April Day to Day notes || Introduction 27th April 2025
———————————————————————

DevOps is not just a goal but a never-ending process of continual improvement and life long learning…

Agenda

1. Welcome
2. Introductions to DevOps
3. SDLC
4. Setup review
5. DevOps Tools
    1. 10

Automate —> Task —> manually —> automate —> automate automate

Intergate —> tools and tech

Inovate —> 2025 —> pipeline —> AI

DevOPs —> mastery

AIOPS

MLOPS

Take 1 hrs —> per day

5 hrs a week

Ubuntu
Centos

DevOps —> pipeline —> Youtube or Medium pOst —> chatGPT

12:30PM —> Lecture —> 12:20PM

12:40 —> Interview Type

SDLC
DevOps

Three interview —> DevOps 1min —> 50% —>

12:45 —> or 12:50 —>

1. Linux
2. 5min

DevOps

Traditional IT and Rest.

Real —> Traditional —> New features—> Overnight —> Dev 403 —>

Code —>
Cp —> Backup

Manual

Phonepe —> new release —>

IRCTC —>

DC and DR —>

Data center Live

Rolling update
Blue Green

DevOps —>

Best Practices—> KB

AWS

Requirement :-

Startup —> Laundry Services
30 Min —> Book

Planning

1. Booking app
2. Status
3. Delivery

Designing

1. React FE NODE Js
2. Python API

Dev :-

1. Dev — FE
2. Python - API
3. Database —>

TEsting :-

1. Login
2. Logout
3. Booking

Laptop —> Code Stop and application Stop

Server —> Data center or AWS or Azure or GCP

Three Months

Three Complete

2 or 4 Week

Scrum Call —> Stand-up —> three days in a week —>30min

Release product —> 4 week —> release Last —>

SRE —> make 100% running —> production impact

1. Bug —>

2.

PE —>
DevOps —>
Cloud Engineer —> AWS cloud

JD

PRoduction —>

PRoduct —>

DevOps

1. Plan the sprint tasks —> New deployment —> infrastructure AWS or Azure
   1. Terrafrom
      1.
   2. Ansible configuration
         1.
   3. Jenkins
2. PRoduction
   1. URL
   2. API
   3. FIx
3. Monitoring

——
——
DevOps Tools || 3rd May 2025
——
——

1. Revisit —> Similar
2. DevOps Tools
3. Linux Start basic commands

Waterfall —> One by one

DevOps Tools —> OpenSoucre

Laundry —> Bring App —> dev 30 MIn to 1Hr

Owner —> one Engineer —> Development

Business and Idea —> Take care

Basic page —> Idea —> Dev

Start working —> day1 day2 —>
Next —> page page

Dev

1. Windows
2. Visual Studio
3. Chrome
4. Nodejs

Page 2 —> Looks

Page 3 —>

Week Day

Amazon EC2 Server —> copy index.html —> Ngixn webserver

1. LInux UBuntu 24.04
2. Nginx Webserver
3. SSH (COPY)

From Website —> office

Dev-> home

LInux —>

Next One—> Product —> 10 to 20

24/7 —>
EC2 new production

Dev1 —> ec2 Server —> Virtual laptop Run 24/7 —>

Dev1 —> PRod

100 to 200 customer

Bug
Feature

Dev1 and Dev2

Code manually

Dev Deploy

Code Merging and collaboration issue between two dev

GIT.

Help you to review the code —>

Diff —> dev 1 10 line

Dev2 10 line

Compare

Pull request and Merge

Compare the code

Approval

Amazon —> Server 24/7
Linux —> open source
Git —> merging

100 Feature —> Bug —>

Dev1 copy —> Dev

Dev1 —> QA —> Feature —> Prod

Dev1 —> Prod

Jenkins —> DevOps FreeStyle —> Pipeline —> Grovvy —

THree Job for each env

Dev —>
QA —>
Prod

Jenkins Helps for deployment of code

Dev1 —>

1. Windows
2. Visual Studio
3. Chrome
4. Nodejs

Dev2

Docker File Docker
Docker Image —> ubuntu Server

Dev3

1. New code

Dev Laptop it working

Server

Docker

Laptop

Amazon —> Ec2 Server —> 24/7
Linux —> Deploy code index.html —> opensource

Git —> Two devs are able merge the code

Jenkins —> Repeate —> Code copy paste from local to Server --> automation —> Click

Docker —> Setup —> bare —> Linux —> Node js or vim and other application —> Suitcase —> docker image

Laundry 1000+

Two EC2 —> 100 customer

20 EC2 —>   1000

Docker —> EKS

20 EC2 —> 2 Server down —> 12 AM —> 11PM

Login —> EC2 Server —> 20 Node

EKS —> EC2 —> Node —> Down —> Auto H

Docker container —>

K8S —> Keep all the required Container up and running

Login —> Customer —> booking —> Pick up —> Submit —> 404

Killed —> cpu and two

Refresh —> Submit

K8s —> Container —> failure —> kill delete

New container —> Docker image

Auto-Scaling 10 Users —> 50 Users

2 Instance

CPU —> 70 % and above —> 4 Increase

Cloudwatch Monitoring Alarm Trigger

K8s —> Auto

10 container —> 10 continaer

Amazon —> EC2 Instance —> 24/7

LInux —> Deploy the code

GIt —> Merger the code

Jenkins —> automation —> Click deployment

Docker —> Runtime and all dependency (one in place)

K8s —> Ors. Container (POD)

Vendors —> 1000 —> 1,00,000/-

EKS —> 20 Server

200 Server —> INfra Scale —> Management —> Manually Server —> Create

Terraform —> Set of instructions file —> .tf —> 20 Server

CloudProvide —>

Cloudfromation

Terrafrom Destory Infra—> Git —> approval of management —> change

Ansible —>

Terraform—> Create new server —

Vagrant —>

Login __>

Terraform —> 200 Server created

Ubuntu —>

Code —> Laundry —> into server —> 200 Server

Copy and paste or Jenkins —> Shell or ANsible or any other at same time

Execution —> 200 Successfully

Terraform —> before login into server

ANsible —> after login into the server

200 Server —>

20000—> server down —> shudown —> manager down application —> login 4 Server —> application —

Reactive approach —>

Disk usage —> Server —> 25GB —> Ec2 —> 3 to 4 Month —>

Light —> /var/log —> 25GB —> Full disk —

Login —> /var/log/nginx/access.log

> access.log 75%

DevOps Proactive —> Promethues and grafana

Prometheus —> 75% —> alert Slove —> Next —> fixed

Database —> Monitoring metrix

Grafana —> Plot

Break Time

10 Tools —>

Laundry Services

1. Amazon —> Latest —> Azure —>
2. Linux Ubuntu —> Code
3. Git —> Coll and VCS
4. Jenkins —> Automating deployment
5. Docker —> runtime and conman management
6. K8s —> Ors container —> Container —> create container (POD)
7. Terrafrom —> EKS or EC2 —> Configuration —> Name —> File —> all
8. Ansible —> Build the docker image —> jenkins and Deployment —> Configuration Management tools
9. Prometheus —> Proactive Alerts and fix the issue
10. Grafana —> Visual

Linux —>

Ubuntu —>

Memory
CPU —>
Green —>

SSH —>

Linux

1. ls —> list
2. ls -l —> long list
3. Mkdir —> maker directory (Make folder)
4. Cd —> change directory
5. pwd —> print working directory
6.
7. Rm
8. Rm -rf *
9.
10.
11.
12.
13.
14.

POA

1. Create new file —> touch file1
2 . Cp file1 file1_backup
3.rm
  4. Rm -rf
  5. Rm —> file
  6. -r ->
  7. -f —> force

4.rm -rf /tmp/*
Ls /tmp/*

- /tmp —> * ls

Rm -rf /tmp/*

Ls /tmp/*

Recap

1. 10 DevOps
2. Linux
3. Centos
4. Server
5. Command Internet

Thank you!

——
Linux || 4th May 2025
——

1. Revisit
2. Linux Deep Dive
3. Linux Basic

Hi, Just want to know what is the difference between Continous Deployment & Continous Delivery

CD

Ansible

Which tool allows you to create or provide server using modules backed by python

Docker —> COntainer

Docker image Set of instructions base or runtime nodes or python file —> Docker image —> container —>

Container —>

LInux —> ?

1991 —> Linus —> Unix —>
Linux —> OpenSouc

1. Ubuntu
2.

What is Linux ?

Kernel is heart of OS? —> Linux

XP —> 2GB —> 2GB
2007 —> 4

1. Opensource —>
2. LLM’s
    1. Managed Azure DeepSeek Module
    2. Amazon Bedrock opensouce based LLM’s

Wsl

LTS —> Long Term Support —>

Application
NOdejs —> LTS

10.0.10—>

10.0.0 —>

Labs

1. ls
2. Mkdir
3. cd
4. Cd ..
5. Pwd
6. Touch
7. rm
8. rm -rf
9. Backup
10. uname -a
11. Cat /etc/os-releae
12. lbs_release
13. cd ..
14.

