Deploy a Static Website by Using Jenkins CI/CD Pipeline

JItendra Mannuru
R171130
RGUKT RK VAlley

Agenda:

1.What is Devops?

2.Evaluation And Objectives

3.Why Devops? And Technical Strategy Approach

4.What is CI/CD? And Importance

5.What is Continuous Integration and Flow Process

6.What is Continuous Delivery and Deployment And Flow Process

7.Project Explanation and Flow of execution

8.When to adopt and when not to

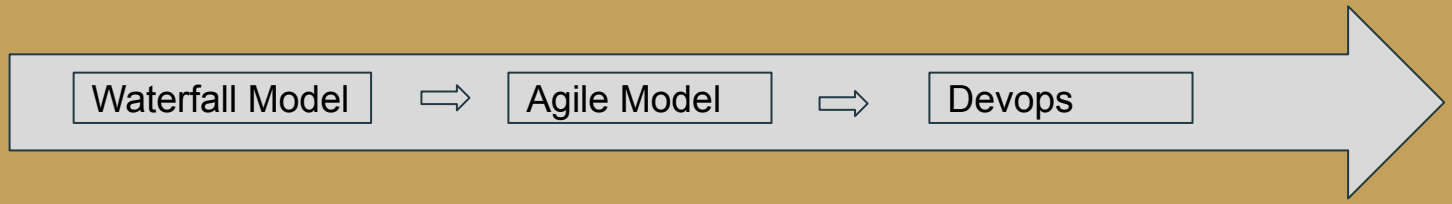
DevOps

What is it?

- A philosophy?Cultural change? Paradigm Shift?
- Alignment of development and IT Operations with better communication and collaboration?
- IMprovement in Software deployment?
- Breaking down the barriers between development and IT operations?
- Set of Tools and Processes
- ***It's All of the above!***
- *It's not a Software but it is a environment to develop applications.Devops combines the words Development and Operations.*
- ***The goal of the DevOps is to help a team work more efficiently by using the right tool and software development processes for the situation.***

Evaluation

Devops is built on agile principles and provide a platform for greater speed ,versatility



- It is a simple and classical Model
- Long pre-planned release cycles
- Business may wait long
- Time for delivery

- Short & frequent releases
- More responsive
- Business gets new features faste
- Testing, risk, impact and recovery time are low

- Continuous delivery to business
- Highly responsive
- Automation reduce risks and improves quality
- Improved productivity & efficiency

Objective

- Increase velocity of E2E delivery lifecycle
  - Deliver measurable business value quickly
  - Eliminate redundant manual effort
  - Automate everything!
- Increase Productivity
  - More Code
  - Faster Feedback
  - Component reuse
  - Eliminate Environment Configuration Issues
- Reduce Cost
  - Cost of poor quality
  - Cost of human error
  - Cost of manual processes
- Improve Quality
  - Software delivery and Release management

Why DevOps:

DevOps is important because it's a software development and operations approach that enables faster development of new products and easier maintenance of existing deployments..

