② What is sre - task and responsibilities.
① What is Devops, really understand it; Devops vs sre
④ What is Devsecops.
⑤ Hybrid cloud vs multicloud
⑥. Why learn python as a Devops engineer.
③ Devops roadmap 2022.

① What is Devops ? Really understand it

Overview.

a.) Application release process.
b.) Challenges that Devops try to solve
c) What is Devops ? Devops principles
d) Devops in practice:
Devops as a seperate role, what are the task and responsibilities.
e) Devops vs sre : How it fits in whole devops process.

Devops is an intersection between Development and operations

Where do boundaries of Devops start and end?
Which part of Dev is not Devops ?
Which part of operations is not Devops ?
Why was there a need between those?

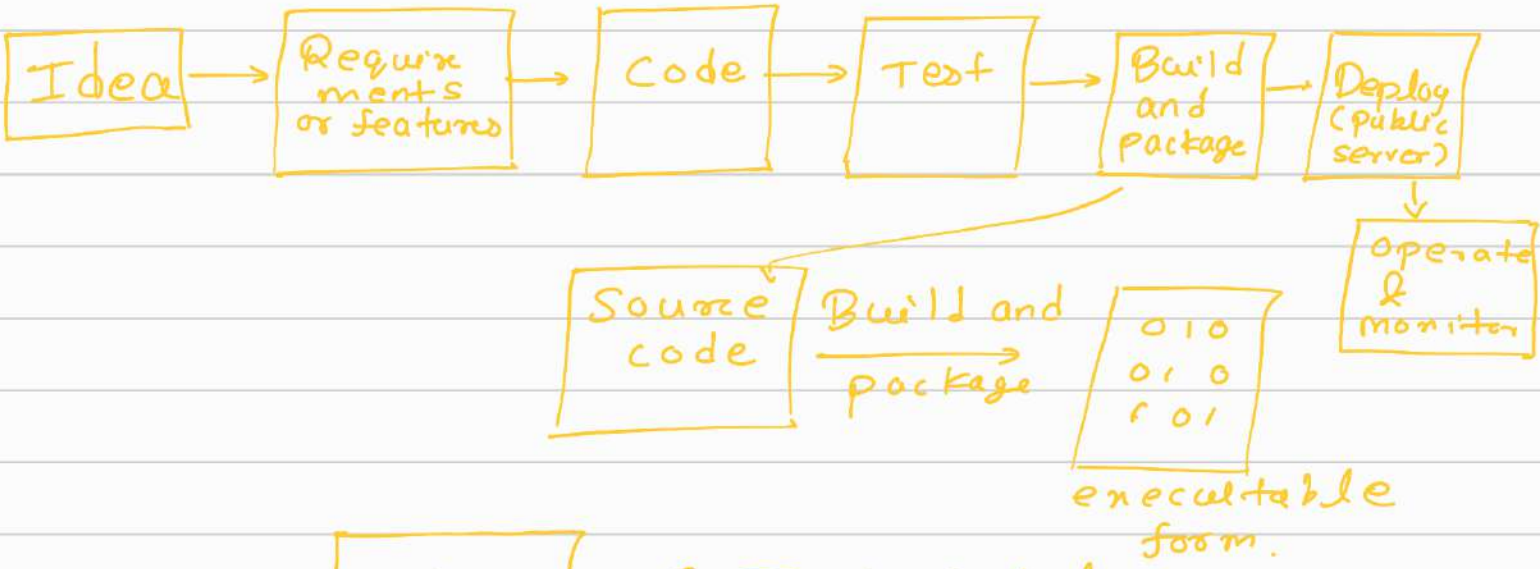
application release process

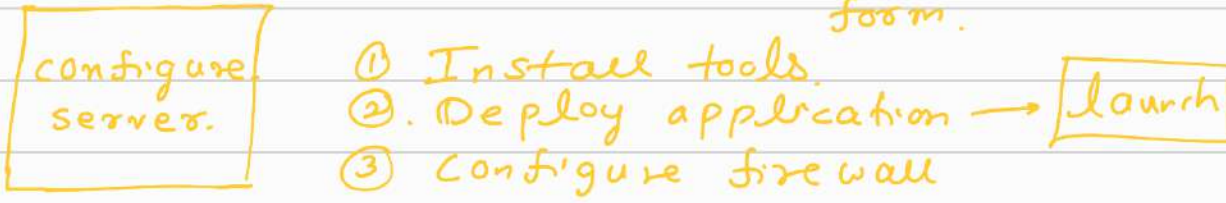
↓

Development + operations.

