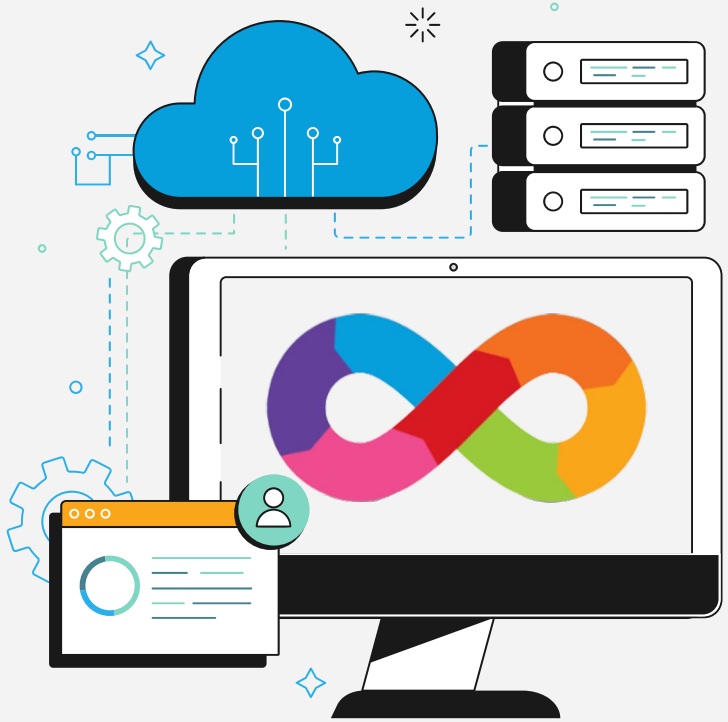
Introduction to DevOps

@ IBA - SMCS

Week 07
CI/CD Pipelines

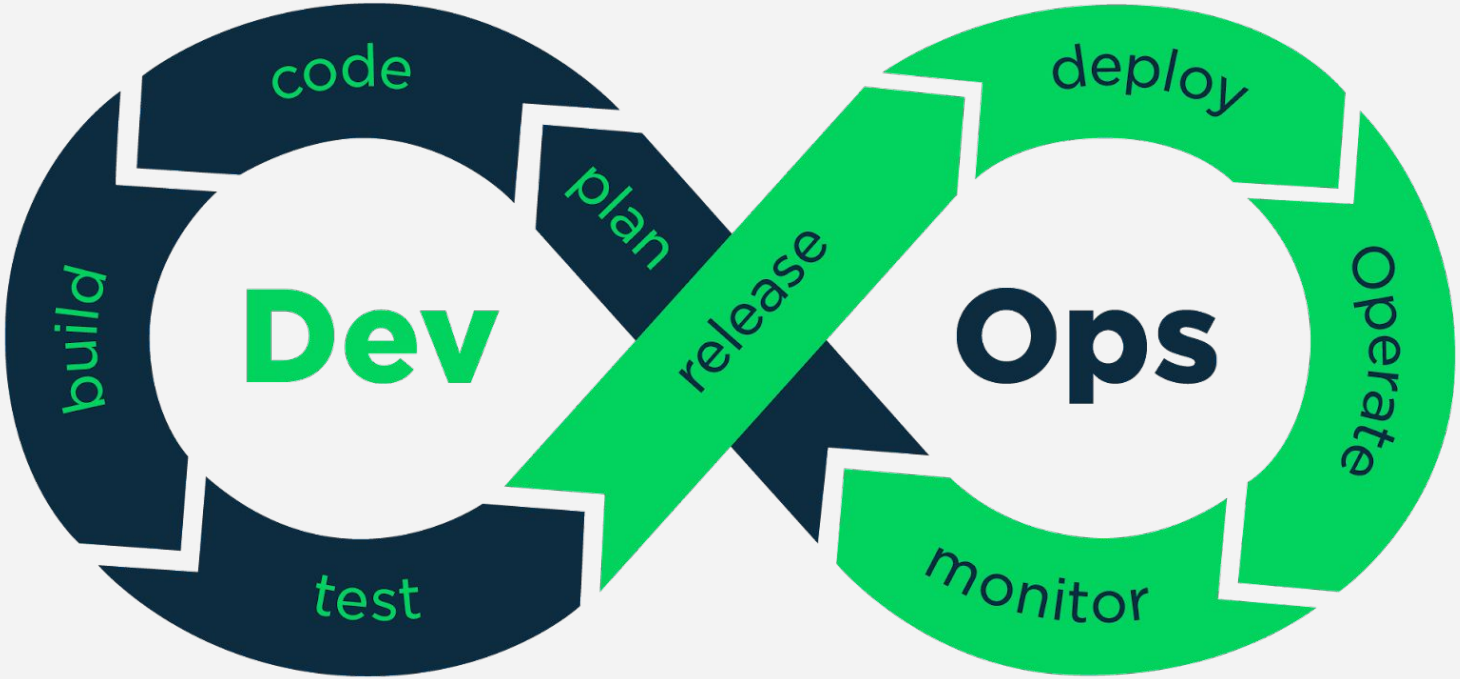


Obaid ur Rehman
Software Architect / Engineering Manager @ Folio3

Agenda

- CI/CD in the DevOps cycle
- What is CI/CD
- Delivery v/s Deployment
- Benefits of CI/CD
- Pipelines
- CI/CD Tools / Provider
- Deployment Strategies
- Hands on CI/CD using Github Actions

CI/CD in DevOps



Continuous Integration (CI)

- Continuous integration (CI) is a process that provides rapid feedback on the consistency and quality of code to all members of a team. It occurs when each user's code commit retrieves and merges the code from a remote repository, compiles it, and tests it.