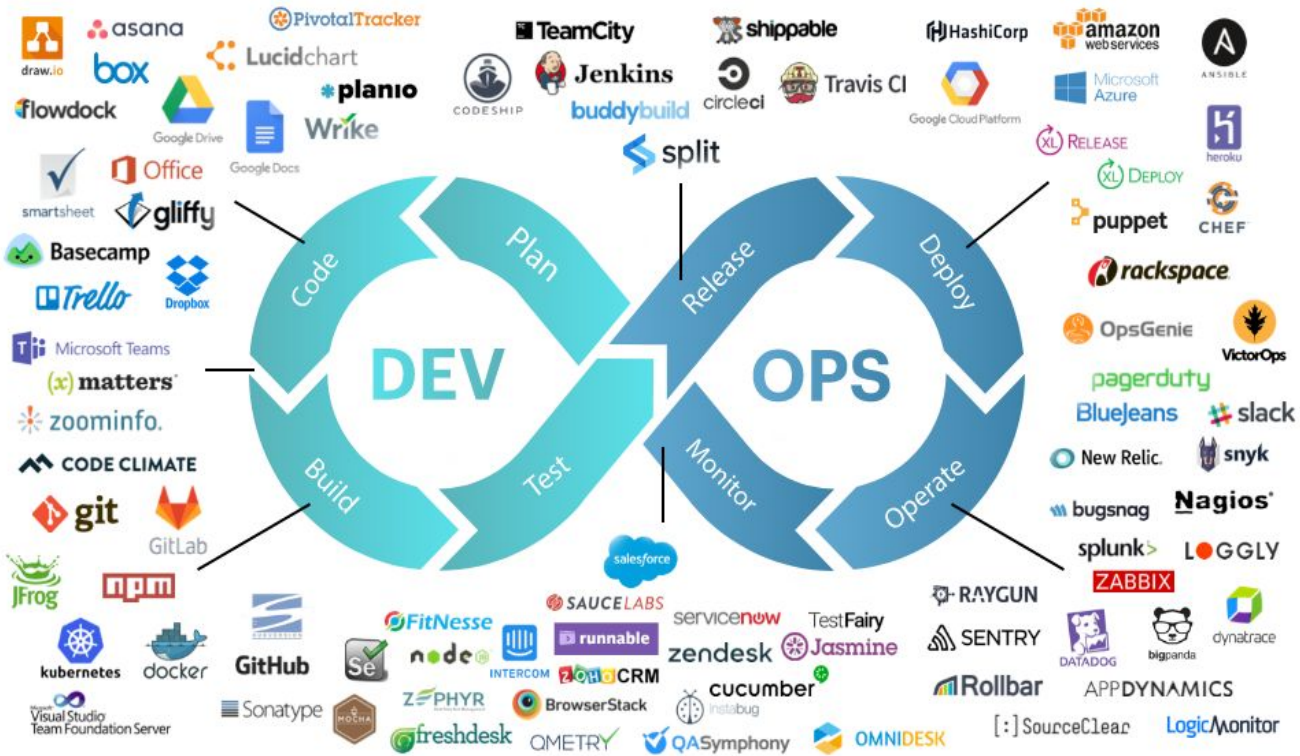
DevOps breaks down barriers between organisational functions which have including product, engineering, security, quality assurance and operations.

The most important DevOps benefits

wishdesk

Technical Strategy Approach

- Phase I
  - Build a CI/CD Pipeline to Automate Build and Deployment
- Phase II
  - Containerize Apps
  - Use App Containers to Build Environments
  - Deploy to Containers as though they were servers
- Phase III
  - Build Containers within the CI/CD Pipeline
  - Deploy Containers to Environments through and Delivery Platform


DEV
OPS
Code
Plan
Build
Test
Release
Deploy
Monitor
Operate
draw.io
asana
box
flowdock
Google Drive
PivotalTracker
Lucidchart
planio
Google Docs
Wrike
smartsheet
Office
gliffy
Basecamp
Trello
Dropbox
Microsoft Teams
matters
zoominfo.
CODE CLIMATE
git
GitLab
JFrog
npm
kubernetes
docker
Visual Studio Team Foundation Server
GitHub
Sonatype
CODESHIP
TeamCity
Jenkins
buddybuild
shippable
circleci
Travis CI
split
FitNesse
node
INTERCOM
runnable
Selenium
ZOHO CRM
ZEPHYR
BrowserStack
freshdesk
QMETRY
SAUCELABS
servicenow
TestFairy
zendesk
Jasmine
cucumber
Instabug
QASymphony
OMNIDESK
salesforce
HashiCorp
Google Cloud Platform
amazon web services
Microsoft Azure
ANSIBLE
RELEASE
heroku
DEPLOY
puppet
CHEF
rackspace
OpsGenie
VictorOps
pagerduty
BlueJeans
slack
New Relic.
snyk
bugsnag
Nagios
splunk>
LOGGLY
ZABBIX
RAYGUN
SENTRY
DATADOG
bigpanda
dynatrace
Rollbar
APPDYNAMICS
[:]SourceClear
LogicMonitor

CI/CD and Importance

- ➔ CI and CD stand for continuous integration and continuous delivery/continuous deployment.
- ➔ In software engineering, a **pipeline** consists of a chain of processing elements, arranged so that the output of each element is the input of the next
- ➔ In the software world, the CI/CD pipeline refers to the automation that enables incremental code changes from developers' desktops to be delivered quickly and reliably to production.

