DevOps: Hello World!!

Speaker

Sanchit Jain

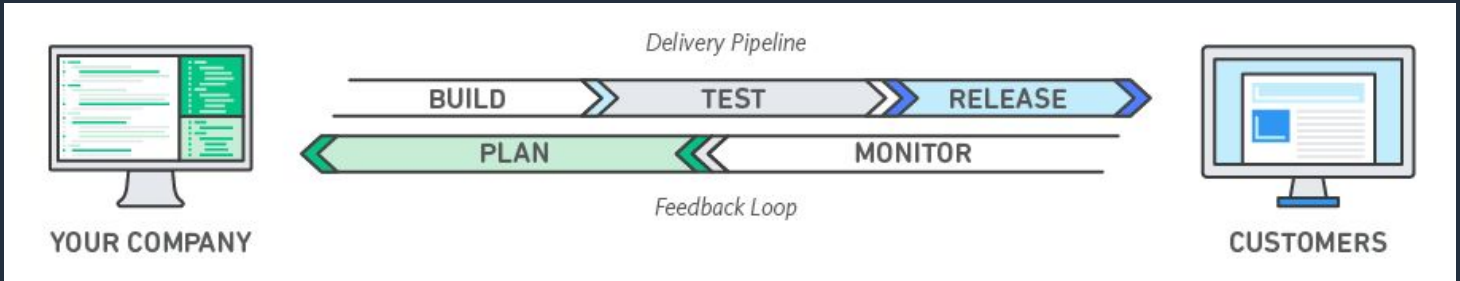
Lead Architect - AWS at Quantiphi

AWS APN Ambassador

FOLLOW ME

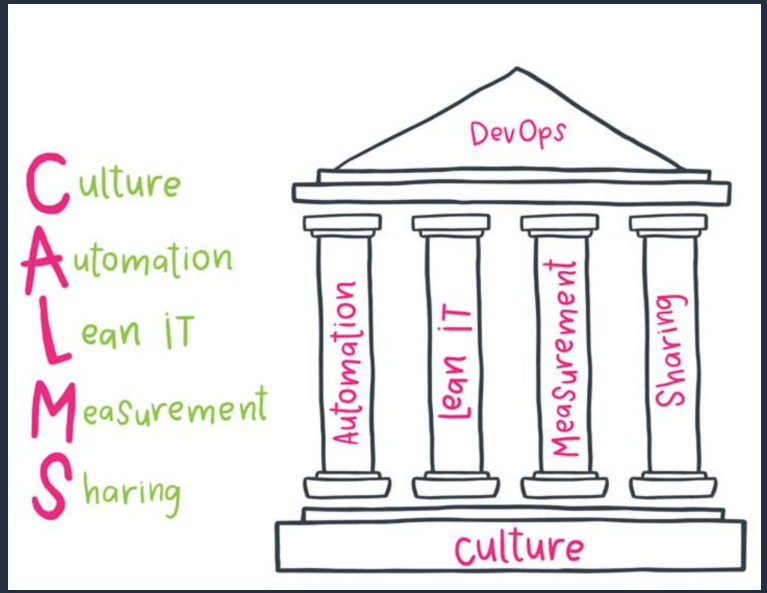
What is DevOps?

- DevOps is a combination of cultural philosophies, practices and tools that increase an organization’s ability to deliver applications and services at high velocity; evolving and improving products at a faster pace than organizations that use traditional software development and infrastructure management processes.
- It combines software development (Dev) and information-technology operations (Ops) that results in shortening the systems development life cycle and providing continuous delivery with high software quality and exceptional performance.
- In order for DevOps to impact businesses, it needs to be embraced and adopted by everyone responsible.



What is CALMS Framework?

CALMS is a framework that assesses a company's ability to adopt DevOps processes, as well as a way of measuring success during a DevOps transformation



Introduction to DevOps

The DevOps Process

DevOps Practices

Impact of DevOps

What Drives the Need for DevOps?