main goal: 'Deliver application to the end user'

"Typical software release process"



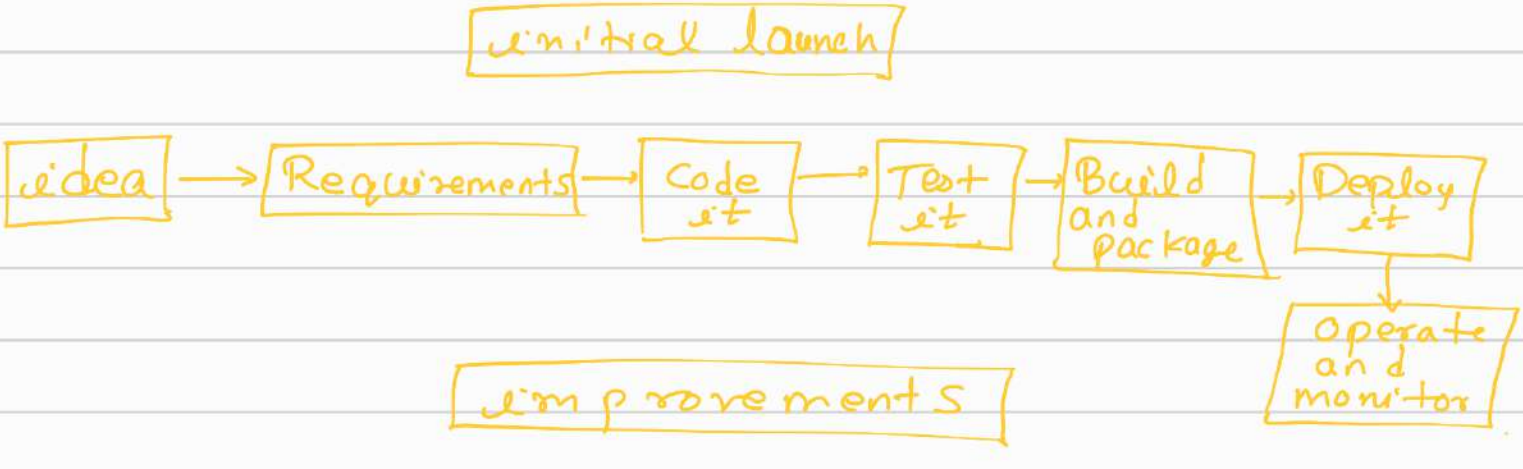


after Deployment you have to operate and monitor.

① any problem with application?

② are user experiencing the issue? any bugs you didn't catch during testing.

③ Can the application handle high user load?



① add new features.
② optimize performance.
③ fix bugs
} make it accessible to users immediately.

after initial launch - multiple updates

initial launch
code → Test → Build & package

update 1
code → Test → Build and package

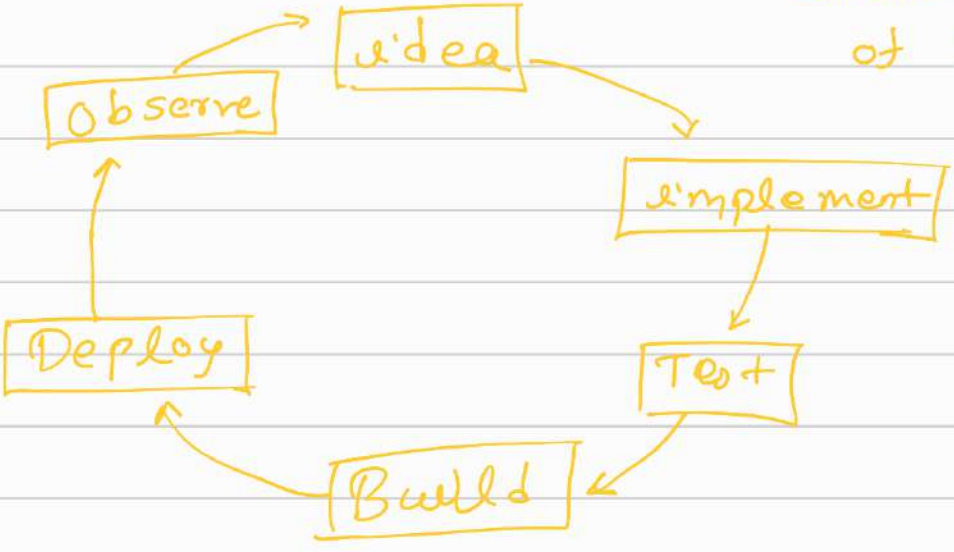
update 2
keep track → version those changes

update 3.
1.0.0
1.1.0
1.1.1
1.1.2.