Continuous delivery (CD)

Continuous delivery (CD) is the automation of the process that deploys an application in different stages (or environments).

Continuous deployment (CD)

Continuous deployment goes one step further than continuous delivery. With this practice, every change that passes all stages of your production pipeline is released to your customers.

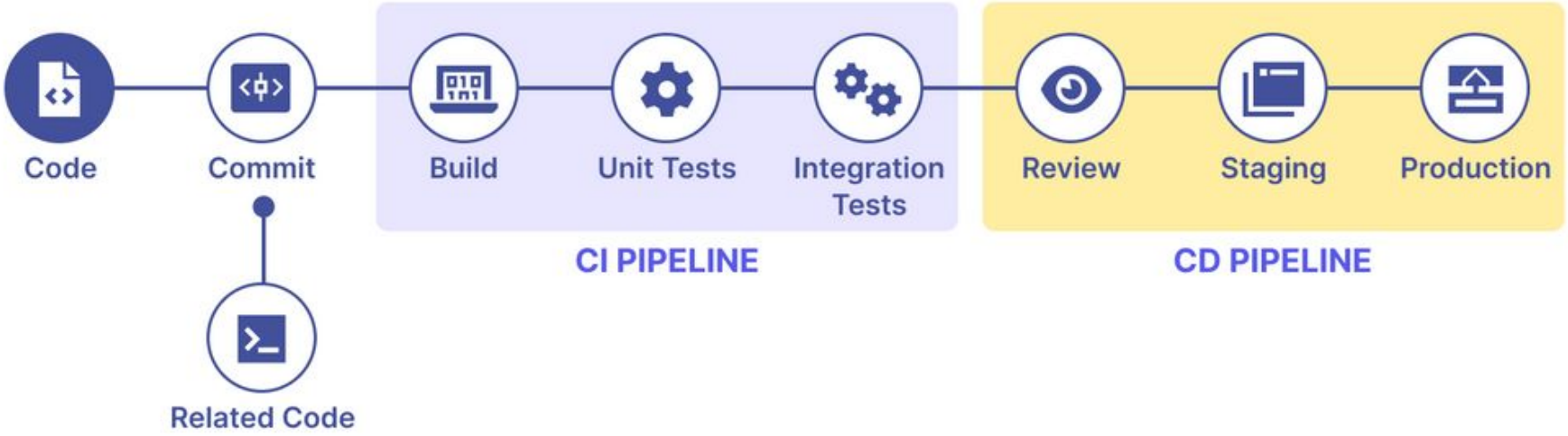
Continuous Delivery V/S Deployment

Pipelines

A route from development to production.

Within a CI/CD pipeline, a software release artifact moves and through various stage of the pipeline right from the code check-in stage through the builds → tests, deployment.