Lack of Automated and Secured Workflow Management Systems in Business organization

Building And Maintaining Servers

Time consuming and unproductive

No Environment Management

Differences in development and production environments

Slow Deployments

Costly error prone manual process and efforts

No Shared Ownership

Lack of feedback and proper metric leads

No Proper Configuration Management

Discrepancies in managing configurations

Deployments Are A Blocker

manual management of application configuration and versions

Production Downtime

Due to lack of improper deployment instructions / checklist

Hacking & Security Threat

Fixing directly in production, instead of a proper hotfix process

Impact on business due to challenges & Problems

Ineffective utilisation of resources

Misuse of working capital

Increased in time-to-market

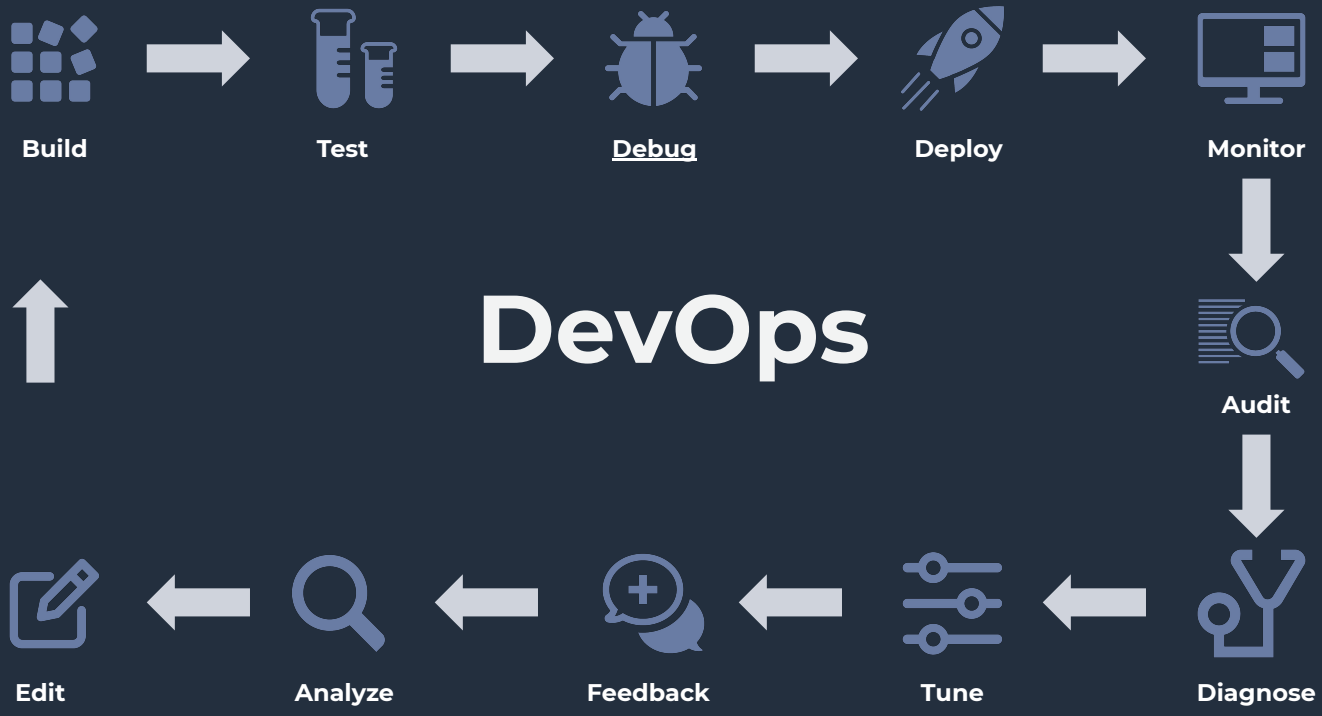
Laboursome process

Traditional Development Life Cycle

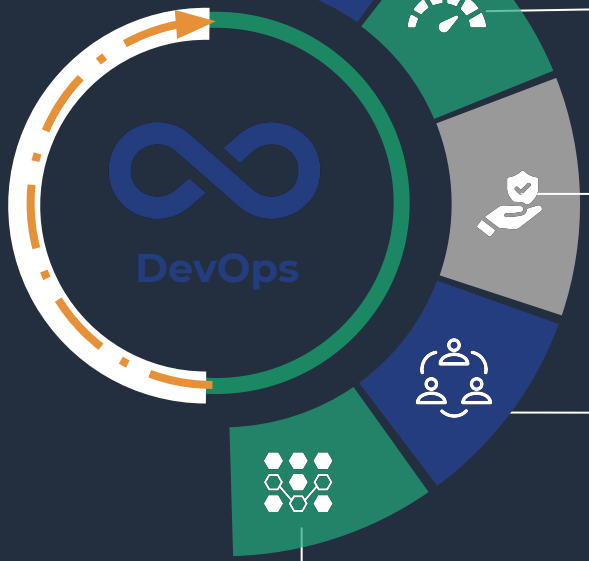
Challenges

- It does not allow for much reflection or revision.
- Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- More IT resources required and less collaboration.

DevOps Life Cycle



Business Impact by implementing DevOps



Improved Build Quality

- Easy knowledge and information sharing
- Regular specialized **code reviews**
- **Combination** of **dev-centric** and **ops-centric**

Accelerated Application Delivery through Agile

- **Improved synchronization** between dev & ops teams to avoid creating **complexities and misunderstandings**
- **Early detection** of errors leading to quick fixes

Improved Application Reliability

- **Automate** the **entire delivery pipeline**
- **Eliminates the time-consuming, cumbersome**, as well as error prone manual processes

