to connect to this db server

cd var/www/html/database

mysql -u admin -h _____ endpoint of dB server

-P3306 -p

Msois1234

then directly

mysql> create database ecommerce;

use ecommerce;

> source onlineshop.sql;

> exit;

cd /var/www/html

sudo vi db.php

servername - endpoint

username - admin

pass - Msois1234

db - ecommerce

restart apache2

Module 8: Database 26-9-23

Section 1: Amazon RDS

managed and unmanaged services *

Challenges of relation database

To overcome these challenges we move to RDS

Managed services responsibilities:

RDS DB instances

Class

Storage

Amazon RDS in a VPC

(the usecase we did above)

High availability with multi AZ

Read Replicas

- If there more read operations in your applications you can create a read replica instance ~~instead~~ of § to reduce workload on master dB.

Use cases:

When to use and when not to use.

You cannot scale RDS horizontally.

RDS Does not handle semi-structured & unstructure dat

Section: Dynamo DB

Relational v/s Non relational dB.

Supports horizontal scaling

Sharding :

Document DB · data is stored as JSON format (key-value pair)

In Non relationals

item → record

attribute → field.

SQL v/s NoSQL *

What is Amazon Dynamo DB

- 1 read capacity unit : application can read upto 4 Kb per second
- 1 write capacity unit: application can write upto 1Kb per second.

↳ Scale read/write

low latency queries : 1. It uses simple queries & scan

2. We use primary key (partition key)

3. data is stored in SSD.

Local & Global Secondary Indexes

partition key (hash)
sort key (range)

Service Dynamo DB
Create table : test • ~~table~~
partition key : Id number.
Table settings : Customize settings
Table class : dynamic standard.
Read/write capacity settings : provisioned
Auto scaling read : off write : off
secondary index
create global
partition key : Product Category String
sort key : Price Number
create index
Create table

Actions :
Create item
Id : 0
Add attribute
Str Product Category Book
Num Price 101
Str Title Cloud computing book
bool In-publication True
Num Page Count 345
Create item

Instead of adding separately everytime, we can write this everything in JSON format file and push this to table
put in S3 bucket.

Create bucket
ecomdynamotable
ACL enabled
uncheck block
Create

inside bucket :
Create folder : Lab-data/ -> select & in actions Make public using ACL
-> go inside and upload that .json file.

Create EC2
nodejs sdk

ssh to EC2
sudo apt-get update && sudo apt-get install nodejs npm
sudo mkdir myapp
cd myapp
sudo vi index.js
node index.js
sudo npm install aws-sdk
node index.js

Create IAM roles
~~Create~~ AWS service
Use case : EC2 next
permission policies : effect, actions, resources
dynamo . Amazon DynamoDB Full Access

to attach Role to EC2
actions Security modify IAM role

node index.js (npm, nvm)
vi index.js
copy paste code, change bucket name
node index.js
change table name
node index.js

Amazon Redshift 27-9-23.

Datawarehouse:
Single source of truth to store data of all applications (or departments) of a organization

To set up a data warehouse in organization we can use redshift. It is fast, fully managed AWS datawarehouse any organization can use to set up cost-effective

- to store peta byte of data and can do analysis on this data using simple sql queries
1. optimization of queries (its advantage is)
2. it stores data as columns, you can perform easy aggregation, easy compression/encoding so it run queries in milliseconds (low latency)
3. you can run parallel queries.

Parallel processing architecture.
leader node: who interacts with client (who submits queries) & manage everything, also interact with compute node, it takes source code, divide & give to compute node
compute node: it compiles & runs code given by leader & gives intermediate res
then leader node aggregates this intermediate res

and send it to client and then configure to store either in dynamoDB or S3.

Automation & Scaling.
- Manage. Monitor, Scale

Compatibility:
you can use any tools (SQL clients or BI tools) to interact with redshift.

Use cases:
1. Enterprise data warehouse
2. Big data
3. SaaS.