1. 4. 2
major minor patch

Continuous Delivery

endless cycle of improvement.



Devops is the process of making continuous Delivery.

① fast ✓
② with minimal errors ✓

Challenges
① Quick delivery
② High quality code.

Challenges devops tries to solve.

Frictions and Roadblocks in the release process.

① miscommunication and lack of collaboration bet (Dev and ops guys)

Developers	operations.
→ 'Code the application'	→ 'Deploy and operate the application'.
→ can code it but can't deploy and run the app.	→ can run the application but don't know how the app works.

This creates miscommunication between two parties

eg- Developer finishes coding but the deployment guide for the operations team is not good (not well documented) so release takes longer.

eg Developer finishes coding but the feature cannot be deployed because it has lot of issues so the operations through it back with improvement suggestions.

→ Because of these things release takes time So when the developer is done with the feature and operation starts deploying it, there is no cleanly automated process of handover.

It is based on X complex Bureaucratic process

× checklist and Documentations.
× who needs to manually approve what for the release and so on

× — no streamline and automated process here.

② 'Conflict of Interest'

Both have diff incentives.

'Responsible for Development'

Dev

new features fast

Responsible for operations

Operations

Those changes don't break anything 'maintain stability' in production

app available ✓
No errors ✓

① Resist the speed of release
② Check that it's 100% safe.

slows down the release process

eg:- Developer Develop new features which was released (version 2.5). But this feature consumed so much resources in production environment, that the server got overloaded, and the application crashed.

> Dev not be as careful

> Ops need to fix it.

Common goal

Deliver high-quality apps to end user fast.

vs

job incentive

Dev → I need to quickly implement new features

Ops → I need to maintain stability.

3.) Security.

Security team - will check wheather new feature will affect system's security.

× same bureaucratic process.

→ Devops include security roadblocks as well

Dev Sec Ops

highlight the imp of integrating security.

4.) Application Testing

Testers — Testing the apps on diff levels.

① test specific features
② end to end Test
③ Testing on diff environments.
④ Performance Tests.

Can't rely on automated Test fully.

often manually

Testing - are there any bugs in applications ? → slow down the process

Tester

5.) lot of manual work.

a.) operations

→ Directly executing commands on server.

- install tools.
- configure stuff and do patches.
- execute scripts manually.
- manually deploying apps.
- manually configuring Jenkins build jobs.
- manually configuring user access and permissions.

× slow and more error prone.

× Knowledge sharing is difficult.

× intransparent, hard to trace, who executed when.

Devops try to remove those roadblocks which slows down the release process.

By creating

→ fully automated

→ streamlined process for release cycle.

→ By step by step removing 1 roadblock after another.

How does Devops help achieve this?

Combination of

Cultural philosophies **Practices** **Tools**

① Anything that creates the process of releasing software fast and with high quality.

② main part was that devs and ops should work together more often.

Devops as a seperate role

concepts of software development.

Software Developer

Develop an application ⟶ code repository. (Git)

Devops

① Understand the concepts of how developer work.

② Which git workflow they are using

③ How the application is configured to talk to other services, databases

④ concepts of automation Testing.

Dev ⟶ deployed ⟶ Ops

application ⟶ Server

so the user can access it.

infraststure

on premise server ⟶ cloud server.

They need to created and configured to run the application.

As a devops engineer,

① Responsible for preparing infrastructure.

② most of the server are linux server.

→① linux basics ✓
comfortable using CLI ✓
Shell commands ✓
linux file system ✓
Server management ✓

→② Basics of networking and security.
① firewall, proxy server ✓
② Load Balancer ✓
③ HTTP / HTTPS
④ IP, Dns name resolution

→* Devops vs it operations.
you don't have to super OS, networking, and security skills, and administer the server from start to finish.
There are own professions.
network and system administrator.
security engineers
that specialize in on of those areas

Your job as a Devops engineer is to understand the concept to the extent that you are able to prepare the server to run your application but not to take managing server and whole infrastructure.