SSH —> Secure Shell —>

SSH —> Tools

SSH Services

Sudo systemctl status ssh
Sudo systemctl status sshd

Sudo —> super user do —>

Run as a administrator —> sudo

Systemctl —> application —> services

Systemctl status

Sudo systemctl status ssh —>

Found not

New application install in ubuntu

Sudo apt update —>
Sudo apt install ssh

Sudo systemctl status ssh —>
Sudo systecmctl start ssh —> To start the service
Sudo systemctl enable ssh —> disable —> Server shutdown or restart

Services file —> /etc/ssytem —> Services file —> path —>

127.0.0.1 —> Any Server —> eth0 —> 127.0.0.1 —> localhost —>

Workaround —> Forwarding —> 22 —> Virtual box —>

SSH —>

1. NAT
2. Bridge

127.0.0.1 —> laptop —> Virtual box —> 22 —> Ubuntu 22 —> NAT

Thank you!

———————————————————————————————————————
———————————————————————————————————————
Linux || 10th May 2025
———————————————————————————————————————
———————————————————————————————————————

1. Revisit
2. Networking
3. Linux Advanced
    1. Apt deep divide
    2. Vim
    3. Paths

A. ssh user@remote_server

User@

Shell scripting —> Automation —> Change

Systemctl < > sshd

Reload —> configuration read
Restart—> Stop and start
Start —> start
Status —> Status
Stop —> Stop
Enable —> Create new services which will make sure services always during any restart or server stop
disable —> delete unlinked the file enable

Sudo apt install ssh or any other

Apt-get older version —>

Apt —>

yum

apt

snap

https://www.videolan.org/vlc/#download

https://get.videolan.org/vlc/3.0.21/win64/vlc-3.0.21-win64.exe

WIndows —> go to that website —> google —> get the actually url —>

Linux —>

1. Apt —> automated
    1. Apt install sshd
2. Rpm or dpkg —> manual way of installation
    1. dpkg —> ubunut
    2. rpm —> Centos
3. Build source code based code
    1. make

    2. Make install

DevOps on Linux (server

1. Installation
2. Configuration
3. Maintaince

Installation

1. Automated
    1. Ubuntu —> apt install ssh
    2. RHEL —> yum
2. Manual
    1. Dpkg —> Ubuntu
    2. Rpm —> Centos
3. Source code
    1. Make
    2. Mkaeinstall

APT —> automated

Install new application nginx in my ubuntu server

1. Apt install nginx
2.

Apt workflow

1. Apt install nginx
2. Nginx is available /etc/sourc.d
3. Download
4. Install
5.

Network —> AWS Cloud —>

Subnet —> /24 —> set of number of IP addresses —>

Server —> apt

Automated —> Choco —> client —> windows laptop

NAT —>

192.168.0.1 —> 103.0.0.1 —> public IP address ISP —> AWS Elastic or Public

AWS

TCP —> Classes —> query —> ask reply —> answer —> https —> 200 —> 400

UDP—> YouTube —> tel—> intreactive —> kakfa

Port open —> 22 or

AWS —> VPC network

Elastic IP —> Static IP —> AWS —> Static —> release IP address

Subnet public —> onboard —> 1,00,000/- rotate —> user a

IPS —> Static —> 5000 or 6,000 —> Static IP —> Static page —> website

can you explain more in dept about ip address. hostid , net id and subnet mask.

IP Address —> In depth —> CCNA or Network —>

Each server —> dedicated IP —> adher card —> Name (DNS Name)

Private IPV4 —> Innovation IOT —> fridge IP address —> smart IP

IPV6 —> mac address address —> ::

Jio number —> IPV6

MAC address —> number provided each—> unique hostid —> hostname —> dev-web.abc.com

Subnet /24 /16 /8

10.10.0.1/24 —> Ip address assigned

VPC network —> 200 /24

500 /8

Path —>

relative path
Absolute path

Shell Scripting

Path /home/devops/april-2024/testing1 —>

Relative shell —> working shell —> directory —>

Ubuntu —> nano

vi —>
and vim —> vi

Editor —> Notepad ++

Short cut
Easy
Color

Lots features

1. Extension
2. Insert mode
3.

Lab

1. Vim lab-linux.txt
2. Insert mode
3.

Cut —> dd
Copy —> yy
Paste —> pp

Searching

Find —> real time search

Relative path

```
devops@devops:/var/log$ cat auth.log | grep devops | wc -l
519
```

Absolute path

```
devops@devops:~$ cat /var/log/auth.log | grep devops | wc -l
519
```

Recape

1. Linux
2.

——
Linux || 11th May 2025
——

1. Revisit
2. LInux
   1. Tar
   2. Awk
   3. Sed
   4. Log-rotate

Mkdir ./testing —> relative path (current path)
Mkdir testing —> Create directly in current location
Mkdir /home/devops/testing

Manual —> .deb or .rpm

Automated

Tar ball based install

1. Which command is used to install cowsay.deb package and automatically resolve any missing dependencies?

Ansible —> Python —> Lib

14.142.38.72

:set number
:%s/source/destination

Dd
Yy

Ls
Mkdir

Mkdir -p

Find <path-where-to-search> -name testing-path

Find ./ -name testing-path

Sir, can you please give a scenario for "Find" Command

Shell script

Manual —> .sh —>

Manual —> nginx —> conf.d —> conf

Find .conf

Find —>

Find ./ type f -name *.log
Find ./ type f -name test*

Locate —> Example —>
Grep

Locate —> Search file or folder in linux server —> Database —> update —>

1000 locations —> File system —> / /tmp

File

Mkdir /tmp/new-dir

Locate —>

1. sudo apt update
2. sudo apt install plocate
3. Locate <file-name>
4. New directory
5. Local new-directory
6. Sudo updatedb
7. Local new-directory

Grep —>

Wc -l

Filesystem
Log-rotate

——
——
Linux || 17th May 2025
——
——
1. Revisit
2. LInux
    1. Cut
    2. Awk
    3. Sed

    4. Tar
    5. Log-rotate
    6. User management
    7. File System
    8. Ownership changes
    9. sudo
3. Shell scripting

Zip

You need to search for the word "timeout" in all files under /var/log, excluding binary files. Which command should you use?

You want to find all empty files in /tmp and delete them. Which command should

Get the data operation Existing —> print —> redirect —>

aws '{print $1}' syslog —> Space

Extration

Sed —> Modiftaion —> change the value of content

| tee —> redirect to specific file

> new-file-redirect

Lab

- vim sed.txt
- Add lots of data
- sed 's/print/printout/g' sed.txt
- sed 's/print/printout/g' sed.txt > new_sed.txt
- sed 's/print/printout/g' sed.txt | tee new_sed_tee.txt

> —> single —> Redirect —> replace

>> —> Append —>

2>> —> if you script —> error —> redirect —>

Cat /etc/file —>

Cat /etc

>> —> 1>>

Standard input and standard output —> 1>>
Standard error —> 2>>

Shell —> 2>>

Tar —>

Zip
Winter
7zip

Comprase —> fiel 500GB —> 7zip —> 100MB —> 10MB
Winrar —>

Tar —>

Lab

1. Create new directory —> mkdir data
2. cd data
3. truncate -s 10M file{1..5}
4. cd ../
5. tar -cvf data.tar data/

Private IP address —> RFC —> In your configure —> AWS VPC

IP address —>

10.0.0.0/8 255.0.0.0 —>

192.0.0.0/8

1720.0.0/8

Public IP —> ISP —> change —> 100 —Static IP address —> 100 103
49

10*5 files —> merge —> single file —> 51MB —

```
cd /tmp
tar -xvf /home/april-2024/data.tar
ls l data
Cd data
Ls file*
```

```
Cd /home/Username/
```

Comprase
```
tar -cjvf data.tar.gz data
```

Extraction
```
tar -xjvf data.tar.gz
```

——
——
Linux || 18th May 2025
——
——

1. Recape
2. LInux
   1. Tar
   2. File System
   3. User management
   4. Ownership changes
   5. Log-rotate
   6. SFTP
   7. sudo
3. Shell scripting

Bzip2 —> 50MB —> 252k

Zip —> 5MB

Extraction

Source —> data1.tar..gz

1. Change the directory the actual where you want to extract.
   1. Cd /tmp
   2. Tar -xzvf /home/devops/data1.tar.gz .

2. Run the extract command from source data1.tar.gz and provide the destination path -C
   1. Tar -xzvf data1.tar.gz -C

Checksum —> value —> to calculate the size of a file or directory

Lab
1. ls -ld data
2. tar -czvf data1.tar.gz data
3. md5sum data1.tar.gz > checksum.txt
4. tar -xzvf data1.tar.gz -C /tmp

File System —>

Cidco —> or under construction—>. 2 BHK —> Colloum box slap

Flat —> 2Bhk —> Key —> Kitchen—> space —> OC

Inter. —>

Hall —> Din table

Kitchen —>

Formating —> Home space

Server or Desktop —> ext4

NTFS

Fat32

XFS —>

R —> read —> v
W —> Write —> Edit —> I —>
X —> execute —>??? —> Script —> Shell —> executable

Chmod u+x sh.sh

New user add —> Linux —> Group Create

Lab

1. Create new file with name user.txt
2. Remove all other permission
3. Give permission to group only with write permission

Chmod ugo-rwx user.txt
Chmod g+w user.txt

User Management

1. Add user
2. Delete user
3. Common ansible add or automation
4. Lastlog
5.

Add user
- sudo adduser april
- New terminal try to ssh with new user ssh user@127.0.0.1

Add user to group —> April user —> devops —> sudo adduser april devops

sudo vim /etc/sudoers

root

Yy
P

With-user

:wq!