CI/CD Pipeline



CI/CD Pipeline tooling


GitHub


GitLab

TeamCity


Jenkins


Travis CI


CircleCI

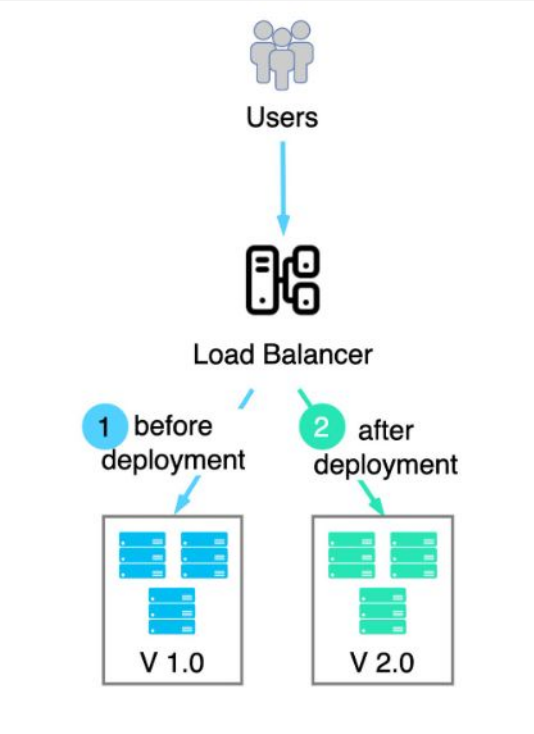

Codeship


Deploybot


JA619R
Hey, The Pipeline is Broken.
I will come and fix it
@paper_itta_techie
@paper_itta_techie
Are you a Plumber?
No. DevOps Engineer