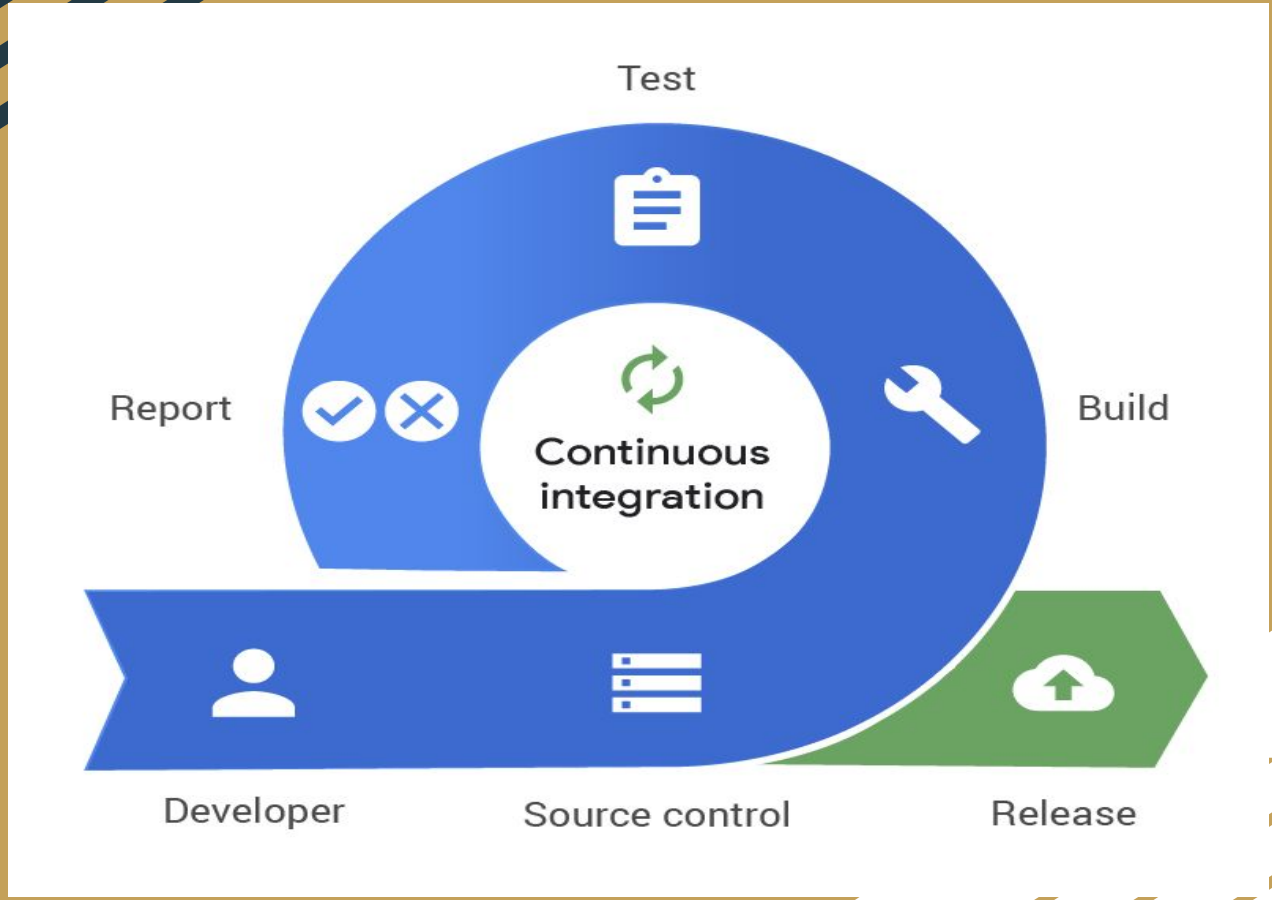
Importance:

- ➔ CI/CD allows organizations to ship software quickly and efficiently. CI/CD facilitates an effective process for getting products to market faster than ever before, continuously delivering code into production, and ensuring
- ➔ An ongoing flow of new features and bug fixes via the most efficient delivery method.

Continuous Integration

Continuous Integration is an automation strategy to resolve merge conflicts arising between individual developers' code and separate code branches. CI enables teams to build workflows capable of compiling frequent code changes, building applications, and testing the newer versions for bugs and errors. After passing through the test suite, the updates are ready for the next stage of the development process.

By frequently merging changes in a shared branch, commonly known as a trunk branch,CI offers an effective solution against merge conflicts. Also, it eliminates the need to wait until the end of the coding stage to combine separate branches, reducing the complexity of issues that may Occur and making them easier to resolve.


Test
Report
Continuous integration
Build
Developer
Source control
Release

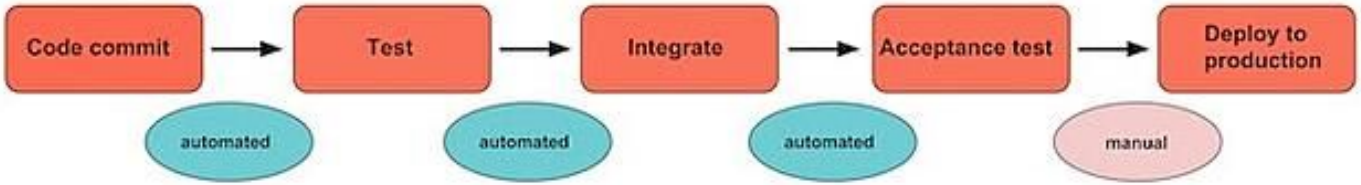
Continuous Delivery

After the code is bug tested, it is directly sent to a shared repository, for instance, GitHub. Continuous Delivery aims to boost communication between the development and operations teams by increasing the velocity at which production-ready code is delivered. It allows operations teams to deploy applications with fewer complications.