Containers
You need to run the application as container on the server

virtualization ✓

Containers ✓

manage containerized application on a server ✓

Container technalogy → Docker.

How to get those features from Dev team to server to make it available to the end user.

How to release application?

Now devops comes in.

continuously (CI/CD) ✓

automated ✓

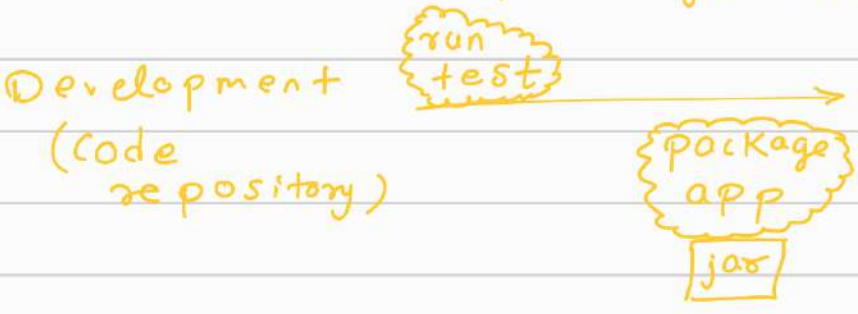
Build automation and CI/CD.

① new features done and run the test and package the application as artifact like .jar file or zip.

In this Build tool and package manager helps ✓

eg - maven, gradle - java

npm - javascript.



② Build Docker image from the applicaton This this image should be store in image repository (Artifact repository)

Docker artifact repository, nexus, Docker hub.

③. You don't want to do any of these manually. you want one pipeline to do all these in sequential manner

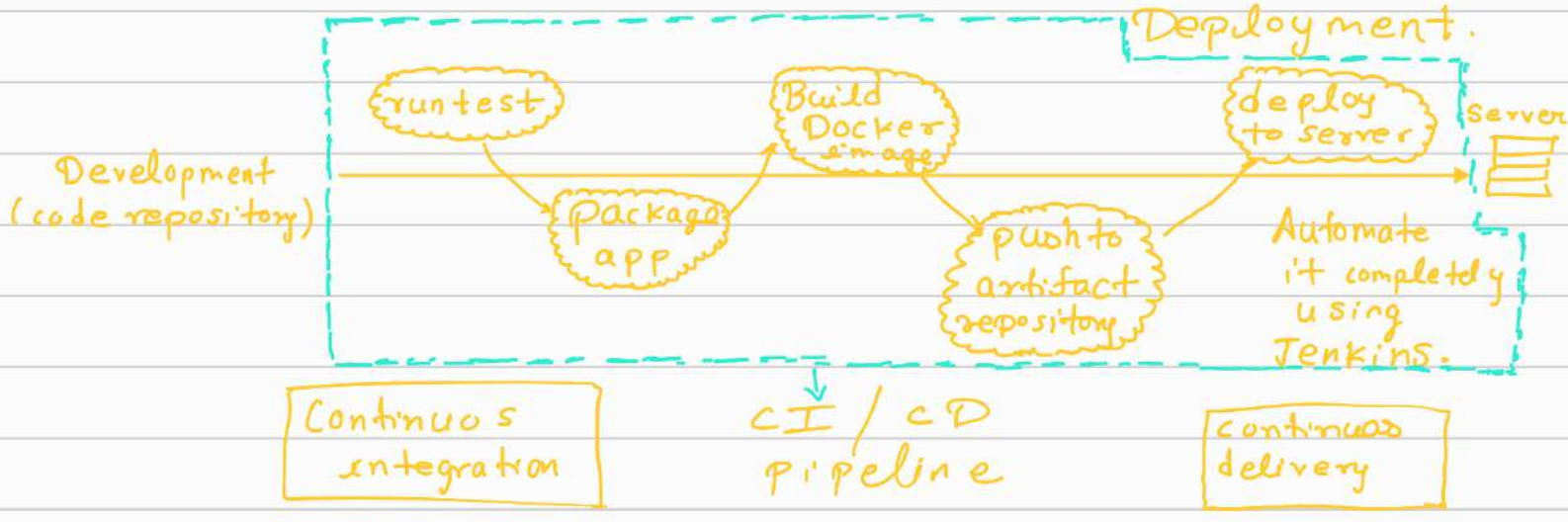
Build automation → Jenkins.

and connect the pipeline to git repository to get code.

+ This is continuos integration.

and you want to deploy that new feature and bug fixes. to the server.