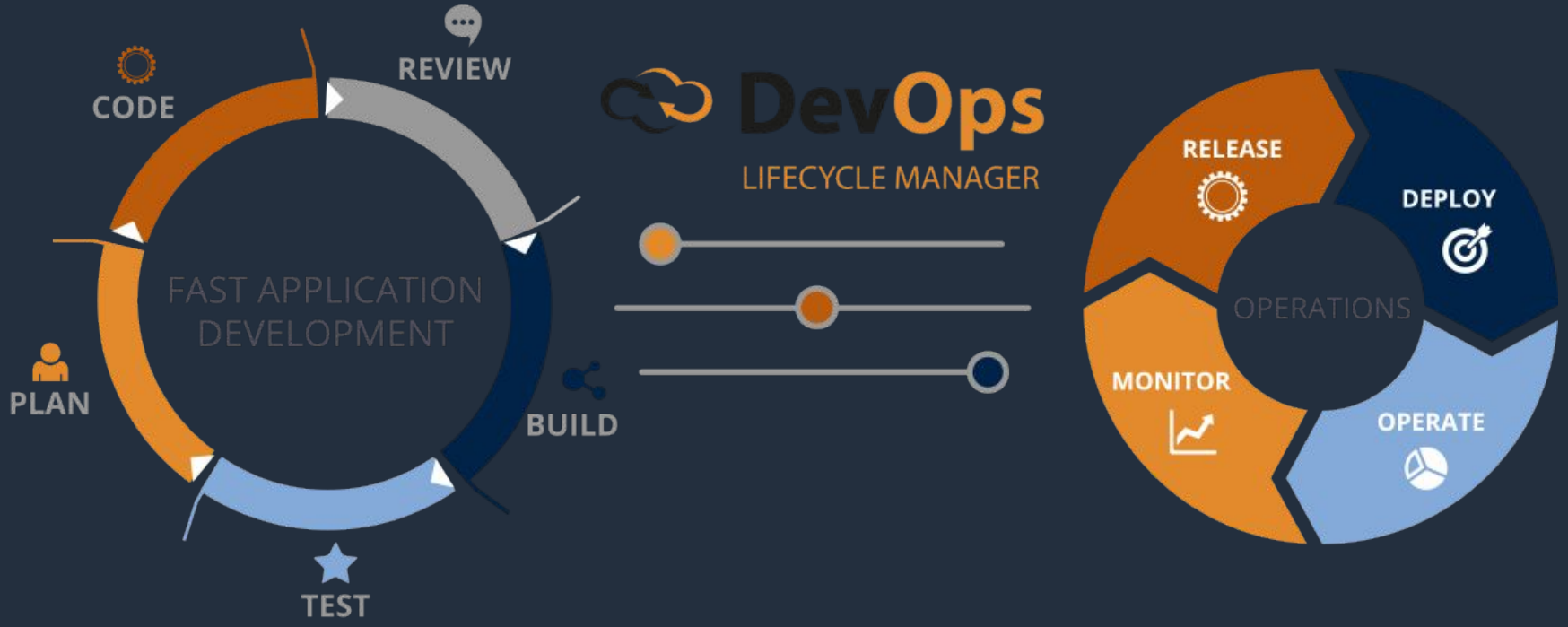
Improved Team Collaboration

- A culture of constant **learning and improvement**
- Working in **collaboration** of Dev and Ops team helps in **swift execution** of projects through an Agile process and dynamically reduces bottlenecks

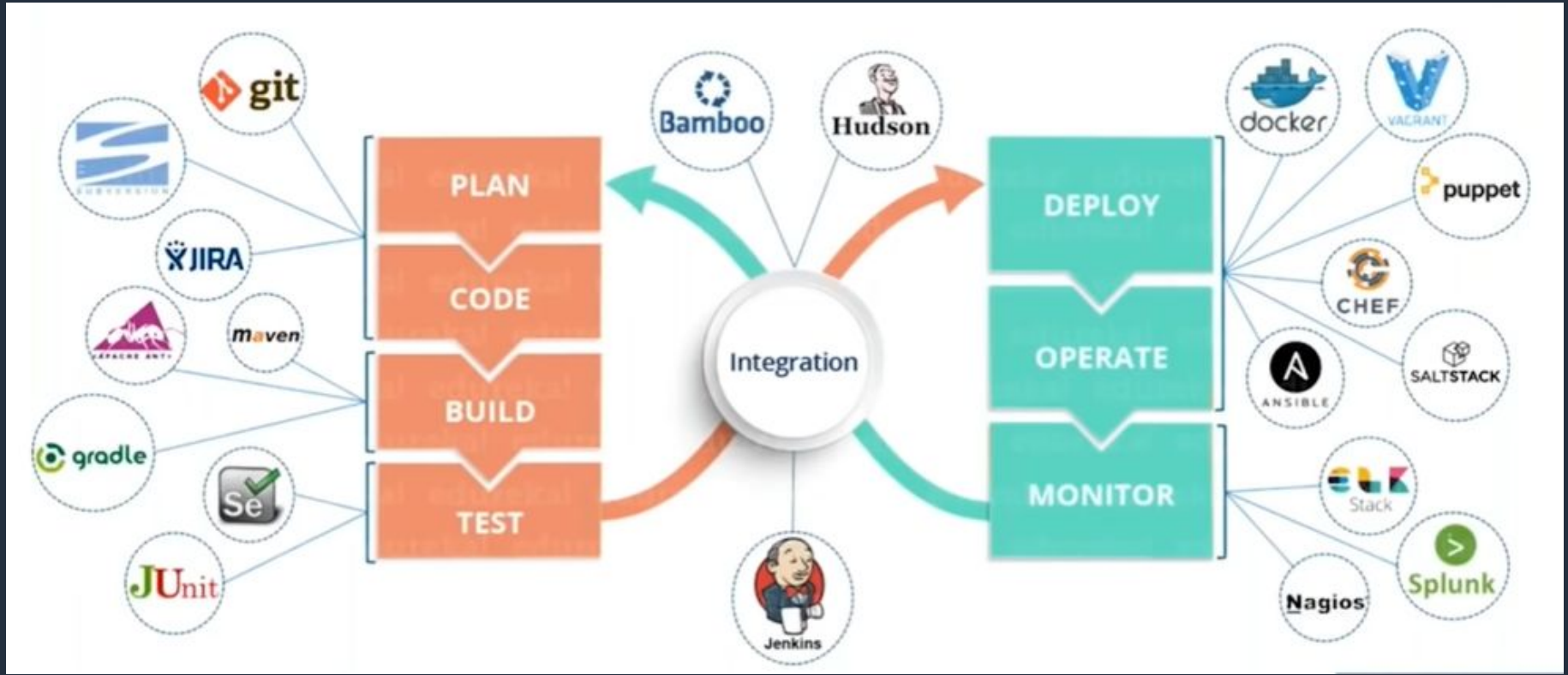
Minimized Impact of Complexity on Software Development