Amazon Aurora:
Its is RDB compatible with MySQL or postgreSQL
- It is also managed dB.
It is designed for instant crash recovery if your primary database becomes unhealthy
- Service benefits: *
- High availability:
- It stores in multiple AZ and later backuped in S3 bucket.
- It

- Resilient design.
(- Redo log files. ← done by Crash Recovery of Database)
this file stores insert, update, delete operations made

In Aurora redo log files in not replayed after crash, it does it after every operations.
or for crash recovery
read

EKS - Elastic Kubernetes Service 9-10-2023

Under Compute Category. Container based Services:

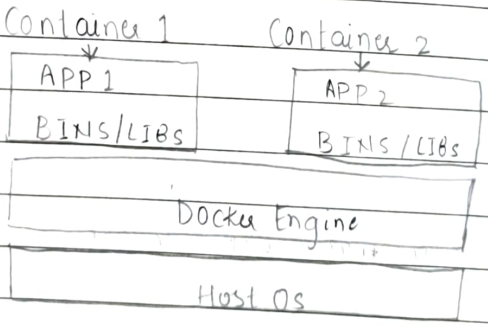
Amazon ECS, EKS, Fargate, ECR.

Why containerization (Virtualization at OS level)

- Makes application hardware independent
- Cross platform dependency
- Interoperability

Software that packages (wraps or binaries) code, optimized OS, packages, libraries into one container that are required to run your application.
(Docker)
(used to containerize application)

Docker:

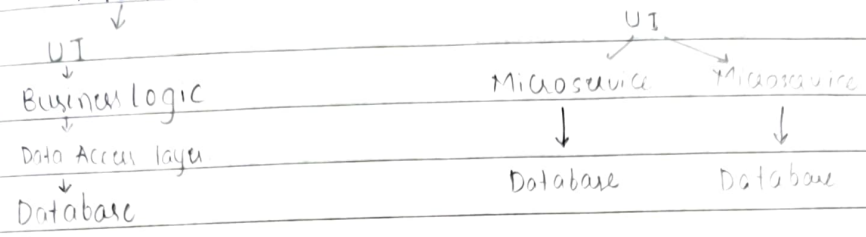


Containers are isolated: Any bug in particular container it effects only that container, other containers or Host OS are not affected.

Containers are light weight because it has optimized OS (not complete OS) so it loads OS faster.

In VM's, Memory has to be allocated before it is launched. memory is static.
where as in docker, memory need not be allocated, memory grows dynamically.

Web application Architecture.
- Monolithic and Microservice



Monolithic : All features are tied together as single code
Microservices : features are treated as diff. module. but interact with each other

Advantages of Microservices :
- Extensibility : New features can be added easily
- Error detection & Correction is easy, and error in one micro service doesn't affect other microservices.

Containers are mainly used to run microservices.

Docker Hub : Repo that stores images
Components of Docker :
Images : template (Read-only) that contains OS, dependencies required to run applic
Container :

Dockerfile —build→ Docker Image —Run→ Container

run → start a container.

sudo apt-get install docker.io

sudo systemctl status docker

To test docker is installed & is able to run

sudo docker run hello-world
image
interacting with docker engine through cli

do we have any images in local system

sudo docker images

To pull image

sudo docker pull centos:latest

Start centos and run a container

sudo docker run --name mycentos -it -d
container name
image name centos:latest

-it → start container interactively and lets interact through terminal

-d → run container in background

if container is started successfully it gives Id

sudo docker ps ← lists all containers that are running

details about containers that ran, exited, running and other containers

sudo docker ps -a

go inside container

sudo docker exec -it mycentos bash
through which shell u want to interact

root@ ... # cat /etc/os-release

yum update

exit

name resolution error

sudo docker run --name myubuntu -it -d ubuntu:latest

sudo docker ps

sudo docker exec -it myubuntu bash

root@ ... #

cat /etc/os-release
apt-get update
apt-get install apache2
copy/download sample index.html
service apache2 status
service apache2 start

cd /var/www/html
ls
rm -rf index.html
vi index.html => vi is not installed so
apt-get install vim ← install all
vi index.html
copy html code.
to create image from Running Container
in other terminal:
sudo docker ps
sudo docker commit 33... (container Id)
myapachedockerimage:latest ← name to your new image
↓
sudo docker images: this should be present
⇒ sudo docker run --name mywebapp -it -d
-p 3000:80 myapachedockerimage:latest
-P => to expose the port where this apache2 application can run → remap

