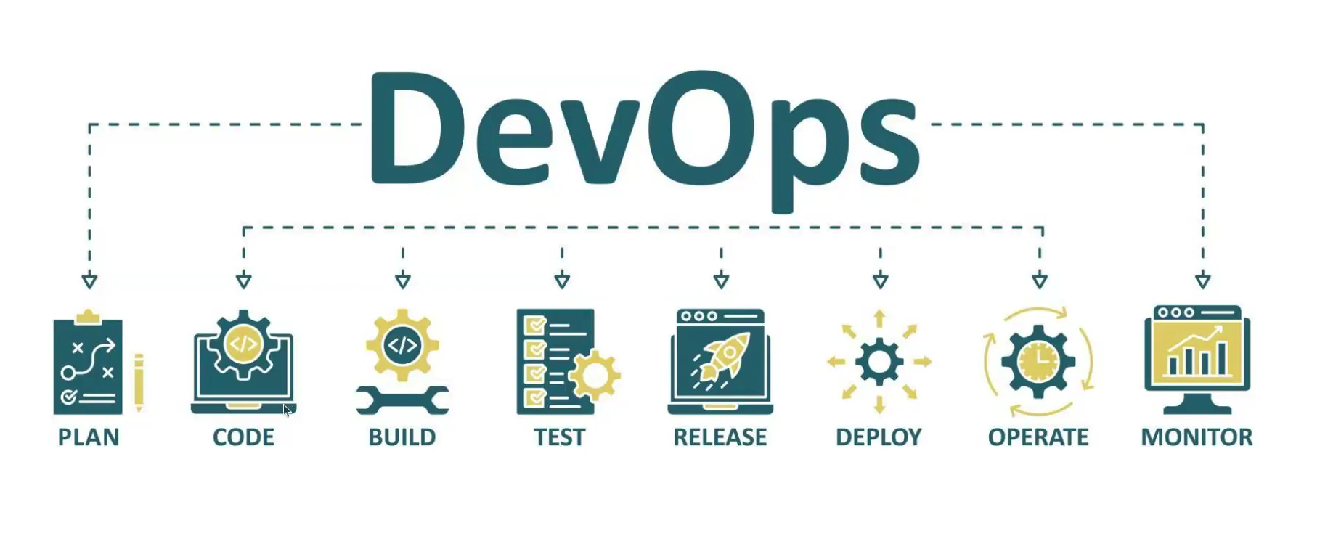
Conceptual Class: 1

Server povition: pre installment software for run a new project in server.



**Plan:** client requirement, server configuration, domain, network, public ip, db derver, frontend and backend, accessible.

CODE:

BUILS: environment consentancy, not depandancy,

TESTING: file structure , code integrity test, security test, vulnerable test, code structure, indentation check, unit test, api test, manual test,

RELEASE: collaborative criteria.