- Eliminates the risk by giving **more control** to the teams by managing smaller chunks of the software solution
- Encourages the use of **automated procedures** to build, package and deploy the application

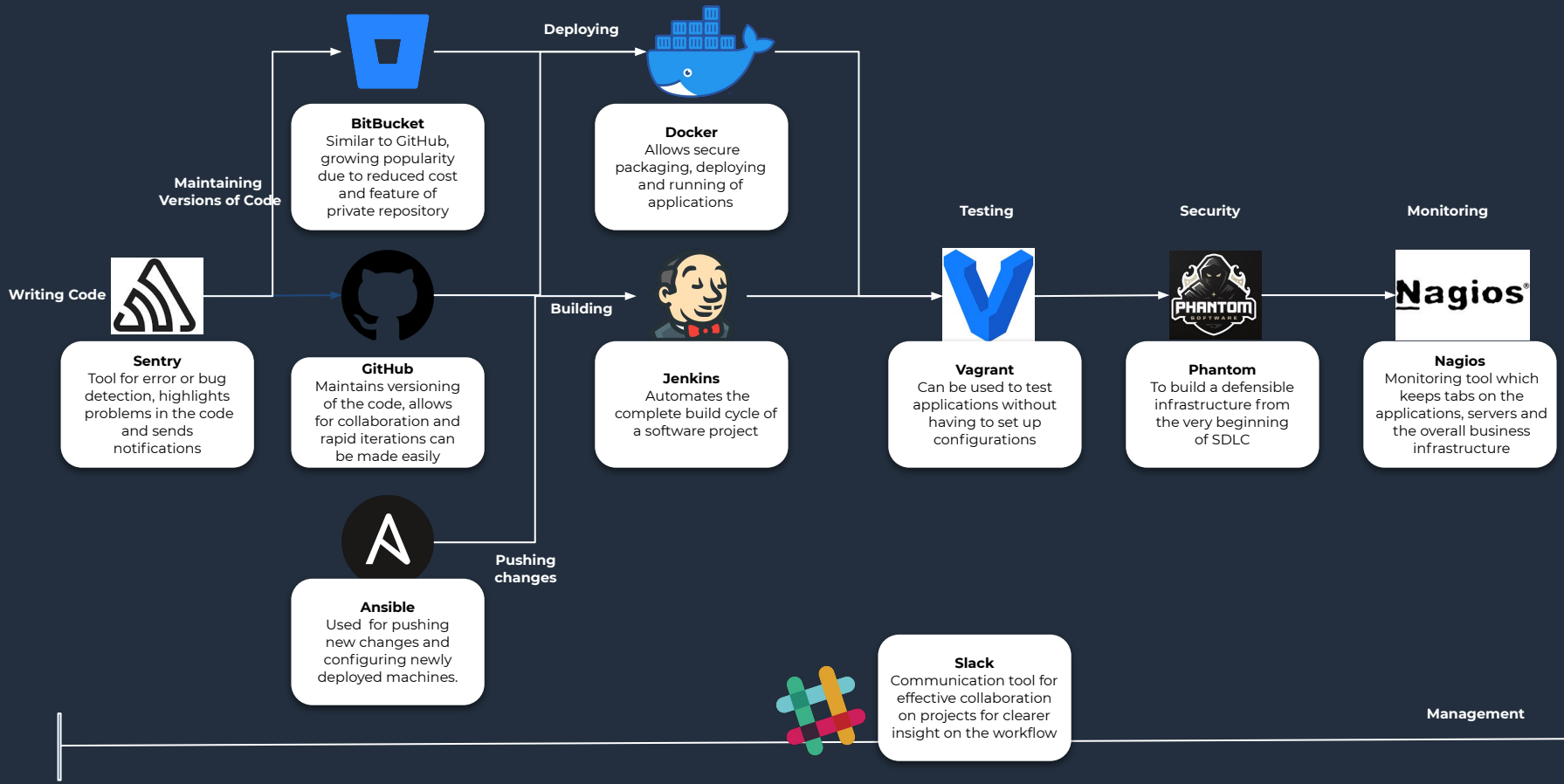
Where do you Start?