- sudo docker ps
In browse 172 3000 it cannot load.
- We have not told image that apache to start

sudo docker exec -it mywebapp bash
service apache2 status
cd /var/www/html => ls index.html => cat index.html

sudo service apache2 start

sudo docker images
to push to dockerhub
sudo docker tag myapache2 / image Id sree docker123/.myapache docker image : latest

sudo docker push sree docker123/

sudo docker login
sudo docker pull

Building image using Dockerfile.

1. FROM → installs base OS mentioned
2. RUN → runs the specified commands inside OS
3. ADD or COPY → Add files from source to destinat^n
4. ENTRYPOINT or CMD → as soon as container starts it is the first task to be done.
   eg: ENTRYPOINT apachectl -D FOREGROUND
5. ENV → if your applicat^n needs to set up some environment variables
6. EXPOSE →

create dockerapp folder
& create index.html

Create Dockerfile :
vi Dockerfile
→ FROM ubuntu:latest
RUN apt-get update

RUN apt-get install apache2 vim -y
ADD index.html /var/www/html
(ADD . /var/www/html → copy all files in pwd to /var/www/html)

EXPOSE 80
ENTRYPOINT apachectl -D FOREGROUND

(WORKDIR /var/www/html → cmds after this should be executed in this directory
ENV /var/www/html)

To build image:
- sudo docker build -t apachewebapp:v1 .
  (image name, tag name, . → dockerfile is in pwd)

sudo docker images
- sudo docker run --name apachewebapp -it -d -p 8081:80 apachewebapp:v1

Dockerize Nodejs web application → link
(Only developer side)
sudo mkdir nodejsapp & cd
sudo apt-get install nodejs npm
[package.json → Here all the requirements, dependencies are specified
(MEAN - MongodB, Express, Angular, Nodejs)
express is a framework
To install all dependencies go to path where package.json and run npm install]

sudo vi (or nano) package.json. copy paste from the link
sudo npm install . -y

or nano

Sudo vi server.js copy code from link
change const PORT = 8082

Sudo node server.js
172 16 51 124 8082

Now dockerize this
Sudo vi Dockerfile copy code from link
change expose 8082

Sudo docker build -t nodeapp:v1 .
Sudo docker run --name nodeapp -it
-d -p 8082:8082 nodeapp:v1

11-10-2023

to restart containe aftu reboot / restart
Sudo docker restart myubuntu containes name/i
removes all containers in one go (running as well exi
Sudo docker rm -f $(sudo docker ps -a -q)

Volumes : to store data of container
irrespective of container state.

Sudo docker run --name myubuntu -it
ubuntu:latest
root@... # mkdir /app
cd app
touch test.txt
cat >> test.txt
Enter (Hello welcome to docker volume)
cat test.txt
exit Thank that :) of any error or bug
container exited/removed

sudo docker rm -f myubuntu

Now we don't have the backup of the data in this containes

so to persist data we use volumes.

mkdir dockervolumes.

cd dockervolumes.

mkdir app

sudo docker run --name myubuntu -it -v app:/app ubuntu:latest.

Sharing folder with container

source folder on host data has to be stored (shared volume)

folder in container

ls app folder is present.

touch test.txt

cat >> test.txt. enter some data

cat test.txt

Now if this container is accidentally removed, the data u stored in shared folder app.

sudo docker rm -f myubuntu

sudo docker run --name ubuntutest -it -v app:/app ubuntu:latest

→ Here we can get the data that was removed

cd app → ls → cat test.txt → exit.