→ This is continuos Deployment.



Some additional steps in pipeline.

① like sending notification to the team about pipeline stage

②. Handling fail deployment etc.

As a Devops,

configure complete CI/CD pipeline. ✓

Cloud providers

many companies using virtual infrastructure on the cloud.

Instead of managing own physical infrastructure

(Infrastructure as a Service)

aws google cloud microsoft azure

① saves cost of setting up own infrastructure.

② offers a range of services.

load Balancing backup.

clustering security.

many of the features and services are platform specific.

you need to learn them.

eg - your application runs on aws

you need to learn aws and its services.

But aws has lot of services, you need to know only those services that you need to deploy and run your application on infrastructure.

Container Orchestration

our application will run as container.

(docker images)

smaller application → (docker compose)

But if you have lot more containers like big microservice

container orchestration - Kubernetes. ✓

you need to administer and manage the cluster as well as deploy application.

Monitoring

To track the performance of application
if infrastructure has any problem.
in real time user experiencing any problem

monitor software ✓
monitor infrastructure ✓

monitoring tool - prometheus, nagios.

Infrastructure as a code

you need more than one environment
production testing development.
you need deployment environment multiple times.

we don't want to manually create infrastructure 3 times. because it is time consuming and error prone.

We want to automate
① creating the infrastructure
② Configuring it to run your application and deploy the app on that infrastructure

Infrastructure provising tool - Terraform.
configuration management - ansible chef puppet

environment becomes more transparent and state is easy to replicate and recover.

Scripting language

- automating task for dev and operations.

eg - backups, system monitoring, cron jobs.

OS specific Bash (mac OS) powershell (windows)

not OS specific - (OS- independent) python ★ golang ruby.

Version control

How to manage the code?

Git ✓

① Development concepts
② OS systems and Linux Basics.
③ networking and security.
④ CI / CD - Jenkins.
⑤ Cloud providers - aws.
⑥ scripting language. - python
⑦ Container - Docker
⑧ Container orchestration - Kubernetes.
⑨ monitoring - prometheus.
⑩ infrastructure as a code - Ansible, Terraform
⑪ Version control - git.

SRE - site reliability engineering.

Devops - what need to be done to achieve automate streamlined release process.

Sre - How to exactly implement this process and how to implement devops principles

SRE	Devops.
① Same Devops principle (release quality code fast)	① more focused on speed or delivery and release quality code fast.
② But more focus on reliability and keeping the system stable	

② What is SRE / Tasks and responsibilities / sre vs Devops

⇒ Why was there a need for SRE?

Devops made release process faster, but these releases were not as stable as ideally wished by devops principles and in Devops team there was no dedicted person that actually focused fulltime to keep system reliable.

S R E emerged

⇒ SRE Definition

SRE conceptualized by google

By Ben traynor

- sre team are made up of software engineer.
- who build and implement software.
- to improve the reliability of the system

⇒ Understanding System Reliability.

What is system?

System is server, infrastructure and platform (whale deployment environment where the application runs)

What does reliability mean?

unreliable services → email provider down once a week.
✗

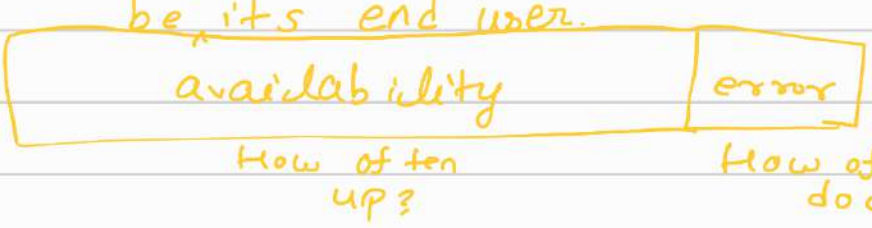
→ online Banking website down regularly.

reliable services - Gmail, youtube, twitter
✓ (rarely inaccessible)

→ How to make system reliable?
What makes system unreliable?
a.) Changes in infrastructure, platform (K8s), services and application

Sre → tries to automate the process of evaluating the affects the changes will have.

→ Sla and error budget.
Service level aggrement.
- commitment between service provider and customer.
- how reliable the system is going to be to its end user.



- expressed as percentage?
100 % Availability - 100 % Sla
↓
very difficult.

- Sla is usually defined by no of nines.
99.9 %
99.99 % 99.999 %

example of SLA.