Home Work

1. Permission
2. User management
3. Sticky Bit permission

——
——
Linux || 22nd May 2025
——
——

1. Recape
2. Linux
    1. Log-rotate
    2. SFTP
3. Shell scripting
    1. Basic
    2. Variable

 /var/www html/

Ansible user or deployment —> other

Ubuntu —> Shared package installation —> shared —> package —> Internet —>

PenDrive —>SFTP —> File sharing

PenDrive —>

SFTp —> central location —> Central Server —> Files —> package or document or any source code

Cho

SSH —> Normal — SFTP

Log-rotate

Scp

What is logs?

1. Login
2. Page
3. Reset ter

What is the importance?

Lab —> Logroate

1. Logs file app.log
2. Logrotate services
3. sudo systemctl status logrotate.services
4. sudo systemctl status logrotate.timer
5. Create own time
6. Create own services
7. Configure custom logroate file for custom create
8. Sudo systemctl status logroate.service

CopyTruncate —> Log roate —> down —> Mv —> New

Two file

Logrotate.services

logratate.timer
April

Sudo

Lab

1. sudo mkdir /var/log/april-app/
2. sudo truncate -s 100M **/var/log/april-app/**app.log
3. Create two services file

- sudo vim /etc/systemd/system/logrotate-april.timer

```
[Unit]
Description=Run logrotate every 2 minutes

[Timer]
OnCalendar=*:0/2
Unit=logrotate-april-2024.service
Persistent=true

[Install]
WantedBy=timers.target
```

sudo vim /etc/systemd/system/logrotate-april.service

```
[Unit]
Description=Test Logrotate Service for april

[Service]
Type=oneshot
ExecStart=/usr/sbin/logrotate /etc/logrotate.d/april
```

sudo vim /etc/logrotate.d/april

```
/var/log/april-app/*.log{
      maxsize 100M
      daily
      copytruncate
      missingok
      rotate 10
      compress
      delaycompress
      notifempty
}
```

sudo chmod 777 april-app/ -R

Sudo systemctl restart logrotate-april.timer

Sudo systemctl status logrotate-april.timer

1. sudo truncate -s 101M **/var/log/april-app/**app.log
2. ls -l

Lab

1. sudo mkdir april-app
2. sudo truncate -s 100M app.log
3. Create two services file

* sudo vim /etc/systemd/system/logrotate-april.timer

```
[Unit]
Description=Run logrotate every 2 minutes

[Timer]
OnCalendar=*:0/2
Unit=logrotate-april-2024.service
Persistent=true

[Install]
WantedBy=timers.target
```

sudo vim /etc/systemd/system/logrotate-april.service

```
[Unit]
Description=Test Logrotate Service for april

[Service]
Type=oneshot
ExecStart=/usr/sbin/logrotate /etc/logrotate.d/april
```

sudo vim /etc/logrotate.d/april

```
<Correct-Path>/april-app/*.log{
        su <User name and group>
        maxsize 100M
        daily
        copytruncate
        missingok
        rotate 10
        compress
        delaycompress
        notifempty
}
```

```
sudo chmod 777 april-app/ -R

Sudo systemctl restart logrotate-april.timer

Sudo systemctl status logrotate-april.timer
```

1. sudo truncate -s 101M /var/log/april-app/app.log
2. ls -l

```
mkdir /tmp/april-app
cd /tmp/april-app
truncate -s 101M app.log
```

```
sudo vim /etc/systemd/system/logrotate-april.timer

[Unit]
Description=Run logrotate every 2 minutes

[Timer]
OnCalendar=*:0/2
Unit=logrotate-april.service
Persistent=true

[Install]
WantedBy=timers.target
```

```
sudo vim /etc/systemd/system/logrotate-april.service

[Unit]
Description=Test Logrotate Service for april

[Service]
Type=oneshot
```

```
ExecStart=/usr/sbin/logrotate /etc/logrotate.d/april
```

```
sudo vim /etc/logrotate.d/april
/tmp/april-app/*.log{
	su root root
	maxsize 100M
	daily
	copytruncate
	missingok
	rotate 10
	compress
	delaycompress
	notifempty
}
```

```
sudo chmod 755 /tmp/april-app -R

sudo systemctl restart logrotate-april.timer

sudo systemctl status logrotate-april.services

cd /tmp/april-app
sudo truncate -s 101M /tmp/april-app/app.log
ls -l
sudo truncate -s 101M /tmp/april-app/app.log

sudo truncate -s 101M /tmp/april-app/app.log
ls
```

——
Shell || 25th May 2025
——

1. Recape
2. Logrotate
3. Shell scripting

What is script?

Movie example?

Director Writer —> Completed —> Imagination —> Write --> 100% —> Failure —> Box —> Scripting —> base strong ->

Subject fail —> Movie —> Script

DevOps —> Audience —> Script —>python power shell or Bash —> Script —>

Basic —> Why to Script

Squence of command

File create in /tmp directory and dump data

```
#!/bin/bash
Cd /tmp
Touch file.txt
Echo “Thes file dump data” >> file.txt
```

Sh script.sh

If you know the th3e command of linux —> Scripting
Mkdir
Touch

Awk
Sed

Loops
Conditions

#!/bin/bash —>

# comments —> Will explain use of the script

! —> not

#!/bin/bash —> /bin/bash —> sh

SHell Scritpts

Born against Shell

Lab1

1. mkdir bash-scripts
2. cd bash-scripts
3. vim lab1.sh

   #!/bin/bash
- #This script will create new directory under /tmp and it will write the the data into that
- #Author DevOps Team
- #Date 25th May 2025
- #Creating directory
- adfadfasdfasdf
- mkdir /tmp/lab1-march
- #Create new file
- touch /tmp/lab1-march/file1.txt
- #adding data to file

   echo "This the 1st content on our 1st lab on Shell scripting" >> /tmp/lab1-march/file1.txt

https://codeshare.io/5woAj5

logrotate rotates the logs, but the application still writes to the old log file (which gets deleted). how do we fix it ?

logrotate rotates the logs —> true

, but the application still writes to the old log file —> False

(which gets deleted). how do we fix it ?

Create new file —> Copy

We are able to 1st Script

#!/bin/bash
/bin/bash

/usr/bin/bash

LLM —>

LLM OpenSOurce —> DevOps —>
React
APi
ChatBot

Shell Scritpt

Variable —> Vary —> Changing in nature —>

Variable != Fixed

DevOps

Scripting —> Dev—> Local

QA —>
Env —

Change in own location —> all over script

Key=value

$key

value

devops1=om

Echo $devops1

OM

De

Variable always —> Temp —> RAM

PreDefined —> Linux —> System provided variable

SHELL

Custom —> Key value
Small

key=value
echo

Never ever keep the random name for variable —>

Abc=file
xyz=

Lab2 —> Variable Lab

1. Create new lab2.sh under same
2. Write the script
3. ./lab2.sh
4. ls /tmp/lab2-march

Variable

Variable num

key="/tmp/"
key=2

Recape

1. Logrotate
2. Shell scripts
3. Basic script
4. Shebang
5. Variables
6. String

Home Work

1. SHell scripting on Variables
2. Shell basic
3. Form use case

——
——
Shell Scripting || 31st May 2025
——
——

1. Recape
2. Loops
3. If conditions
4.

Q1. What is the correct way to declare a variable in a shell script? a

key=value

Key=

+
-
/
%
=
**
*

Great then. -gt
Less then -lt
Equal to -eq

#Plan of action
#Create a new file with vim lab6
#Write the code

#Save the file with :wq
#chmod u+x lab6.sh
#bash lab6.sh

jump.sh —> jump sever

Conditons

Weekend —> Classes —>

Condition

If

Fi

If

Else
Fi

If
elif

elif

else

Fi

-gt

-ls.