But this is fine if both have same file system.

If both have different file system, docker came up with docker managed volume: instead of we creating volume in host, docker will create volume.

sudo docker volume create myvolume
sudo docker volume ls mounted or present
To check where volume is present in host
sudo docker inspect volume myvolume
name

-v used managed volume

sudo docker run --name ubuntuvolume -it
--mount source = myvolume, target = /app
volume managed by docker
ubuntu : latest

ls
cd app
touch file1.txt file2.txt
cat >> file1.txt Enter data into files.
Cat >> file2.txt
exit.

sudo docker run --name centosvolume -it
--mount source = myvolume, target = /app
centos : latest

ls
cd app -> ls we get files created in ubuntu

Docker Compose : uses micoservice architecture
Eg: ~~For ho~~ To ~~edeq~~ split application into multiple containers, tie them together and start, run them together.
Eg: Host dynamic website. Containerize communication
1 container presentation layer 2. Database.

sudo git clone https://github.com/sreepathysois/
phpmysql-app.git

cd phpmysql-app/
ls

vi docker-compose_updated.yml.

=> This file tells that it contains 2 containers
1. web 2. db. and how these containers is started

cmap cat php/online-shopping-system/Dockerfile_update_webapp.

cat php/online /database/
Dockerfile-db
(docker-entrypoint-initdb.d) where all sql files are stored

→ sudo vi php/..
change → FROM php:7.4.3-apache.

To install docker compose:
sudo apt-get install docker-compose

Sudo docker-compose -f docker-compose_updated.yml
up --build -d
bring up all containers start in background

(Wordpress deploy with docker-compose linode) link
cd in home
mkdir wordpress
cd wordpress/
sudo nano docker-compose.yml copy code from link
change ports to 8009:80.

sudo docker-compose up --build -d.

Kubernetes : (K8s)

16-10-2023

Consider you don't have sufficient resource or network issues on hosts or there is no replica of your container (ie scaling up)

So we need a service to monitor health of the containers, scaling, to manage workload - and the service is Kubernetes.

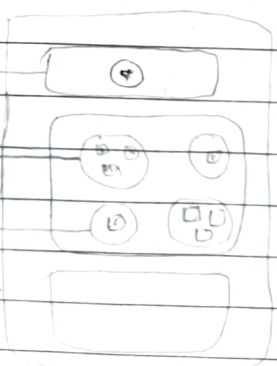
BORG : Container management service

Kubernetes is a light weight
The containers that we have pushed to dockerhub, ~~kubb~~ kubernetes pulls it and deploys it. (through Automation)

Kubernetes Architecture & Components.
Master node, Worker node (currently it supports 5000 worker nodes)

Worker node : Runs your container application
Master : manages everything.

↳ Pods : Basic scheduling unit of K8s (It is to keep containers isolated)
It is collect^n of compute resources to run your applications. (CPU, SSD, n/w performance)



Worker node components.

You should install container engine to make worker node to work

1 pod may contain 1 or more containers.

watches state of cluster & try to modify current state of cluster to desired state

It is where all administrative tasks are performed

Master → kubectl (command), UI (K8s dashboard)

API - web interface to interact with master (to do tasks like create, manage, delete, update real K8s Object/pod) ← API server

API server - It does these tasks | Gateway to your complete cluster.

schedules all work in the form of services & pods

Scheduler : schedules your pod by finding best possible worker node as per requirements / specification
It also keeps track of state of all pods

Control Manager : It manages the health of pod or cluster or node

→ Node, Deployment, Endpoint, Replica, Service, Token
(node) (pods) (endpoints) (service is working fine)
controller processes

subordinates.

etcd : distributed key value dB.
(sharding) Complete cluster info is stored here or current state of cluster.

* Worker node components

Kubelet : primary node agent which has to be installed in all clusters/nodes

- It continuously checks health of node by sending heartbeat. to control manager.
- It also checks health of pod
- If node is unhealthy, kubelet informs control manager it restarts by it own. if it doesn't start ↑

Kube-proxy : It takes care of networking between hosts, pods, containers.
- It also exposes the services to internet or container application.