→ SLA for accessibility

multiple aggrement like
- SLA for Response time
- SLA for error rate.

app → 1,000,000 request/week
99% sla
9,90,000 request successful

* Who defines SLA?

① How many request should be successful?

② How much downtime is acceptable?

① Business people ⟷ ② engineers sre/Devops

SLA? for their app.

Depending on
① industry benchmarks
② competition
③ user feedback.

Define sla on a higher level

- Define on a technical level
- Integrate into Devops and sre processes.

error budget → allocated downtime of a service

Defined sla → 99%.

97% uptime.
↓
- sre must work on making the systems reliable.
- until system is recovered, fewer changes will allocated

99.9% uptime
- release more changes
-

Simple way to regulate the release speed

→ SRe task and responsibilities.

① Creates automated processes to calculate and evaluate whether the service is within sla

②. Configure monitoring and logging.
(Observability for system performance)

What to do when outage happens?
How to prepare?

③. monitoring and alerting
(early indication issues)
Issue alert ✓.
DO: Service A in cluster B is throwing 500 error.

4.) Develop custom services to do it.

5.) Do on - call support.
Benefits of sre - doing support.
① What issues to expect ✓
②. How does the support deals with issues?
③. What improvement to be made to make support process easy?
↳ ① alert message, logs have enough info. to identify issue

② Issues identified too late ?

The main goal of sre:

→ Make sure the scope of outage is small when it happens, outage doesn't last long and is fixed very quickly, and less people and services are affected by the outage

Challenge when detecting the issues:

when outage happened.

① you have multiple things to fix
- chain of issues. (multiple level)
× fixing is complicated
× Takes more time.

② lesson learned and avoid again (outage)
↘ Postmortem. (Post-incident reviews)
— 'after issue', after-outage' analysis.
what caused outage?
Who fixed what?
Who did what?
What systems were affected?
- Stay blameless
- Document everything.

* SRe VS Devops.

Devops - more focused on speed on Delivery.
'Release quality code fast'.
Sre - Release quality fast But

more focused on reliability and keep systems stable

③ Devops roadmap 2022

Development.
→ developing of application
→ testing of application

operations.
→ deployment of app
→ maintained on a server.

↓
Devops

* Concepts of software Development:

Development.
→ program app.
→ Tech stack
→ code repository (Git)

Devops
→ How developers work
→ which Git workflow.
→ How the application is configured.
→ Automated testing

* Operating system and linux Basics.

application need to be deployed on server.

Devops
→ prepare the infrastructure.
→ linux Basics, CLI, shell commands, linux file system, server management.

* Basics of networking and security.
① firewall, proxy server.
② Load Balancer.

③ HTTP / HTTPS.
④ IPS, Dns name resolution

* Devops vs IT operations.

Devops - you dont need advance OS and networking skills and be able to administer the server from start to finish (network and system administrators)

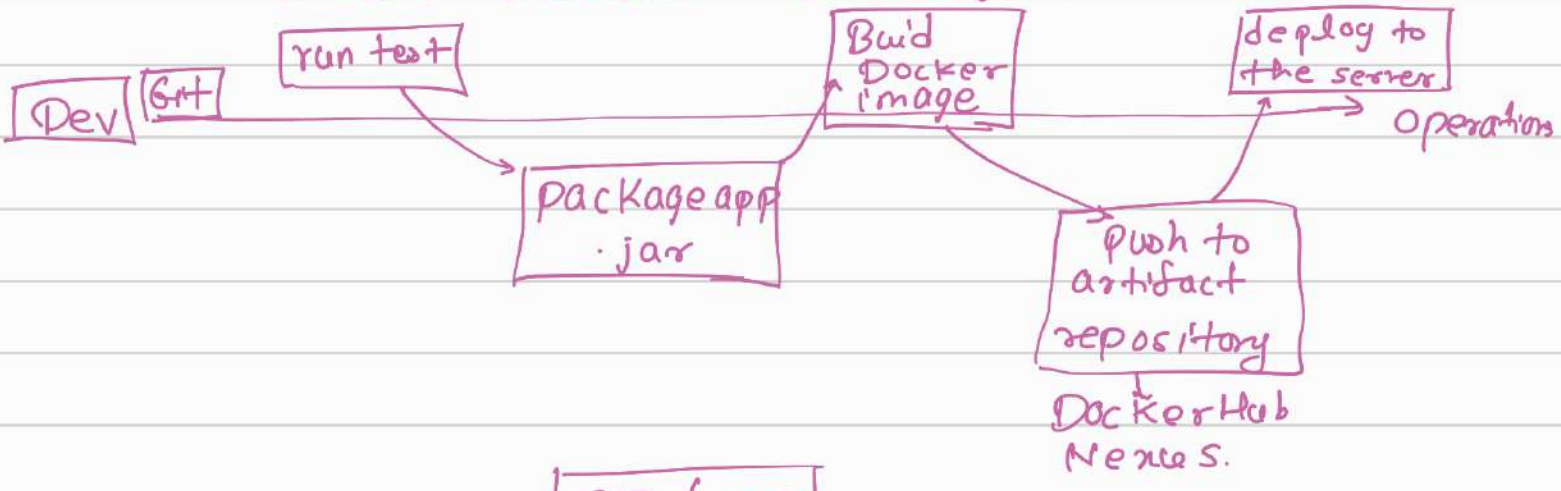
- Basics only - you should be able to prepare the server to run the application but not take over managing the whole infrastructure and server