-eq —> ==

AWS Code —> == !=

Can you provide an example of how to check if a critical service is running using an if-else condition?

```
Systemctl status critical.service
If [ echo $? != 0]

        echo “Send the notification or slack”

else
        echo “services is working fine"



If
Fi

If
Else

Fi


If
elif
```

Deployment —> 6 env —> 6 files —> jenkins jobs

Argument based input
Env=uat

1. SSH to the server
2. git hub clone
3. Copy the env file database credentaisl database
4. Sudo systemctl restart services

```
#!/bin/bash
#Author DevOps Team
#Date 31st May 2025
#This script will get the code from git and deploy to all the env for application frontend
#This script will deploy on qa dev preprod uat and prod
#Variable
env=$1
git_url="https://github.com/devopsdecode/nov-2024-pub.git"
path_dir="/tmp/app"
echo "This deployment starting on $env"

#checking whether the path is present
if [ -d $path_dir ]
```

```
then
	echo " $path_dir is present"
else
	mkdir $path_dir
	echo "Creating new directory $path_dir"
fi
#Starting the deployment
if [ $env == "qa" ]
then
	#ssh devops@192.160.0.1
	touch $path_dir/qa.env
	cd $path_dir && git clone $git_url

elif [ $env == "uat" ]
then
	#ssh devops@192.160.0.2
	touch $path_dir/uat.env
	cd $path_dir && git clone $git_url
elif [ $env == "dev" ]
then
	#ssh devops@192.160.0.3
	touch $path_dir/dev.env
	cd $path_dir && git clone $git_url
elif [ $env == "prod" ]
then

	#ssh devops@192.160.0.4
	touch $path_dir/prod.env
	cd $path_dir && git clone $git_url

fi
```

Add step to delete the git clone file and env file delete

Execution

———

———

Shell Scripting || 1st June 2025

———

———

1. Recape

2. Loops
    1. For
    2. While
    3. Real world example

Loops —> Hyper Loops —>

Train Public —> Thane —> Panvel —> Panvel Thane

CSTM —> Karjat —> Loops —> Class —> Ghar —> XXX

Repet —> Any repativtiy —> Loop

For loop
While
Until

Example —> Hotel —> Housing keeping —> Sunset —> light ON

Sun raise —> Light off —>

IST —> Calculation —> Raise —>

Python

IOT

SHell Scripting

```
for  variable in variables

Do
	light on

done

For variable in variables

Do
	light off

done
```

```
find "$path_dir" -mindepth 1 -maxdepth 1 ! -name "$env" ! -name "$env.env" ! -name "$repo_name" -exec rm -rf {} +
```

prod.env —> current

Db password —> Changes —> exist —>

Rmdir /tmp/app

ANsible —> Serial deployment

Loops

Real —> log backup

Cronjob -

Panvel —> End —> For

End —> While

Keeping on counting numbers —> 1 +1 2+1 3+1 —>

For loop with Actual use case

Loops Backup of file
/var/log/

Comparse —> tar -cjvf tar.tar /var/log/nginx.log

Cronjob —> Scheduled feature —> Linux —>

Cron job one user

GLobal —> /etc/cron.d/—>

Devops —> crontab -e

0 h w day tar
10+

Github.com —> Labs

———————————————————————————————————————
———————————————————————————————————————
Git || 7th June 2025
———————————————————————————————————————
———————————————————————————————————————

1. Recape
2. GIt
    1. Basic
    2.

GIT —> VCS
CVCS
DVCS

GIT LAB —>
1. Company
2. High Secure

GIT HUB SAAS —> public
Bitbucket

Repostiroiy

ROBOT

2.0

Git lab1

1. Mkdir local-git
2. Cd local-git
3. Git init —> own local repos. All the git activity
4. ls -a .git
5. File —> files untracable —> add to staging
6. Git add . —> stage
7. Git commit --> Sha Commit id

GIt push towards —> Github

Git lab 2

1. Create new repository in gihub

SSH —> User/Password
GitHub —> User and password

2023—? XXXXXXX CLi or SDk

CLI —> Command
PYthon —> modules

Access Token

SSH Key based login or clone or push

SSH

Pubic Key
Private Key

SSh-keygen

Generate key using

1. Ssh-keygen
2. id_
3. cat .pub

Clone the repo to new directory

1.

Authentication

Git clone

1. Password
2. Automation

Lab3—> Clone remote repository

1. cd ~/git/
2. Mkdir GitHub-common
3. Cd GitHub-common
4. Git clone git@github.com:devopsdecode/april-2025.git
5. Git branch
6. Git log
7. Git checkout -b <username-branch>
8. Git branch
9. Create new file <username-branch>.txt
10. Git add .
11. Git commit -m “ <username-branch> new file added”
12. Git push origin “<username-branch>”

——
——
Git || 8th June 2025
——
——

1. Recape
2. Git
    1. Push
    2. PR
    3. Git rebase

Lab 4:- Branching with Pool request

1. Create your own branch
2. Create with same file name and put some content
3. Raise the pull request
4.

WHy PR and branch

Payment —> —> or
1. Deb —> A
2. Cred —> B
3. UPI
4. Cash
5. EMI
6.
7. Loan

Lab5 —> tag creation

1. Got to main devoops
2. Tag
3. Click on release
4. Click on draft new release
5. Give release name as your_name1.0.0

Git fetch

Git pull vs git fetch

Git fetch
Git pull
Git switch
Git merge

Git stash —> majic

Git stash —> 6

1. Create new file in working space
2. Work on your own branch
3. Git add .
4. Git status
5. Git stash
6. Git stash list
7. Git stash pop

——
——
Git || 15th June 2025
——
——

1. Recape
2. Git advance
   1. Rebase
   2. Revert
   3. Reset

Rebase

RRR

Rebase —>

Base —> Main branch or the branch

Lab 7 on git rebase

1. Checkout with sprint branch
2. Git pull origin sprint

3. New branch create dev7
4. Start creating 10 line of code
5. Git rebase main

Git revert —> Commit

For example commit 10 + commit —>

Git revert —> It will help in reverting to specific commit id —> 1 to 48 —> New commit head

Zero loss of commit id
Data loss

Lab 8 —> revert

1. Create new commit
    1. Create new file
    2. Git add .
    3. Git commit -m “New git revert lab”
2. Git log
3. Git rever commit-id
4. git status
5. Git log

Jenkins CI/CD —> After working —> DevOps —> Dev.

1. Jenkins failure —>
2. Revert commit —>
3. Commit history —>
4.

Revert

1. Create new file1
    1. Line no.1 —> commit
    2. Line no.2 —> commit
    3. Line no.3 —> commit

        4. Line no.4 —> commit
    2. Git revert #Line no.1 —> commit
    3. Ls
    4. Cat file1

———————————————————————————————————————
———————————————————————————————————————
Git || 21st June 2025
———————————————————————————————————————
———————————————————————————————————————

    1. Recape
1. Git advance
    1. Reset
2. Cloud

RRR —> Rebase , Revert and Reset —> Commit and Commit history

Reset —> Commit —>

Soft
Mixed
Hard

Hard —> You should never run on GitHub remote

Mixed —>

Git reset commit-id

9f3beed (**HEAD** -> **devops-new-7**) Revert "added melbinFile.md" —> HEAD
6b68c71 New commit for testing git revert
a8ade81 (**tag: yash1.0.0**, **tag: vcheif1.0.0**, **tag: v1.0.0**, **tag: tisha1.0.0**, **tag: takshak1.0.0**, **tag: takshak**, **tag: shubham1.0.0**, **tag: samir1.0.0**, **tag:**

pratik_1.0.0, **tag: om-V1.0.0**, **tag: nitesh1.0.0**, **tag: monika1.0.0**, **tag: melbin-v1.0.0**, **tag: Vivek1.0.0**, **tag: Santo1.0.0**, **tag: Ashu_v1.0.0**, **main**, **devops-branch-1**) Merge pull request #22 from devopsdecode/sprint-100
4e8e750 Merge pull request #18 from devopsdecode/Ritik-branch —>

Mixed

POA
1. Go the repository all the commit
2.git log —oneline
3. Note down the commit
4. Git status
5. Select your desired commit-id
6. Git reset commit-id —mixed
7. Git status

```
8e0c23a (HEAD -> devops-new-7) After testing --mixed
4e8e750 Merge pull request #18 from devopsdecode/Ritik-branch
c23c6ed (origin/Ritik-branch) Ritik-branch.txt new file added
dac37c4 Merge pull request #8 from devopsdecode/tishapacheco-branch
c83b958 Merge pull request #5 from devopsdecode/tishapacheco-branch
c87f38b Update file3.txt
8a615a0 modified abc.txt
73deaab monikaspawar1 branch added
46c0f53 tishapacheco branch added
0f4746e Initial commit
```

```
 git status
git rm abc.txt
git status
 git add .
 git commit -m "After testing --mixed"
 git branch
 git log
 git log --oneline
 git status
 git reset c83b958 --soft
 git status
 git log --oneline
```

```
01b52b2 (HEAD -> devops-new-7) After testing --soft
```

```
c83b958 Merge pull request #5 from devopsdecode/tishapacheco-branch
c87f38b Update file3.txt
73deaab monikaspawar1 branch added
0f4746e Initial commit
```

```
ls
Ashutosh.txt  Vivek.txt       nitesh.txt     santosh-branch.txt      testing-stash.txt
vaibhavcheif.txt
README.md     devops-branch-1  om-ghag18.txt  shubham.txt
testing.txt        yashyb.txt
Ritik-branch  file3.txt        samirfile.txt  takshak-new-branch.txt  tisha.txt
```

```
 git log
git status
git add .
git commit -m "After testing --soft"
git status
 git log
git log --oneline
git status
  git reset 0f4746e --hard
 git status
ls
git log
```