Container Runtime.

Kubernetes Pod & Network. 19-10-2023

cAdvisor : ~~continuously~~ continuously monitors the metrics

Overlay Network : Flannel, Calico, Wavenet

Kubernetes Container Deployment.

~~Node~~ load balancer, Node port, Cluster IP

You don't want to give access to outside world

Deployment is object used to create a pod & run container

Service file :

Demo :

ssh ansible@172.16.51.50

kubernetes master

Get details of cluster :

sudo kubectl get nodes

hostname cat /etc/hosts

Master has role → control-plane

sudo kubectl get pods --all-namespace

Create deployment and expose service

sudo

$ kubectl create deployment nginx --image=nginx:latest --replicas=2

to see whether deployment is created

sudo kubectl get deployments

sudo kubectl get pods

classmate
Date
Page

name of deployment
sudo kubectl expose deployment nginx --port=80
--type=NodePort → automatically remaps port no bet^n 30k to 65k
type of service

sudo kubectl get svc service
Go to browser · give IP address of worker node : port

Delete deployment & service
· sudo kubectl delete deployment nginx.
· sudo kubectl delete svc nginx.

sudo kubectl create deployment mybookalbum
--image=sreedocku123/bookmoviecalbum:latest
sudo kubectl get pods

Get details of pod
sudo kubectl describe pod mybookalbum-....
name of pod created

sudo kubectl expose deployment mybookalbum
--port=80 --type=NodePort
sudo kubectl get svc
Go to browser IP address of worker node : port /index.php

Now we can write these command in deployment file & service file

cd basic-php-website
vi bookalbum_deployment.yaml
vi _service.

sudo kubectl apply -f bookcatbuu_deployment
" " " " " -service.y

sudo kubectl get pods
" " get svc

kubectl, kubeadm, kubelet
install weavenet
sudo kubeadm init master

sudo kubeadm join

In master:
~~kubein~~ $. sudo kubeadm init
Your control-plane has initialized successfully

$ mkdir -p $HOME/.kube
sudo cp -i /etc/ ....
Sudo chown .....
export KUBECONFIG ...

→ sudo chown -R msis:msis /etc/
kubernetes/admin.conf
kubectl apply -f https://.....

kubectl get nodes → master & node should be ready
Kubectl get pods --all-namespaces
Here all status should be Running

Then you can create deployment and svc.

sudo systemctl restart docker
sudo swapoff -a ⟶ ~~sudo kubeadm reset~~
sudo kubeadm init

28-10-2023.

Serverless Computing : → abstracted service
eg. AWS Lamda.
Under Compute Category

When to opt for this.
- Event triggered or scheduled based.
- When your code needs to run for only small set of amount of time. (We don't need any servers, computing resources, managing server.)

1. When customer uploads any object to bucket, it ~~Create~~ should call Lamda and check what type of data is uploaded (.jpg, .png,), and that has to be updated in Cloud Watch. and can be duplayed through CloudWatchLog.
Cloud Watch monitors

We need to create IAM role so that Lamda can access bucket data. and cloud watch
read policy put / post

1. Create a bucket
bucket name : msislamdas3test us-east-1
Rest all default → Create bucket
Go to bucket & upload file.

2. Create IAM role
Policies should mention ⇒ i What are the actions you perform.
affects: ii Whether these actions are allowed or denied
on what iii resource. i.e S3, logs

Go to IAM → Policies → Create policy.
JSON → paste code → next
policyname - s3lamdareadpolicy → next

next goto roles in IAM only
Create role → AWS service
use case → Lamda → next
Select S3lamdareadpolicy → next
role name → S3lamdarole → create

3. Go to Lamda
Functions.
Create → name: S3lamdatest
Runtime → python 3.9
Architecture → x86_64
change default executive role → use existing role
→ Create

Code → paste code in lamda function
→ deploy
Add trigger → S3 bucket
bucket → msis1amda s3 bucket
Event type → all object create events
☑ I acknowledge → add.