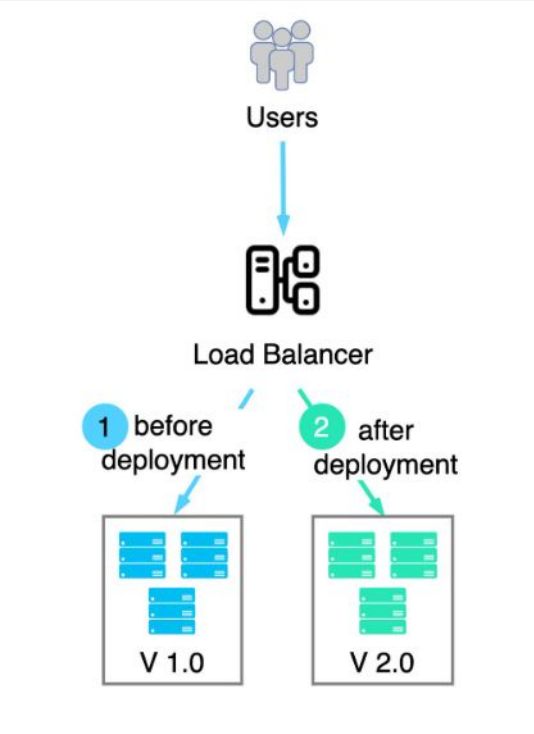
Deployment Strategies

Big-bang / One shot Deployment



- Deploy new version at once for all users.
- No Downtime.
- Faster deployment

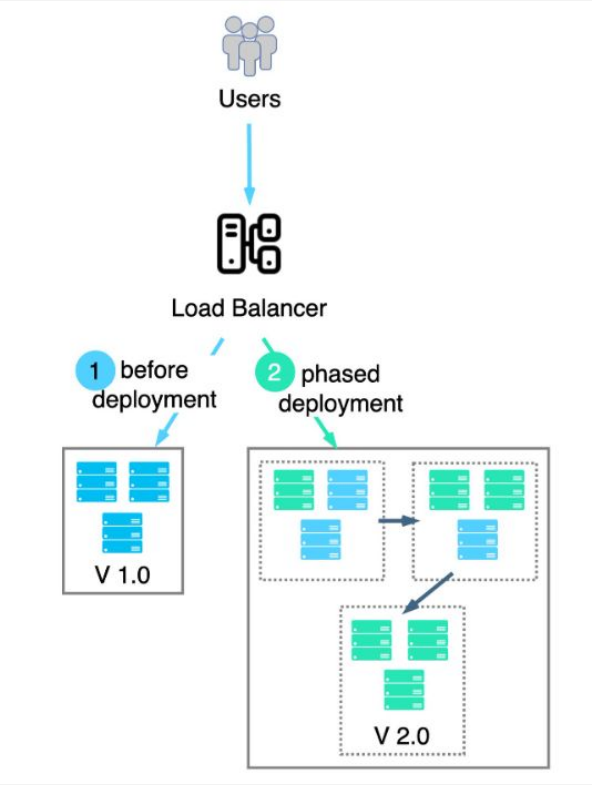
💣 Big-bang / One shot Deployment



✅Deploy new version at once for all users.
✅No Downtime.
✅Faster deployment

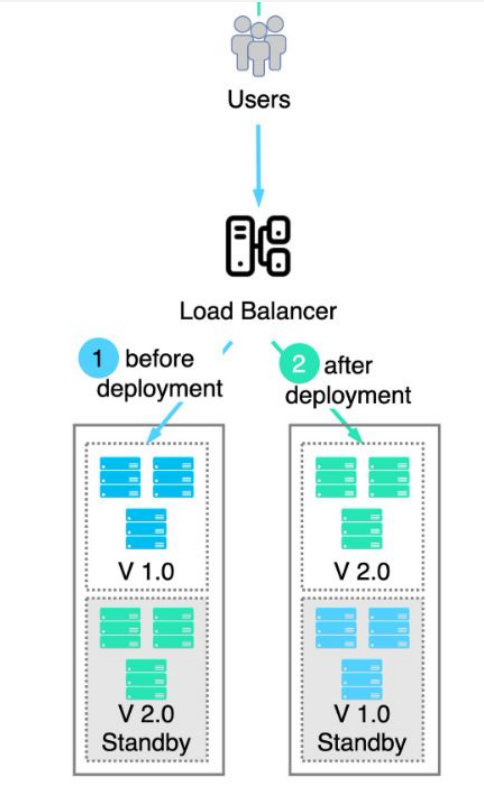
❌ If failure occurs users can be impacted while we rollback

Phased Deployment



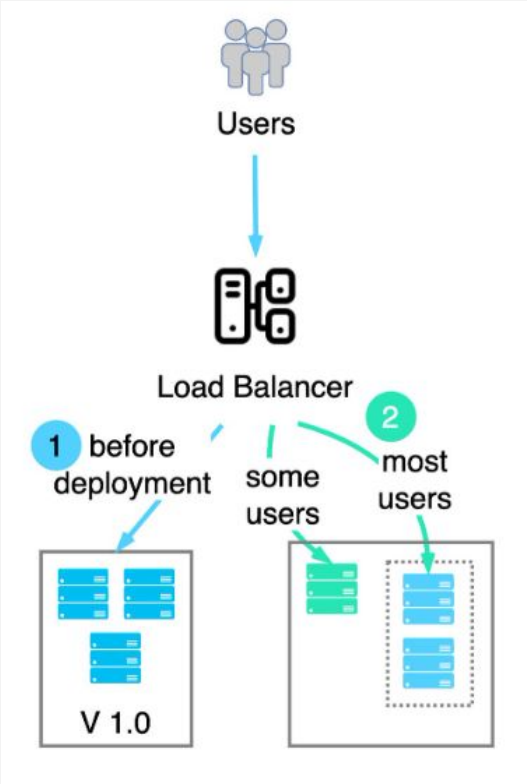
✅Deploy new version in phases.
✅New version is deployed while keeping the existing version running.
✅It enables a smoother transition, as updates are applied in stages, ensuring minimal disruption to the overall system.
✅If failure occurs we rollback easily.

Blue-Green Deployment



✅Blue Green Deployment is a release strategy where two identical environments, **blue** and **green** are maintained.
✅The blue environment represents the current production version, while the green environment is an exact replica of where the new version is deployed.
✅Traffic is initially directed to the blue environment.
✅ Once the green environment is tested and validated, the switch is made, redirecting traffic from blue to green.
✅ This approach allows for zero-downtime deployments, easy rollback, and quick recovery in case of issues, as the previous version is still available in the blue environment.
❌ Expensive, since 2 deployments are maintained.

Canary Deployment 🐦



✅ Canary Deployment is a release strategy where a new version of the software is deployed to a small subset of users or servers, known as the “canary group.”

✅ This group receives the updated version while the rest of the users continue with the existing version.
It allows for testing and monitoring the new version in a real-world environment, ensuring its stability and performance.

✅If the canary group experiences no issues, the new version is gradually rolled out to the remaining
Canary + Rolling Deployment ensures the best results

Reference

Alex Xu: https://www.youtube.com/watch?v=AWVTKBUnoIg

Hands-on

Deploy a NodeJs Todo App into GKE Cluster using Github Actions.

Link to the Demo shown in class:
https://github.com/ObaidUrRehman/iba-gke-cicd

End of Week 7

Q&A