Amazon Q

Covid —> Online —>

1. Axis

VT —>

Type2 —> Virtual BOX

Type1 —> Xen —> Cloudprovide

OPenStack. —> Terrafrom

Example —> Metro vs Urban Area

Virtual World —> 9

2025

AWS —> 29% —>
Azure —> 19%
GCP 15%

Shell Script :- Git local changes backup

1. Create new branch only if any change in current branch
2. Write a script to create new branch based on template
3. 5PM Evening it should
    1. Git add
    2. Git commit
    3. Git new branch
    4. Git push new branch

———————————————————————————————————————
———————————————————————————————————————
Cloud || 22nd June 2025
———————————————————————————————————————
———————————————————————————————————————

1. Recape
2. Cloud
    1. Budget
    2. IAM
    3. EC2

India —> DevOps —> World Wide —> Application country —> Copy

Region AWS —> Company create —>

JIophone —> FE or —> Latency

USA —> Region —> XXXXX

Laws —> Commodity -> Ba

Latency

AZ

HA

Server —> Goes —> application is down Letency

7 sec —>
 3. Sec

Region
AZ
CLoudFront POP CDN

Budget
IAM Services —>

1st Lab —> AWS —> Budget
Budget —>

1. Budget
2. Template
3. Provide the name of budget
4. Click on Create
5. Configure the email
6. Email notification
7. Subscribe

IAM—>

DevOps

I —> User to AWS or Azure Or —> Create new user
A —> Policy —> That user —> EC2 —> Delete Instance —> please Dev
M —>

Least Privlegdegt access —> AWS —> Production —>

Administrative —>

MFA —> AWS —> USer —> MFA —>

Social Networking —> Insta —> FB —> WhatsApp —> MFA

Public Network —>

Lab2 —> MFA for root account

Lab3

1. Create new devops-team group
2. Policy ec2fullaccess
3. Users create new user devopsuser1
4. Select group devops-team
5. Go to back to that enable console login
6. CSV
7. New browser edge
8. Login
9. MFa active from root account user
10. IMA user logout and try login I should

Azure
GCP

Create IAM User

Cloud || 28th June 2025

---

1. Recape
2. Cloud
   1. EC2
   2.
   3. S3

Chrome —> Root user —>

Safari —> Edge —> IAM

EC2 —> Elastic Compute Cloud
VM —> Virtual machine
GE —> Google Engine

Type —> CLoud

IAAS —> Infra as a services
PAAS —> Platform as a services
SAAS —> Software as as services
    Google
        SIgn —> android -> Gmail —>

SMTP —> Server
Exchange —>server
Public IP
Domain
N

Email —> Company email domain —>

Subscribe —>

Platform as as services —> AWS —> Elastic Beanstalk —>

Just write a code —> build -> tar —> upload

Ec2
ALb
Storage
HA
Load balancing

IAAS —> DevOps engineer —> UI or SDK or IAAC —> CLoudfromation

Server
Storage

NEtwork

OSI

TCP —>.

UDP

IAAS —> EC2

1M —> 1000Server

2M —> 1500Server

100 Servers —> 20 —> Quota limit

SSL —> Offloading

ILO —> Management —>

Lab4 —> EC2 instance creation

1. Search for EC2
2. Ubuntu 24
3. Select the instance type with Free teir t2.micro
4. Create the instance

#D

Ssh-keygen —> Linux

User data - optional

Website —>

sudo yum update

Sudo yum install
Sudo git clone

POA —> 6 Lab

1. Create new amazon ami
2. T2.micro
3. Key pair
4. SSH to the server
    1. Terminal or git bash
    2. Cd ~/Download
    3. Chmod 400 <test>.pem
    4. Ssh -I test.pem ec2-user@<iP>

Home —>

Azure —> VM
GCP

——

Cloud || 5th   July 2025

——

1. Recape
2. Cloud
    1. Azure and GCP service
    2. S3
    3. SNS
    4. SQS
    5.

SImple Storage services —> S3 —> AWS —> 100% —> S3

Set of file formate —>
Onedrive —> SQL XLS mov zip and known —> SH —> 1B—

S3 —> Unliminted files — 3TB

3TB —>

B —> KB —> MB —> GB —> TB —> PB —> ZB —> TB —

S3 Tables

S3 —> Uniq name —> ALL AWS --> all region —> az

Testing-s3-april-2025

Lab —> S3—> 10

1. Create new s3 bucket
2. Name should be unique
3. Create new file and upload it
4.

1. netlify.app/
2. vercel.com
3. mellowbricks.co.in
4. nirankariduniya.blogspot.com
5. https://omghagresume.s3.ap-south-1.amazonaws.com/Resume_webpage/index.html

Data dump —> Indexing —>

1. Type
2. Name
3. Age
4. Key

GCP —>
Azure

Azure —> RG—> Resource Group

AWS —>

GCP —> project

GUI —>Based

CLI

DevOps ClI

1. USer/password
2. SSH
3. Token
4. Access key and secret key —> CLI or SDK

PYthon —> ROR -> Developer —> S3

Real —> EC2 —> S3 —> Fetch data or file -

1. IAM user
2. Credentails generate key
3. AWS CLI
    1. AWS CLI laptop
    2. Configuration
    3.

    API —> They never change

    FE —> UI change

    Lab —> Access key and Secret Key
    1. Generate the access via console for respective user
    2. Teriminal
        1. Aws configure
        2. Update the access and secret

3. Update the policy for that user s3 full access
4. Aws s3 ls

Project :-

1. IAM Role
    1. Name testing-role-based
    2. Policy add as s3 full access and ec2 full access
    3. Trust ec2 policy
2. EC2 instance
    1. Name
    2. Instance type
    3. Ami
    4. Security Group
    5. Advance IAM role
3. Login the the server
    1. Install aws cli
    2. aws s3 cp ./testing.tx s3://testing-s3-april-2025/testing/

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

AmazonEC2FullAccess	AWS managed	2
	AmazonS3FullAccess	

——————————————————————————————
——————————————————————————————

Cloud || 6th July 2025

——————————————————————————————
——————————————————————————————

1. Recape
2. Cloud
   1. VPC
   2. SNS
   3. SQS
   4.

VPC —> Virtual Private Cloud

Ec2 —> change the default VPC —> AWS

10.0.0.0/16 —> CIDR Value —>

Subnet mast]

Subnet —> Division of network —>

Public Network —> google.com —> public —> routable —>IGW —>XXXXXXX —>

Private —> VPC —> Private VPNC

Private network —> Database —>Private subnet

Lab

1. Create own vpc
2. Create four subnet with two public and private 2

POA
1. Create Internet Gateway
2. Attache IGw to VPC
3. Create new Public Routable
   1. 0.0.0.0/0 with internet gateway

4. Select public subnet
5. Attach the routable
6.

NAT Gateway —>

1. EC2 instance using public subnet
2. SSH to that server
3. Copy the ssh key to that
4. SSH to new create private subnet internal 10.0.0

1. Private Subnet
2. NAT gateway
3. Routable
4. Attach the NAT
5. Create new EC2 instance
6. Curl -v google.com

——
——
Cloud || 6th July 2025
——
——

1. Recape
2. Cloud
    1. CloudWatch
    2. SNS
    3. SQS
    4. Route53
    5. CodeDeploy
    6. API Gateway
    7. AI for DevOps Amazon Q
    8. Lambda
    9. RDS
    10.

Plafrom Engineer

Terrafrom

ANsible

Jenkins

SAAS

IAC —> Terrafrom —> OpenTofu

Cloudwatch —> Monitoring —>

Promethues and Grafana and ELK Stack —> Log monitoring

CLoudwatch —>

Monitoring —>

1. Proactive monitoring
    1. 60% CPU Warning
    2. Production 75% of CPU usage —> Ciritical
2. monitoring
    1. Production
    2. Df -h /100 disk

Monitoring
1. CPU usage —> Directory cloudwaqtch
2. Ram usage —> CloudAgent —> main cloud watch metric
3. Disk usage —> CloudAgent
4. Network —> CloudAgent
5. Process monitoring —> Cloudagent
6. Logs monitoring —> Cloudagent
    1. Appication logs
    2. System logs

200 response
504

Df -h —> SSH

Installation
Configuration
    1.CPU
2. RAM
Log

IAM role c

.json —> Keep it

System manager —> Server maintain —> specific —>

SQS —> Queue

SNS —> Simple Nofification services

Shutdown —> Start Mobile

Lambda —> SQS —> SNS

SNS —>

1. Create new SNS topic
2. Add subscription to email
3. Open the gmail and confirm the subscription
4. Publish one message from SNS topic
5. Check the gmail for the email
6.