Test → to test your code works properly before releasing
Eventname - mytest
private
Template: S3 put
Event JSON.
input ⇒ change region, bucket, name of image.
S3 (in arn & name) Object (key-
Save & Test
View Cloud Watch Logs

classmate
Date
Page

(2) Create thumbnail of the image you upload in src bucket and put it in dest bucket.
Event trigger function - create thumbnail whenever image is uploaded and push it to dest bucket

Create 2 S3 bucket (src, dest).

1. msissrcbucket name all default create
upload an image.

2. ~~msisdestbucket~~ name all default create.
msissrcbucket-resized

Create IAM policies:
IAM → Policies → Create Policy
JSON → paste code → next.
policyname : S3lamdathumbnailpolicy → Create

Goto Roles
Create role → AWS service
use case → lamda → next
select the above policy → next
role name → S3lamdathumbnail role → Create.

Goto lamda → Functions
Create → name : s3lamdathumbnail
runtime → python 3.9
Architecture → x86_64
Executive role → use existing role → above role
→ create.

Add trigger → S3 bucket
bucket → src bucket
Event type → all object create events.
☑ I acknowledge → add.

EC2 write scheduled based trigger to stop & restart EC2 instance

If you have your own libraries or dependencies for your code to run you have to package it all. (It is lambda funct^n package). Here we use pillow library

To create package - install all dependencies binaries, and make zip file of all this along with code also

Code → upload from → select zip folder

Test →
event name - mytestevent
private
template : s3 put
Event JSON
change region, bucket name of image
Save & test

Syllabus : Databases : diff b/w managed & unmanaged
diff b/w sql v/s nosql
RDS, with use case
DynamoDB use case
Redshift, Aurora

Compute : Containers ie Docker → all write a docker file for it

k8s - need of it, Architecture
how you deploy & expose service
Serverless Computing

Module 10: 7-11-2023

Automatic Scaling and Monitoring.

Cloud infrastructure for scaling automatically, highly available, fault tolerance, elastic

Theory:
- Elastic Load Balancing
- Amazon Cloud Watch
- Amazon EC2 Auto Scaling

1. Elastic Load Balancing:

You cannot provide multiple users concurrent access with only 1 server. In modern era; we use more servers.

ELB which accepts requests, then divert traffic of client to targets (EC2, containers, IP addresses, lamda functions)
- also divert it to multiple Availability zone
- Elastic → scales on it own.

Types of load balancer

1. Application LB
2. N/w LB : faster compared to applicat^n LB designed to handle more requests of communicat^n bet^n machines
3. Classic LB → No target groups.

How ELB works:

Listener : process that checks for connections. requests

two things : what protocol it should listen to
: port

when it receives request on this it diverts it to diffet target.

> configure listener.
1 Create target group, register them

Target group : Collection of targets.
LB tells listener to send requests to target group.
You can have multiple listeners
Load balancer also checks health of targets i.e it sends ping (Protocol) to all targets. for every 3 or 5
- if it receives response back target is healthy
- if it doesn't receive response successively for 3 times, then target is unhealthy & LB doesn't divert request to that target for traffic

Load balance Monitoring:

1. Cloud watch :
   Continuously monitor services & application on top of services.

2. Access logs doesn't track all API track
3. AWS Cloud trial logs : track all API data

2. Amazon Cloud Watch :
- To get insight into your AWS services
- It is monitoring & observability

1. monitor 2. Collects & tracks metrics 3. Alarms 4. Events

metrics : variables that are used to measure performance.
Eg: CPU utilizat^n, Network I/O throughput disk utilizat^n

Alarms : you can create alarm for particular metric.
Eg: If CPU utilizat^n of EC2 instances goes higher than 60% send alarm (action) & tell what should be done.

Events:
Eg: When EC2 instance change from running to stop state, (take snapshot) route these events to one or more target function.

namespace: AWS resource on which it should to monitor.