* Container.
① virtualization
② containers

Docker.

How to release the application?
Devops.
CI / CD. ✓
automated ✓

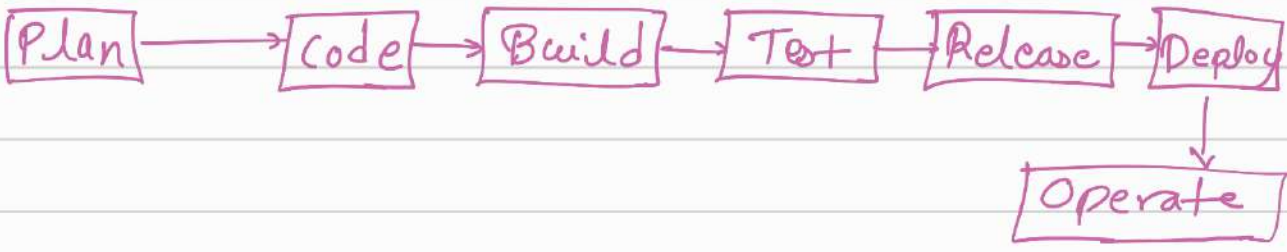
* Build automation and CI/CD.



CI / CD

CI / CD.

Devops → configure complete CI / CD pipeline



* cloud providers.

Virtual infrastructure on cloud. ✓
physical infrastructure (X)
(on premise)

Infrastructure as a service
platform (IaaS)

1.) aws
2) GCP
3.) Mic Azure 4.) linode.

→ save cost.
→ offer a range of services.
load Balancing, backup, clustering, security
platform - Based services
only learn services in a cloud which you actually need

* Container orchestration.

small app → docker compus, docker swarm.
lot of container (Big microservice) → Kubernetes

learn Kubernetes. ✓
administer and manage the cluster,

and deploy application in it

* monitoring.

Thousands of container running on kubernatives on hundreds of server

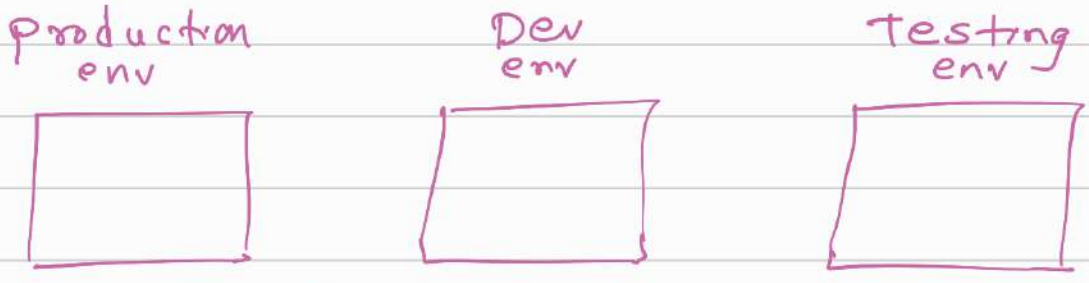
(Track performance)

real time user is experiencing a problem or not?

- ✓ monitor software.
- ✓ monitor kubernetes, server.

Tools - prometheus, nagios

* Infrastructure as a code.



To properly test your app before deploying it to the production

Same Deployment env multiple times.