Where do you Start?



DevOps Products - Sample Life Cycle



Market Trends in DevOps

Automation

Zero Touch automation is a major focus

DevOps Assembly Lines

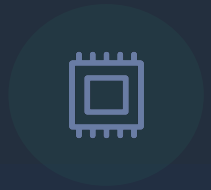
Shifting from CI to assembly lines that have perfect CD (continuous delivery) with interoperability

Rise in Artificial Intelligence

Increasing number of AI / ML companies are looking to DevOps to manage their apps

Kubernetes

Companies are embracing containers for running cloud-native apps



More Embedded Security

DevSecOps is to include security in the app development life cycle, decreasing vulnerabilities and improving business reputation

Microservices

Microservices are independent entities and they make adding new development features easy

Serverless Architecture

Using serverless architecture is time and cost efficient and enables developers to focus on the application

Everything as Code

Automating scripts that increases the software production cycle efficiency

What is Jenkins?

- Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes.
- Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build.
- It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies

Advantages of Jenkins

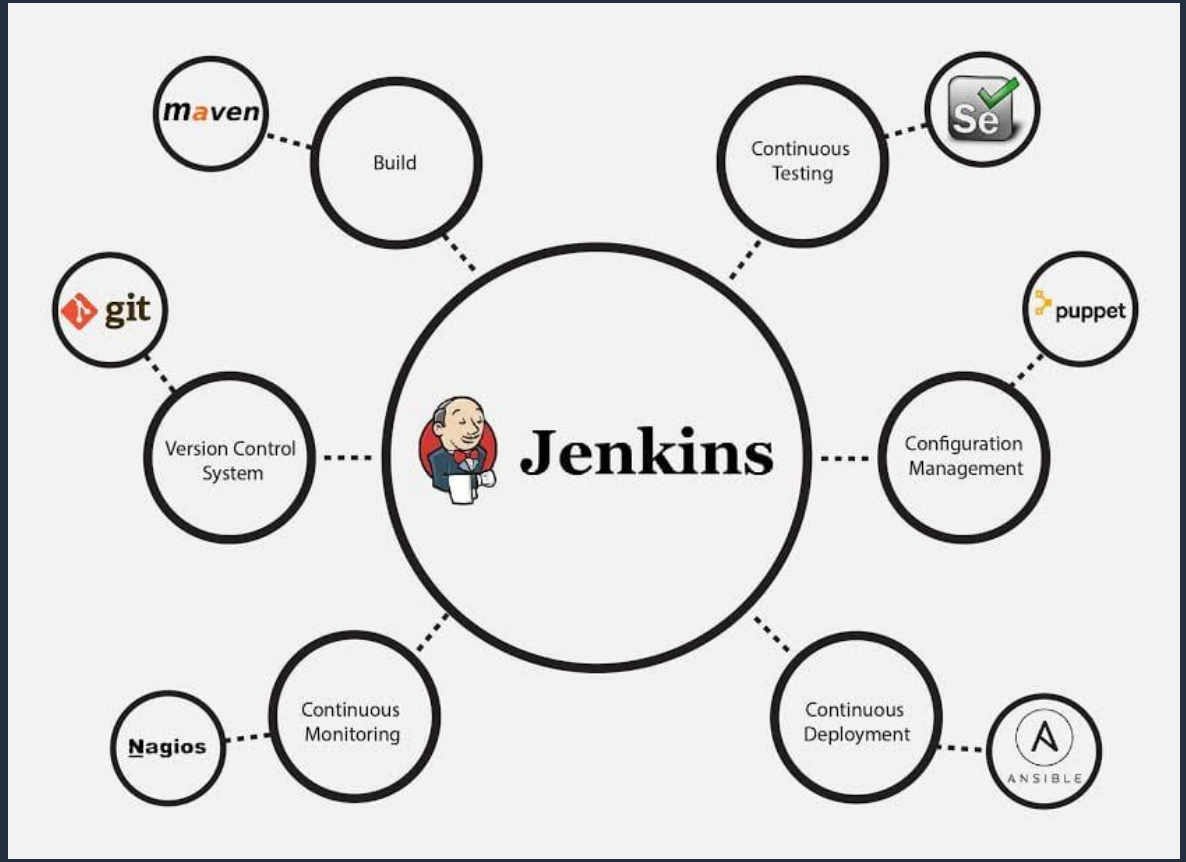
1000+ plugins

Fee of cost

Portable to platform

Open-source tool

Jenkins integration with DevOps



Impact of Jenkins

Before Jenkins	After Jenkins
The entire source code was built and then tested. Locating and fixing bugs in the event of build and test failure was difficult and time-consuming, which in turn slows the software delivery process.	Every commit made in the source code is built and tested. So, instead of checking the entire source code developers only need to focus on a particular commit. This leads to frequent new software releases.
Developers have to wait for test results	Developers know the test result of every commit made in the source code on the run.
The whole process is manual	You only need to commit changes to the source code and Jenkins will automate the rest of the process for you.

What is a Code Repository?

- A source-code repository is an archive with the code as well as the hosting facility for these software archives, where you can also have the project's technical documentation, web pages, snippets, patches, etc. which can be accessed publicly (open-source) or privately.

- What is the purpose of a code repository?
  - Version control for your project is critical
  - Code repositories are great community builders
  - Code repositories are great digital portfolios or resumes

- What can the code repository contain?
  - Commit objects' set
  - References to commit objects (also known as heads)
  - A historical record of the repository code changes

What is a Code Repository?

- Two important terms you should also know