3. Amazon EC2 Auto Scaling: service for achieving application auto scaling as mentioned
- fleet mgmt: it makes sure that desired number of servers are running at any point of time.
- checks health of web servers.

Scaling options:
Manual, scheduled, dynamic or on-demand, predictive (AWS Auto Scaling) ← predictive scaling option

Auto Scaling Group:

Scaling out v/s scaling in.

How EC2 Auto Scaling works:
1. Launch configuratn: AMI, instance type, IAM role, (details abt how to launch instance) security grp, EBS vol
Auto Scaling grp
VPC & subnets, load balancer
When - type of scaling

Implementatn of dynamic scaling
ELB, EC2 Auto Scaling, Cloud Watch
Auto Scaling group.

8-11-2023

Web instance and database are in private subnet, application load balancer in public subnet

* Create VPC.

VPC and more
name - ecomvpc
IPv4 CIDR - 10.0.0.0/16
No IPv6
No. of AZ - 2 2 public & private subnets
customize CIDR
public subnet -1 : 10.0.0.0/24
2 : 10.0.2.0/24
private subnet -1 : 10.0.1.0/24
2 : 10.0.3.0/24

NAT gateway : in 1 AZ
VPC endpoints : none
s/w resource that is used to access resources sitting outside the VPC

* Create EC2 instances

name : mywebserver
AMI : ubuntu 22.04
Instance type : t2.micro
key pair

N/w
Select created vpc
subnet : public 1 us east 1a
Auto assign public IP : enable
create security grp : type → ssh Anywhere
add rule Http 0.0.0.0

* Create.
After Creating do ssh to this instance.
In terminal
scp -i ~Downloads/ -r Html.
Keypair Downloads/
ubuntu@ : /home/ubuntu/.
Public IP of EC2 Src of static website

→ $ cd. html.
$ sudo apt-get update && sudo apt-get install apache
$ sudo mv html. myapp/ · rename static website
$ sudo cp -rf myapp/ /var/www/html/.
$ cd /var/www/html → ls see myapp
$ sudo vi /etc/apache2/sites-available/000-default.conf
→ Document Root /var/www/html/myapp
$ sudo systemctl restart apache2.

* Create one more image (AMI) EC2
Action → Image & template → Create image
name : ecomami
descp : This is AMI of ecom applicat°
Add new tag : Key : Nametage
Value : Myecomami
Create image

Create target group : type : instances
name : myecomtargetgroup
protocol : & http : 80 where targets listen to
IP address : IPv4

VPC : the vpc you created
protocol version : HTTP1
Health checks : protocol : http
path : /
next → create target group

Create load balancer :
type : applicat^n L.B
name : myecomelb
scheme : internet facing
IP add typ : IPv4
VPC - the VPC you created
Mappings : check both AZ & choose public subnet
Security groups :
. allow http from anywhere
Listeness & routing :
Protocol:HTTP : 80 ← Port . select target group you created

Create launch template - details about how to launch instance
name - myecom launch template
descrp - template to launch ecom website.
AMI - the ami you created.
Instance type : t2.micro
key pair :
N/w setting
VPC - subnet → private 2
Security group :
Storage :
Resource tag : key: name - value : myecomweb
Create.

Create
Actions → Auto scaling group
name : ecomasg
launch template : one you created
next → VPC - one
AZ & subnets - select 2 private subnets
next → load balancing : Attach to existing LB
choose from LB target grps. - one you create
Health check - turn on ELB health checks
grace period : 120 sec
Additional settings : monitoring : check enable. ..... CloudWatch
next → Group size & scaling policies
desired capacity → 2, min capacity → 2
max cap → 4.
Scaling policy : target tracking scaling policy
metric type : Avg CPU uti
Target value : 40
warmup : 60 sec
next → add tags Key : name value : myecomweb
Create.

In browser search with load balancer DNS
try terminate on EC2 instance, auto scaling
group creates one more instance.
Test scaling policy →