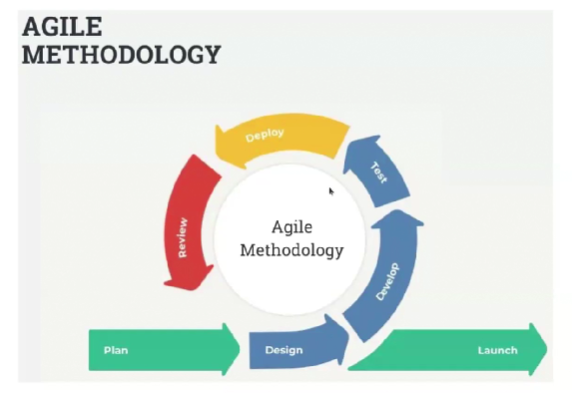
artified

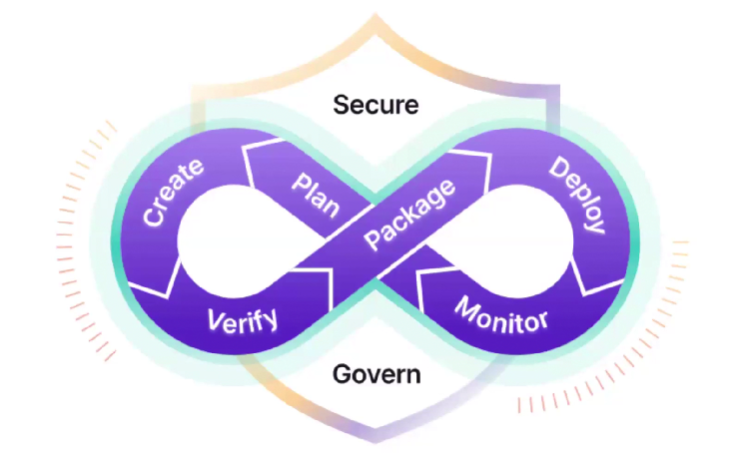
**DEPLOY:**

**OPERATION:**

**MONITOR:**







DevOps: deploy :-: continue :-: monitor :-: feedback

How many traffic in server, response time of API,

Monitor: some of project deploy in a server, how much resource need each project, resource means (cpu, ram, ssd, storage)

The main reason of server down is **(CPU & RAM exhaust)**

**If we notice that on project need so much space, cpu, ram, ssd then we can take a decision for this project.**

**Alert: for space,**

**Security, DevSecOps,**

**Some time: ip can access but don proper response, for this reason specific port must be open, some port must be secure by ensure port secure,**

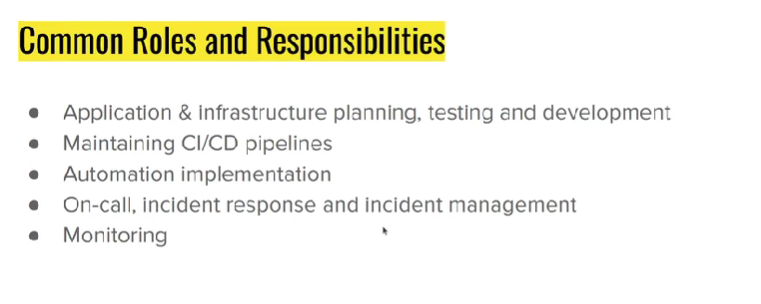
**Infrastecture design: which request by ip is eligible, which is not eligible.**

**Scal up: ram 80% uses, then we take a decision.**

**A single application have multiple server, when a user request and application get a request, then load balance term will come in front.**

**Multiple server, database structure, multiple db deta synk,**

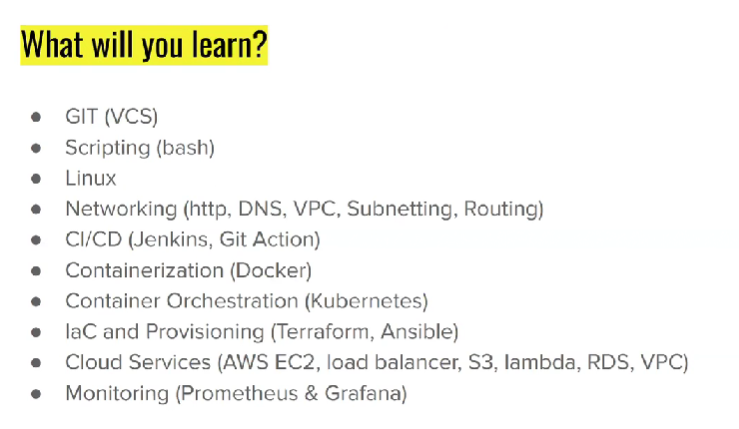
**Common roles and responsibility**



CICD Pipeline: Version control system, branching strauagy, code marge, cicd pipeline is important for avoiding manualy deploy avoid .

NB: server fail korle druto recover korar jonno automation

Monitoring Tool:



**Git Scripting (Bash)**: monitoring, linux commend in server environment.

# h top it’s a for how many resource(ram/ cpu) is used du, df, search file:

**Networking: https , http, dns,**

**Cicd (Jenkins, git action)**

**Containerization (docker)**

**Container orchestration (kubernetes)**

**laC and provisioning (terraform, ansible)**

**cloud service**

**montoring**