  - Branch – when you want to diverge from the main code line, you can “branch” the code from the main “trunk.” If the original code is deleted, the branch has no support either.
  - Fork – when you want to go a completely different way, you can fork the main repository, which means you copy or clone the original and take it from there. The fork, unlike a branch, is not relying on the original repository and will remain if the source was deleted. It becomes its own new trunk.

What is Github?

GitHub is one of the most popular code repositories out there. It is a great repository for documentation purposes and is commonly used for collaboration. If you are working in teams, with other developers, GitHub is a great starting spot.

GitHub offers a list of popular features:

- Commit history is visible
- Pull requests
- Issue tracking
- Email notifications

What is Gitlab?

GitLab is also a very common repository. Just as GitHub does, GitLab offers many of the same features: visible commit history, pull requests, issue tracking.

A bonus of GitLab is that it offers a free tier for private repositories. If what you are working on contains sensitive information and you want to have more security right off the bat, then GitLab may be the option best suited for you.

What is Bitbucket?

Like GitLab, Bitbucket also offers a free tier for private repositories. Another bonus to Bitbucket is that it has a REST API to build 3rd party applications. You can also search for code in a dynamic and efficient manner.

Like GitHub, Bitbucket offers many similar features:

- Commit history
- Pull requests
- Issue tracking

Installing Jenkins

Step 1. Firstly, add Jenkins repository using wget, so that yum get to know where to install Jenkins from.

```
sudo yum install wget
sudo amazon-linux-extras install java-openjdk11
amazon-linux-extras install epel -y
```

Step 2. Now, let's add the Jenkins GPG key to our trusted keys, so that we will be able to verify/trust the files that are being sourced (while installing Jenkins)are from trusted site.

```
sudo wget -O /etc/yum.repos.d/jenkins.repo
http://pkg.jenkins-ci.org/redhat/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo yum install jenkins
```

Step 3. Jenkins services needs to be started, with the following command:

```
sudo service jenkins start
```

Installing Jenkins

Step 4. Make sure to open port 8080 (default port to which Jenkins listen):

(i) Go to your AWS management console → ec2 dashboard → Network & Security → Choose the security group of your instance

Step 5. Go to your browser and connect to jenkins via default port 8080

http://<IP_address>:8080
IP_address : you can use public DNS of your ec2 linux instance

Step 6. To unlock jenkins fetch the administrator password by typing following command:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Step 7. Click on 'Install suggested plugins' in the customize Jenkins window.

Step 8. Create first admin user:

THANK YOU