SQS

Amazon SQS —> kafka

Producre and subscribe

Data producer and subscribe —>

SQS —> duration 14 days —> DQL

Lambda or application —> D

Serverless services —>

Server start —> weekends stop —> Aws lambda —> ec2 stop

Monday

EventBridge schedule -=> Lambda —> Ec2 instance

Lambda function —>

Def lambad handler

Use Cases —>

Statfull application —> Ec2 —> ECS —> EkS

Stateless application —> add hoc basic —>

1. Batch jobs

Laundry Application —> Public website —> Ec2 or website

Customer support —> Compllaint or exchange —> submit —> lambda job —> customer id —> SQS —> Database —> SNS —> notification send

Benefits —>

1. any code lang.
2. Manager server XXXX

3. SDK
4. AWS CLI

Loss

1. 15MIn
2. Batch —>5min
3. 512MB
4. 10240MB

While —> un

Lambad function

1. Create new lambda function
2. Update basic
3. Run the execution

Home work Pipeline

SQS —> Lambda —> SNS —> Notification

——
——
Cloud || 6th July 2025
——
——

1. Recape
2. Cloud
    1. RDS
    2. Route53
    3. CodeDeploy
    4. API Gateway

SQS —> Queuing —> Multiple —> Sale Mobile —> 11th July 2025 —> BI tools —> Lambda —> SQS —> EC2 —> ECS —>

Relational Database Services

SQL

1. Table—>
2. Row —>
3. Colomn —> mell

MySQL —> MariaDB
Postgresql
AroraDB
Oracle

NOSQL

1. Key value —> Name = Mel

Mongodb
Dynamodb
Cassandra
Cosmos

DevOps
1. Create the database cluster
    1. Storage Class
    2. Instance type t2 series
    3. D
2. Operation
    1. Database high CPU and memory
    2. Storage full
    3. New upgrade database engine
    4. Security Group
3. Backup Snapshot
4. Database dump
5. Audit logs

Masterdim2025

Create new database from AWS RDS for PostgreSQL

1. Default
2.

Database

Ec2 instance —> Database

1. Create new instance on the VPC under public subnet
2. Install sql tool
3. Sql shell

EKS
Ecs

Lab EC2 connect with database

1. Create new instance make same VPC
2. Create under public subnet
3. Ssh to the server

```
    1  telnet  april-database-1.cn2g0kaqmpre.ap-south-1.rds.amazonaws.com
5432
    3  telnet  april-database-1.cn2g0kaqmpre.ap-south-1.rds.amazonaws.com
5432
    4  sudo apt update
    5  sudo apt install postgresql
```

4. Open the database and go security group
5. Add new rule and ip address should from you server private ip
6. Save it
7. psql -Upostgres -hapril-database-1.cn2g0kaqmpre.ap-south-1.rds.amazonaws.com

AWS route 53 —> SSL —> ACM —> Only Inside the AWS —> Cloudfront —>

Code Pipeline —> Jenkins -

CodeCommit —> Github —> Private repository

Build —> Run build

Artifacats —> vlc.xec postgtresql.tar —>
Code Deploy —>

1. Blue and green deployment
2. Roll back

Cost higher code deploy

Amazon q —> Summit —>

Amazon q —> builder Id

Ubuntu
Or Mac

WSL

Jfrog
Nexus

mvn

Npm
Gradel

Code deploy —> Copy the server —> Jenkins Plugin —> Jenkins

DC —> Active

DR —> Region
———————————————————————————————————————
———————————————————————————————————————

Cloud || 19th July 2025

AWS:-

1. AI for DevOps Amazon Q
2. Cloudformation
3. Load balancing
4. API gateway
5. Comparing

Builder id —> amazon login

Ubuntu —> Amazon q —> AWS

Cop —> Azure

Warp

Amazon q with you local
Doing the exercise the

N8n

Amazon q

1. Bulder create —> https://docs.aws.amazon.com/signin/latest/userguide/sign-in-aws_builder_id.html
2. Download Amazon q on Ubuntu server
3. Configuration aws cli for access and secret —> IAM User

Cloudformation —> IAAC —> Amazon —> Create and maintain

Ec2 —> Security Group —> Cloudfromation —> configuration update and creating new resources

Terraform —> OpenTofu —> Free

User —> 10 Server —> dev env —> API Servers

Template —> Stack —> change one file —> Update —> 10 server —>

User —> 10 Server —> staging env —> API Servers

Cloudformation

1. Create new cloudformation
2. Create the stack
3. Infrastructure composer
4. S3 —> change the name
5. Check the s3 bucket
6. Two bucket

Cloudfromation delete —>

Load balancing —>

Shitft —> EC2

Application Load Balancer —> Target group —> Ec2 instance or EKS or ECS target

1. Create new Application load balancer
2. Create new target group
3. Create new instance
4. Allow the security group between ALB to Ec2 instance
5. ALb secrutirty 80 to 0.0.0.0/0

Home Work :-
1. Docker desktop is working fine
2. Docker hub

———
———
Docker || 20th July 2025
———
———

1. Docker
2. Microservices
3. Docker basic
4. Dockerfile
5. Container
6. Image

Docker —>

1. Container
2. Containerise the applicationce
3. Swarm
4. Dockerfile
5. Combination
6. Small virtual env.
7.

Docker —> Just company —> Container

Container.d —> Runtime

Podman

Image —>

OCR standard —> Best container —>

Runtime —> Docker engine —> Desktop

Docker —> Virtual machine —>

Bare-metal Server —> Ubuntu —> Nginx/ Pythong/ Database —> XXXXX
16GB

2GB

Virtual machine —> 16GB 1GB 2GB 4GB

4GB —> Server —> nginx webserver —>

Nginx —> 1.18 —> 21.

unInstall complete

Python —> Lib —> 2.7 —> 3.12

2.7 —> 3.01 —> 3.12

Sysadmin / DevOps —>

Docker / container —>

Blue Print —> Dockerfile —> all dependency —>

Server or laptop

E.g.

Dev —> Python Django —>

20th July —> KT —> Sn dev —>

1. Laptop Windows
2. Basic
    1. Visual studio
    2. Python 3.14
    3. Database
    4. Chrome
3. Application server run
4. Database run
5. FE
6. API
7. Local env setup

1st line of code

Server —> Dev

1. Server

2. Basic
    1. Python 3.11
    2. Database
    3. Vim
    4. git
    5. zod
3. Application server run
4. Database run

1st line of code

Server —> QA

1. Server
2. Basic
    1. Python 3.11
    2. Database
    3. Vim
    4. git
    5. zod
3. Application server run
4. Database run

1st line of code

Server —> Prod

1. Server
2. Basic
    1. Python 3.11
    2. Database
    3. Vim
    4. git
    5. zod
3. Application server run
4. Database run

1st line of code

Docker —>

1. Laptop
2. Visual
3. Docker desktop

4. Docker compose
5. Docker compose up

10th Day —> Github

1. Docker image —> Github —> Docker hub —> clone /pull

Docker image —> combination —> all decency and code

Docker image —>

Blue print —>

Dockerfile —> Classes

    Laptop
    water
    laptop
    mobile

Karjet —>

Take

1. Resources
2. Depends installation
3. Isolation

1st lab docker

1. Get the ready made docker images
2. Run as a Contianer
- docker pull nginx
- docker pull nginx:latest

2nd start the docker image as container

docker run nginx

Image —> Windows + docker desktop —> Python or for —> Windows

Windows .dot application —> windows —> Linux XXXXXXX

1. Docker pull —> to get the image from docker hub
2. Docker images
3. Docker run nginx
4. Docker run -d nginx
5. Docker run -d -p 80:80 nginx
6. Docker stop <container-id>
7. Docker rm <Container-id>
8. Docker rmi nginx
9. Docker logs
10. Docker inspects
11. Docker rm $(docker rm -a -q)
12. Docker start
13. Docker run -d (interactive mode)
14. Docker run -ltd -p 80:80 nginx
15. docker exec -it 44eaed06d3b4 /bin/bash
16.

——
——
Docker || 26th   July 2025
——
——

1. Recape
2. Dockerfile
    1. Container
    2. Image

Container —> a process —> Stop —>

Kiill

Coding part one docker —>

Dockerfile —> docker engine

-f

FREE Container

F—> FROM
R—> RUN
E—> EXPOSE
E —> ENTRYPOINT
C —> CMD

Deploying the own docker image in docker hub

1. Dockerfile
2. Docker build . -t devops
3. docker login
4. Docker push

ENtrypoiint —> Hard coding —> Nginx —> run —> image as container —> nginx

Process and exit

Application port and Dockerfile —> Should be same —> Dev

3012

3306

Lab

1. Create a docker file
2. Docker build . -t devops-base-nginx
3. Docker run -d -p 8080:80 devops-base-nginx:latest
4. Chrome localhost:8080

Docker image

Website Static vs dynamic website

Website staiics —> From filling page —>

I wanted to do fill the form

Name
Age
Married/Unmarried
Nal:-

FE —> API —>DB

Admin UI —> See the content

FE —> Website —> Our

Context :- Create a website for me for “docker basic commands for devops engineer”
Output :- Provide the output in index.html format

Lab2 Dockerfile with own website

1. Get the index.html and place it in specific location where the Dockerfile is present
2. Update the Dockerfile
3. Docker build . -t docker-website:v1
4. Docker run -d -p 8080:80 docker-website:v1

Docker image ready in your laptop —>
Nginx —> docker pull

Public our docker image

Two
1 public image
2. Private image

Home Work :-
Publish image

1. docker tag docker-basic-website:v1 devopsdecode/april-2025:v1
2. docker login
3. docker push devopsdecode/april-2025:v1

docker pull devopsdecode/april-2025:v1

——
——
Docker || 26th   July 2025
——
——

1. Recape
2. Dockerfile
    1. Container
    2. Image

Container backup
1. Start the container
2. Stop the container
3. Take container id
4. docker commit container id <image-name>