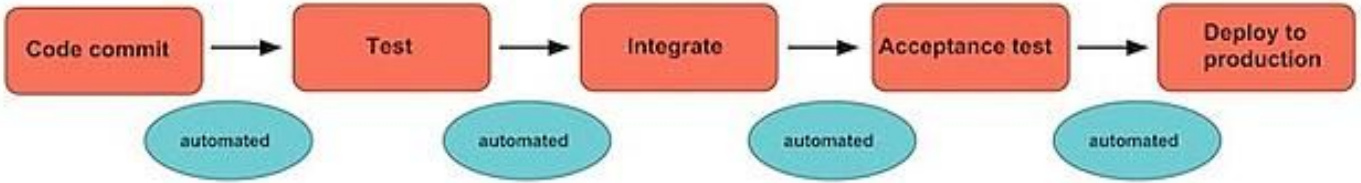
*Difference Between **Continuous Delivery** and **Deployment**:*

Continuous delivery is a partly manual process where developers can deploy any changes to customers by simply clicking a button, while **continuous deployment** emphasizes automating the entire the process.

Continuous delivery



Continuous deployment



Project Explanation

- This is a simple DevOps Project where we can deploy a static Website.
- By this project ,We can create a Continuous Integration and continuous Delivery
- For this I have use the technologies ,
  - Jenkins-It is famous CI/CD tool
  - Github- Source code repository
  - Apache Tomcat- Server to deploy
- Firstly,Explain about **Github**
  - It is Source code maintenance system
  - I took a basic project from github and forked into my account and saved it as in devops folder.
  - I copied the http URL of my project folder of githu account from code option section
  - This is about github.

- Next Coming to **Apache Tomcat** installation
  - I have downloaded the software from apache site
  - And installed the Tomcat server in system

- Coming to **Jenkins** Installation
  - The jenkins is the automation tool which consist of the Continuous Integration and Continuous Delivery Processes.
  - Installed in system and logged using http://localhost:8080 portal .
  - Connected to Jenkins Dashboard via login using initialAdminPassword and installed required plugins.
  - The Jenkins run on plugins when it can regulate and maintain these phases by using appropriate Plugins

Project Setup and Execution Process:

- Login into your github account and goto your project folder and open it and copy the path of the github http URL request on code option.
- Login into Jenkins Dashboard and create new Item ->project name->select free style project click ok and you will redirect to Project Management Configure Dashboard
- Set up the github url into Source code management and select build tool as invoke top level maven.
- In POst build action,select tomcat as server and create credentials and select those credentials click apply and save it
- Then you will redirect to project dashboard and click on build now option and It starts building your project.
- Here we are using automated deployment so after successful built it will redirect frploy the project into tomcat server
- Login http://localhost:8000 to view required project.

When to adopt::

1. Overcoming the dev versus ops mentality-The objectives of these two groups often counter each other, causing friction points and resulting in handovers and increased costs, along with longer feedback loops

2.Moving from legacy infrastructure & architecture to microservices-Older infrastructure and applications with complex architecture stacks can be problematic, even if they have served the company for years

3.Implementing a test automation strategy-Having a clear understanding of how to implement the test strategy can go a long way in getting test automation adopted across the wider organisation, shortening the feedback loops and getting your products out the door quicker!

4.Too much focus on tools-However, with the introduction of new tools comes the need to train your staff on how to use them, ensuring they meet security requirements and are well-integrated with the existing infrastructure.

5.Team ownership for deployments & releases-In organisations where DevOps practices are being implemented, we still see that teams do not have full ownership of their deployment and release cycles of their software.This is often due to lack of understanding of the difference between deploying and releasing

➔ **When not to Adopt:**

- Your business doesn't need regular releases
- Your business is satisfied with the current state of software.
- Your business has lots of M&A activity on the horizon-This may include when the business is in the middle of a lot of merger and acquisition (M&A) activity.

Why I Choose This Project

- I chose this project ,Because the CI/CD is the heart of the devops.Before going deep into this methodology/process causes some conflicts and Misunderstanding of the project and configurations and maintenance process.
- So that I learnt this CI/CD process and developed a mini project about how the CI/CD pipeline and Configurations.

Thank You..🙌