2/08/24

DevOps

Overview of DevOps Architeture Design

Unit 1 : DevOps Workflow :-

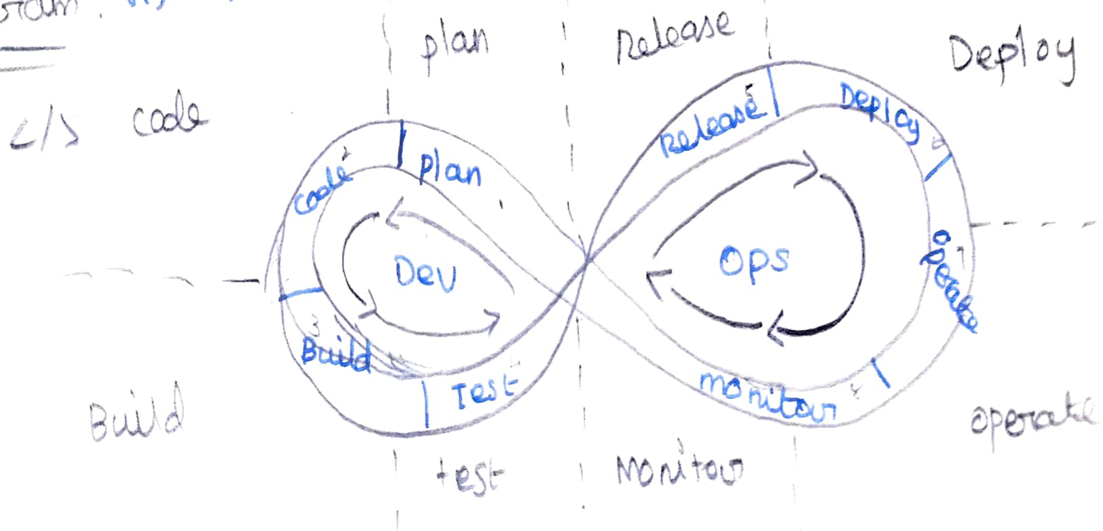
1.1. i) Definition and goals of DevOps
1.1. 2) DevOps Architeture
1.1. 3) DevOps Architeture workflow

Definition and goal of DevOps :-

the speed! The main goals of DevOps are to improve

i) increase Deployment frequency
ii) improve Deployment Quality
iii) Reduce Lead time for changes.
iv) Enhance collaboration and communication
v) Improve Recovery time
vi) Automate and Streamline processes

Diagram :



DevOps Architecture :-

Key components of DevOps Architecture :

⇒ Version control system (VCS) :
purpose : manages code versions, tracks changes and facilitates collaboration among developers.

⇒ Continuous Integration (CI) :
purpose : Automates the process of integrating code changes from multiple contributor into a single software project.

⇒ Continuous Delivery / continuous Deployment (CD) :
purpose : Automates the deployment of code changes to various enviornment, ensuring that software can be released reliably at any time.

⇒ Configuration Management :
purpose : Manages and maintains consistency in software environments (development, testing, production)

⇒ Infrastructure as code (IaC) :
purpose : Manages and provisions computing infrastruc -ure through machine readable definition files, rather than physical hardware or interactive configuration tools.

⇒ containerization and orchestration :
purpose : packages applications and their

dependencies into containers to ensure consistency across environments and simplifies deployment.

=) Continuous Monitoring and Logging:
purpose: Monitors applications and infrastructure to detect performance issues, error, and sesurity threats.

=) collaboration and communication tools:
purpose: Facilitates communication and ~~tool~~ collaboration among team members enabling faster decision-making and issues resolution.

Dev Ops Workflow:

Code: Developers write and commit code to a version control system (e.g. Git)
Build: The CI server automatically builds the code into executable files, creating artifacts that can be deployed.
Test: Automated tests are run to ensure the quality of the code. This includes unit tests, integration tests, and sometimes security checks.