Infrastructure provising → Terraform

configuration management → Ansible, chef, puppet

* Scripting language

- you need to automate the task for dev and operations.

eg- Backup, System monitoring, cron jobs.

OS specific - Bash (mac), powershell (windows)

OS independent - python, golang.

* Version control.

manage the code → Git
(Iaac)

④ What is Devsecops.

Security especially imp for

① Banking apps
② Social media apps
③ ecommerce (personal data)

Before Deploying to the production the security team must test the new versions for any vulnerabilities

- Outadated 3rd party libraries.
- Licensing issues
- Sensitive Data is leaked
- Vulnerable Docker Based Images.
- K8S misconfigurations

Security audits before release may block the whale process, delaying the release for weeks.

→ Why Does Security audit take so long?

more security risk nowadays.

How does Devsecops fix the problem?

Dev Sec Ops
Developer Security operations

Security should not be done only before release it should be part of Devops process.
- Create security policies
- Select automation tools for detecting security issues.
- Train Developers and operators.

In short - In DevSecops you use automation security tool in between CI/CD pipeline after every change so the feed back cycle is small and you dont need to test security of whole app right before release process which is very expensive.

④. Hybrid cloud vs multi cloud.

Hybrid cloud - when you use private cloud and one public cloud. together.

Befor cloud.
- companies would buy their own physical servers, routers, switches, cables etc. (on premise infrastructure) very costly.

Cloud
Somebody has bought all the servers, routers and set up physical infrastructure and they are allowing us to access it and use it. to run our applications
eg aws , GCP, micro azure.
for startups very beneficial
you can add or remove the server

with just a couple of clicks.
scale up or scale down the server depending on user traffic.
✓ Pay-as-you-go-model
Cloud providers have range of services.

Shouldn't all companies move to cloud?
Why still some companies still use on-premise infrastructure?

① Control over own infrastructure.
over cost, managed, secured.
Be not dependent on another company.

② Compliance and security.
Data privacy (medicle and payment regulation)
stricter regulation.

→ Hybrid cloud use cases
on-premise infra - online gift shop running
public cloud - when user spikes, spin couple of server to serve that temporary extra demand

'Cloud Bursting'

Data protection
on premise - database.
public cloud - your web application

Multi-cloud.
Use 2 or more public cloud.
1.) Replicate same workload
2.) Split workload
Cloud providers have multiple data centre throughout the world

reduce latency

— serve the user from data centre, which are closest to them

Higher availability

Many companies dont want to depend on one cloud provider

Why- bankrupt, outage

Use cases for splitting workload.

Choose best features of each cloud provider:

aws	GCP	azure
Best for web apps	Best for big data, analytics etc.	Best for microsoft products.

also when 2 companies are merged.

— multicloud

→ Challenges of Hybrid and multicloud

① multiple different environments to manage.

- each cloud provider has different API and services.

certifications for each cloud

on premise team	aws team	GCP team

lot of resources.

2.) Migration not easy

- Beacuse each environment is different.
- Re-architecting is needed

* How to address some of those challenges?

Kubernetes.

- abstact deployment and managing your application
- not dependent on underlying infrastructure.

Packer.

allows to build artifacts for multiple platforms from a single source configuration

Continuous Delivery platform for multi-cloud deployment

eg- spinnaker

⑥ Why learn python as a devops engineer

Why is python popular in general.

① easy to learn

② large ecosystem (many libraries)

③ flexible. (not limited to language specifics)

used for data science, web dev, ml, Devops, automation.

Why is python demanded in Devops?

flexible, simple, easy to read

① no compile-build cycle (like Java)

fast and painless

✓ great for simple scripts

② very lightweight

③ Simple python syntax.

✓ easy scripts to understand

✓ maintainable

④ platform independent.

⑤ many great libraries to automate devops task.

What is Python used for in Devops?

① many different tools you need to combine.

②. automatically update jira ticket info after jenkins build has run successfully

② automatically trigger jenkins job on specific events.

③ send notifications to team members on specific events

④ doing regular backups.
nexus server, app databases.

⑤ clean up old Docker images from the server to free server space.

⑥ automation scripts and programs for your teams
Dev, Testers, operations.

Minimum python knowledge as Devops engineer.

must
- automate manual task
eg monitoring, doing backups, cleanups.
- Basics of python.
- package management
- working with 3rd party libraries and api's

not a must
- know all the details of python
- implement complex apps
- create websites.
- advance oop programming
- web frameworks

How to get started with python?
course