Multi Stage Dockerfile

Nodejs Application on Docker

1. Git clone git@github.com:devopsdecode/april-2025.git
2. Git checkout docker-labs
3. Cd lab3/**welcome-app**/
4. Docker build . -t nodes-app:v1
5. Docker run -d -p 3000:3000 nodes-app:v1

DevSecOps

1. Wash
2. AWS security Hub

1. Nginx —> static
2. Nodejs —>

Docker volume —> PVC —> AWS EFS —> NFS —>

Docker run. -v source:/var/www/html nginx:webserver

Update tehe co

Lab —>

1. Dockerfile
2. Index.html
3. Get the exact path of your current location pwd
4. docker run -d -p 80:80 -v <change the path here>:/var/www/html/ basic-website:v1
5. Chrome open localhost:80
6. Update the code
7. Chrome open localhost:80
8. docker exec -it bd6211be76db /bin/bash

Home Work

1. Multi Stage docker for python application

———
———
Docker || 2st Aug 2025
———
———

1. Recape
2. Docker compose
   1. YML
   2. Docker yml syntax

YML —>Yet Another mark up language —>

YAML —>yml

XML —>

YAML —> Content —> Agentic AI —> Read —> write XXX —>

Read
Write
Debug

Basic

Story —>

Class going to back call from Mom or Spouse

1. Veg Bhendi

Key =
Veg =Bhendi
Fruti = Banana

Veg ?

1. Palak
2. Bendi
3. Pudina
4. Karela
5. So ??/

Fruit
   1. Banana

YML

Array of List

```
Fruit:
   - Banana
   - Mango

   Veg:

   -  Palak


  –  Rice
  –  Dal
  –  Wheet
  –  Chena
```

Diet new year

```
Fruit:
Banana
           FAT: 20
           Sugar: .2
   - Mango
          FAT: 5
          Sugar: 14

   Veg:

   -  Palak


  –  Rice
  –  Dal
```

- Wheet
- Chena

Key = Value

List

Dis.

```
Key: value
 List:
   - Banana

   Dis
   - List:

          :
```

Docker Compose —> to run multiple different container in one network —> like FE , BE and DB

Docker create network

1. Create a docker compose file
2. Write to container details
3. Docker-compose build
4. Docker-compose up -d

Goos
Crush

Three teir app
1. Copy the code to new directory

2. Docker-compose up -d
3. Docker-compose down

Docker container —> Container —> Micro services —>

HA
FT

FE-admin —> Client Mic app slido app

In memory database —> RAM —> Change —>

PostgreSQL —> Hard disk

Flask
Gunciron
Django
FastAPI

Home Work

1. Create new folder
2. Clone the repository
   1. git clone https://github.com/dockersamples/example-voting-app.git
   2. cd example-voting-app
3. Docker-compose up -d
4. Docker-compose ps
5. Docker-compose ps
6. Chrome localhost:8080 —> Voting
7. Chrome locahost:8081
8.

Home Work 2

1. Create new server with arm64 AMI selection
2. T4g.nano
3. AMI Ubuntu
4. SSH

5. Sudo apt update -y
6. Sudo apt install docker.io
7. Sudo apt install docker-compose
8. Create new folder
9. Clone the repository
    1. git clone https://github.com/dockersamples/example-voting-app.git
    2. cd example-voting-app
10. Docker-compose up -d
11. Docker-compose ps
12. Docker-compose ps
13. Chrome localhost:8080 —> Voting
14. Chrome locahost:8081
15.

——
——
Docker || 3rd Aug 2025
——
——

1. Recape
2. Docker

```
 minikube start   --driver=none   --apiserver-ips=172.31.40.165 ,13.201.49.250
--apiserver-port=6443 --driver=docker
```

Agent configuration on Jenkins with Keys

```
ssh ubuntu@13.201.49.250
sudo mkdir -p /home/ubuntu/jenkins
sudo chown ubuntu:ubuntu /home/ubuntu/jenkins
chmod 700 /home/ubuntu/jenkins
Sudo apt install openjdk*
```

ONE IMAGE —> MULTUPLE CONTAINER —> MULTI STAGE

DOCKER IMAGE —> DOCKERFILE

DOCKER CONATINER —> IMAGE —> NOT LOCAL —> DOCKER HUB

DOCKERFILE —>DOCKER BUILD . -T DOCKER-IMAGE

DOCKER RUN -p docker -image

Docker-compsoe

1. Clone the repository
    1. git clone https://github.com/dockersamples/example-voting-app.git
2. cd example-voting-app
3. docker-compose build
4. docker-compose up -d
5.
6.

Vertical
Horizontal

Monday —> 100
Sunday —> 10

Scale —> Horizontal

Train CoaCHS —> 9

CURRNE T—> 15

Vertical

DB —> 1

DB —> 3

Horizontal scaling

T2.micro —> 1 server ==> t3.largee —> Vertical

T2.mirco —> 2 Server —> horizontal

Home Work

1. Build the docker image using buildx
2. Prompt to Gemini
3. Nodejs
4. Gemini CLI
5.

———————————————————————————————————————
———————————————————————————————————————
K8s || 3rd Aug 2025
———————————————————————————————————————
———————————————————————————————————————

1. Recape
2. Kubernetes
    1. Basic
        1. POD
        2. Replicaset
        3. Replicacontroller
        4. Deployment

Kubernetes —>. Microservices
What is k8s

1. Contianer org.
2. Multiple Container
3. Scale up and down
4. Load Balancing
5. Self Healing /auto healing / FT
6. Alerts

K8s
Microservices —> K8s —>
Breaking down into single —> Service

Bigger application —> Amazon.com and google.com

Monolith Vs Microservices

FE + API + Two + —> Ec2 based
Nginx
Apache2
API —> node issue
Same server

Minikube —> Lightweight Kubernetes single node cluster

Pod —>

Key= Value
Array
Disc

Code Part —>

AKMS —>

A===> apiVersion
K ===> kind
M ===? metadata
S ==> Spec

Kubectl get pod —> default
Kubectl get node
Kubectl get svc
Kubectl get namespace

Kubectl get pod -n kube-system

Home Work
Container for monitoring

——
——
K8s || 10th Aug 2025
——
——

1. Recape
2. Kubernete
    1. Replicaset
    2. Deployment
    3. Svc

docker network ls

POD

1. One main container and sidecar in one pod
2. Kubectl delete pod nginx-pod-multiple
3. It will one server went down
4. FT
5. POD Downtime
6. Scale ?

ReplicaSet in the ReplicaController

Replicase 3 —> 6
1. Image —> nginx1
2. Node —> nodeexport2

Kubectl apply -f replicaset-defination.yml

Update the replicaset with image change on both

Nginx1
Nodeexpoter1

Kubectl apply -f replicaset-defination.yml

Kubectl get pod
Kubectl get replicasets
Kubectl delete pod nginx-pod-relicaset

New pod with new image —. Imagepullback error

POD

Home Work

1. Create new repliaset for Nodejs application
2. API monitoring state hopsoft/graphite-statsd
3. New replicaset

——
——
K8s || 16th Aug 2025
——
——

1. Recape
2. Kubernete
    1. Deployment
    2. Services
        1. ClusterIP
        2. NodePort
        3. LoadBalancer

AKMS —>

apiVersion: apps/v1
Kind: Replicaset
Metadata:

Spec:

Selector
Replicaset 3

Node Request —> Response —>

User1 —> Node1 —> pod1 —> Session —> refresh —> Pod2

pod2 —> running status —> ALB —> Register —> traffic —> User2 —> pod —>

Blue and Green

DR to PR —> No release —>

Rolling update

Apply ek ap

Canary update

1000 users/4

250 —> divert wait for sometime

250 —> new version

500 —> 250 new diver

250 —> new

100% Deployment

Rolling update

10 server

2 server v2 —> deployed —> Canary wait revert deployment stop

1 lakh —> 10 server —> blue

20 server —> Green

Jenkins —> Green —>QA —> Poiting —> LB —> Send new

Deployment

1. Rolling update
2. Blue and Green
3. Canary

Lab —> Deployment

1. Create new deployment for rolling update
2. Create own content for deployment
3. Kubectl create -f deployment.yml
4. Changes the image id
5. Kubectl apply -f deployment.yml

Kubdeadm —> based installation —> k8s cluster

AWS —>Bedrock

Sagemaker —> own Model

Bedrock

CastAI

Grafana — AIOps

Services —> Networking —>

Cluster ??? —> Group of node, group pods or groups of server

Minikube —> minikube start —> cluster —> Services —> ClusterIP

Lab —> Cluster IP

1. Create new services file for internal networking within the k8s cluster
2. Kubectl create -f deployment.yml
3. Kubeclt create -f services.yml
4. Kubectl get svc
5. Kubectl get pod