Release : If all tests pass , the code is packaged and prepared four development

Deploy : The code is automatically deployed to the target environment (e.g. staging production)

Continuous Deployment involves deploying to production automatically , Whereas continous Delivery might requiered manual approval.

operate : The deployed applications are monitoured four performance, reliability and security , continuous monitouring tools collect metrics and logs , providing insights into the application behaviour .

Monitoor : Feed back is collected from monitouring and users providing data four continuous improvement , Any issues detected are fed back into the development process four resolution.

1.2 Devops vs Traditional IT operations :

1.2.1. Difference between DevOps and traditional software development and IT operations.

1.2.2 Benefits of adopting DevOps practices:

1.2.3 Building a culture of collaboration and communication between development and operations teams

1.2.4. The role of automation and monitoring in enhancing team efficiency.

1.2.1:

=> collaboration and communication

ⓐ Traditional approach : Development and IT operations teams work in silos, Developers focus on writing code, and operations teams are responsible for deploying and maintaing the application. This often leads to miscommunication delays and a lack of shared understanding.

ⓑ DevOps Approach : DevOps encourages continuous collaboration and communication between development and operations teams. Both teams work together throughout the software development lifecycle, fostering a culture of shared responsibility.

process and workflow :

- Traditional approach : uses a sequential development process (e.g., waterfal model) where each phase must be completed before the next begins. This can create bottlenecks and slow down the process.

- DevOps Approach :- Follows an agile and iterative approach where development, testing and deployment are done continuously and concurrently. This helps identify and fix issues earlier in the development process.

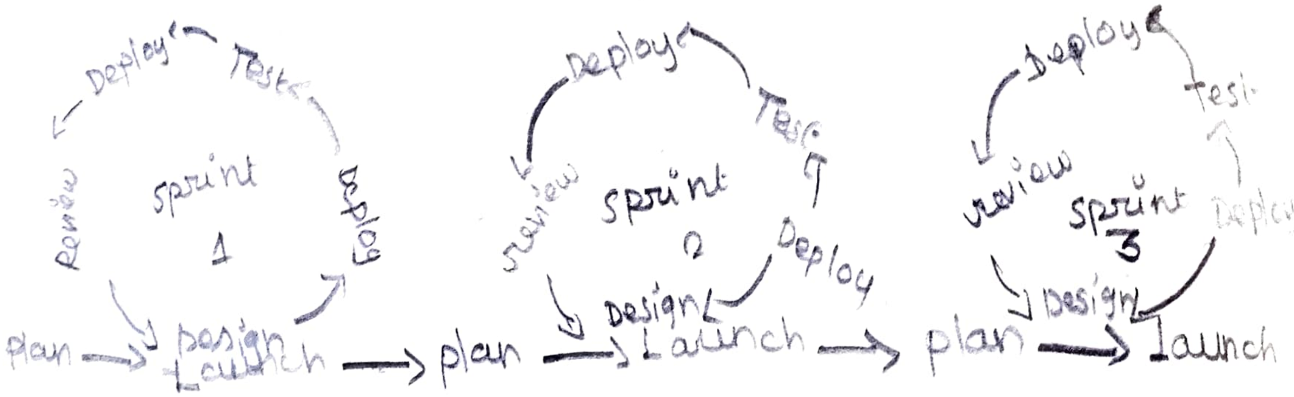
Water fall Model :-

It can make your project flow smoothly, avoid bottlenecks, help you hit deadline ensure deliveables are met before the next phase begins, and allow the team overall to shine with perfection. This in-depth guide analyses the advantages of the waterfall methodology.

i) Requirement gathering & analysis ii) System design, iii) implementation, iv) testing v) ~~Development~~ Deployment, vi) Maintenance.

Agile :-

Agile development is important because it helps to ensure that development teams complete projects on time and within budget. It also helps to improve communication between the development team and the product owner. Additionally Agile development methodology can help reduce the risks associated with complex projects.



Benefits