——
——
K8s || 23rd Aug 2025
——
——

1. Recape
2. Kubernetes
    1. NodePort
    2. LoadBalancer
3. EKS readyniess

NodePort —> bring network —> Minikube / Docker k8s —> Laptop Chrome

#AKMS

apiVersion
Kind: service
Metadata

Spec:

  type: NodePort
  targetPort
  port
  NodePort: 30000 to 32623 —> nodeprot

Lab —> NodePort

1. Create the deployment of nginx-deployment
   1. Kubect create -f deployment.yml
   2. Kubectl get deployment
2. Create new nodeport.yml
3. New code update
4. Kubectl create -f nodeport.yml
5. Chrome
   1. http://localhost:30100/
   2. http://localhost:30080/

AWS EKS

1. Basic IAM configuration
2. Cluster

ECS —> Docker images —> as services

EKS —> Multiple

e.g. app —>
  api
  database

GKE —>

AKS —>

EKS —>

Helm Chart —> Automate Amazonq Simple cluster helm

Helm —>

EKCTL

Issue —>

1. Debugging
2. Configuration

1. IAM Configuration

Jenkins + Docker + Kubernetes —>

1. Jenkins
2. Kubctl create -f deployment.yml
3. LOgin EKS cluster

Lab

1. Cluster IAM role creation
    1. Create IAM role by selecting Cluster role

Lab —> worker-node

AmazonEC2ContainerRegistryReadOnly	AWS managed	Permissions policy
AmazonEKS_CNI_Policy	AWS managed	Permissions policy
AmazonEKSWorkerNodePolicy	AWS managed	Permissions policy

ECR

1. Create new Dockerfile in new directory
2. Create new index.html from Amazon q
3. Create new repository on ECR ap-south-1
4. Aws configure
    1. Access key and secret key
5. Take the login command from your ECR
6. docker build -t april-2025 .
7. docker images
8. docker images | grep april-2025
9. docker run -it -p 80:80 april-2025:latest

```
#FREE C create new nginx based fronent application
FROM ubuntu:latest
RUN apt-get update && apt-get install -y nginx
COPY index.html /var/www/html/index.html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

Push the same image to Docker hub

———
———
K8s || 24th Aug 2025
———
———

1. Recape
    1. Services
        1. LoadBalancer
2. EKS Deployment

Deployment —>

DaemonSet —> systemctl status —> All ways running —> Nodes —>kube-proxy

StatefullSet —> database —> make sure apple services —> Configuration —> stateful

Webserver —> Stateless —> k8s —> S3 bucket

Use case

Kubectl get namespace

Kube-system

Kube-proxy

Kubectl edit kube-proxy

Kube

K8s —> main controller —> main server

Worker node —> kubelet services —> docker services —> sudo systemctl status kubelet

Kube-proxy —> pod —

EKS —>

1.

GUI —>COnsole

Ekctl —>

AWS CLI

Terrafrom

SDK

Pulumi

AWS API —>

Lab —>
AWS EKS cluster

1. aws eks update-kubeconfig —region ap-south-1 --name my-cluster
2. kubectl get pods
3. Create new deployment file “deployment.yml”
4. Create new services file “services-alb.yml”
5. Kubectl create -f deployment.yml
6. Kubectl create -f services-alb.yml
7. kubectl get svc

Delete action plan

1. kubectl delete svc app-services-eks
2. Delete eks worker group
3. Delete eks cluster
4. Delete ear image

——
——
Jenkins || 31st Aug 2025
——
——

1. Recape
2. CICD
3. Jenkins

Helm Chart

Build and release

CI —> continuous integration

Car making —>

1. Raw Steel
    1. Door
    2. Colour
2. Raw other
3. Ready —> staging
4. QA
5. Stock yard

CI

Car

1. Prduction

Build

Test —> unit test

echo “hello world

Testing

Hello world

Print

Push

Build —>

Test syntax

CircleCI
GItlabCI
GIthubAction
Travis
AWS CodeCommit ,

JenkinsX
Jenkins —> Multi Cloud —> 2 Decodes —> Jenkins

1. Plugins
2. Mult Cloud
3. CI and CD
4. Parameter One -> multi
5. Pipeline script
6. Nexus artifact

CD —> continuous Deployment and Delivery

1. SBI bank —> User Impact more —> Ciritcal —> Trans.
2. continuous Delivery
    1. Downtime time
    2. Apporavlas
        1. QA
        2. BS
        3. Management approval

E-Com application

1.

Lab1 —> Setting up Jenkins

1. Docker base docker pull jenkins
2. Docker run -itd -p 8080:8080 jenkins/jenkins:lts

Lab2 —> Create basic Jenkins job with Freestyle

1. Create job
2. Select the style
3. Select the build execution
4. Save the job
5. Run the job
6. Console output the script

React application —. Deployment

CI

Lab2

1. Copy the job from other
2. Schedule the job for every 5 execute
3.

Jenkis CLI

GUI —>

Create a job and view the job

Jenkins CLI —> Jenkins server

——
——
Jenkins || 6th Sep 2025
——
——

1. Recape
2. Jenkins pipeline scripts

Jenkins pipeline —> PS3

Pipeline
Stages
Stage
Step

{
}

Lab3 —> Creating 1st basic pipeline script

1. Create new jenkins job

2. Select the pipeline script

```
pipeline{
    agent any
    stages "build" {
        stage "one" {
            step {
                sh 'echo testing one'
            }

        }
    }
}
```

3. Save the job
4. Execute the job

Lab 4

1. Create new junkns with clone of existing one
2. Three to write multiple stages
3. Save the job
4. Run the jenkins job
5. Make some change
6. Run the job again

```
pipeline{
    agent any
    stages {
        stage('one') {
            steps {
                sh 'echo testing one'
            }
        }
        stage('two') {
            steps {
                sh 'echo testing two'
            }
        }
        stage('three') {
            steps {
                sh 'echo three'
            }
        }
        stage('four') {
            steps {
                sh 'echo testing four'
            }
        }
        }
}
```

Lab 5

1. Add agent to your master Jenkins server
2.

Lab 6 — Source Code clone

1. Docker + Nexus as one unit
2. Docker compose
3. Jenkins + Nexus

Lab 7

1. Connectivity check between Jenkins and nexus
2. Create new Jenkins job on New jenkins server

```
 pipeline{
    agent any
    stages {
        stage('nexus-check') {
            steps {
                sh 'curl -v http://nexus:8081'
            }
        }
    }
}
```

3. Trigger the Jenkins job
4. Response 200

Home Work :- Configuration of token between Jenkins and Github repository

———————————————————————————————

———————————————————————————————

Jenkins || 7th Sep 2025

———————————————————————————————

———————————————————————————————

1. Recape
2. Jenkins CI Jobs
    1. Token —> Done
    2. Clone the repository —> done
    3. Build
    4. Test
    5. Push
3. Gradel vs Maven —> done
    1. Token —> Done
    2. Clone the repository —> done

SCM —> Downloading the source code

Authentication

1. User name password
2. SSH Key pair
3. Access key and Secret key
4. Token
    1. Git token
        1. Expire —> Unlimited —> user name password 3MOnth
        2. 30Days or 7 days 6Month

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

Lab —> Token Generation

1. Login Into github.com
2. Right side developer setting
3. Personal access tokens (classic)
4. Select all read option
5. Click on generate
6. Copy it and keep it safe

Lab —> Generate —> Configuration Jenkins

Build—> Gradel and Maven —> Java

Framwroker —> Node — npm

Build — Source —> Depenedccy

Harwader <— 0101010101010101010101010101001 <— Boot < — kernel <—
ASCII <— Language <— C++ <— Java <— dependucy <—

Build —>
Pain Text + decency install —>

Maven validate

Gradel build

Maven. build

Maven complain

java.jar
python.pyc
index.js
index.html

Workflow

Maven and Gradel —> Package builder —> application —>

CI —>

.dotnet application virtual

.exe

.dlr

.jar

Mvn --version

Home —> research the code used to change the directory before executing the steps
dir(“maven-app”)

——
——
Jenkins || 13th Sep 2025
——
——

1. Recape
2. Jenkins
    1. Build
    2. Test
    3. Nexus
    4. push

Repository

Plain text —> bytes or Kbs MB
Github
Bitbukcet

Artifact Repository??

Vivek player

Vive.exe —>

Download double start —> Deployment

1. KB , MB and GB —>
    1. Tar.gz
    2. Bin
    3. Exe
    4. Java
    5. Ts

Frog
Nexus

COdecommit AWS
Azure

Lab

Add new stage for maven validate

Home Work

1. Deploy the same application
2. Maven to grade

Maven Command	Gradle Equivalent	Notes
mvn validate	gradle tasks	Implicit validation in Gradle tasks.
mvn compile	gradle compileJava	Compiles source code, resolves dependencies.
mvn clean compile	gradle clean compileJava	Cleans build directory, then compiles.
mvn test	gradle test	Runs unit tests.
mvn package	gradle jar	Creates a JAR file.
mvn verify	gradle check	Runs tests and verification tasks.
mvn install	gradle publishToMavenLocal	Publishes to ~/.m2/ repository.
Deploy (implied)	gradle publish	Publishes to a remote repository.

____
____

Jenkins || 13th Sep 2025

____
____

1. Recape
2. Jenkins

1. Build
2. Test