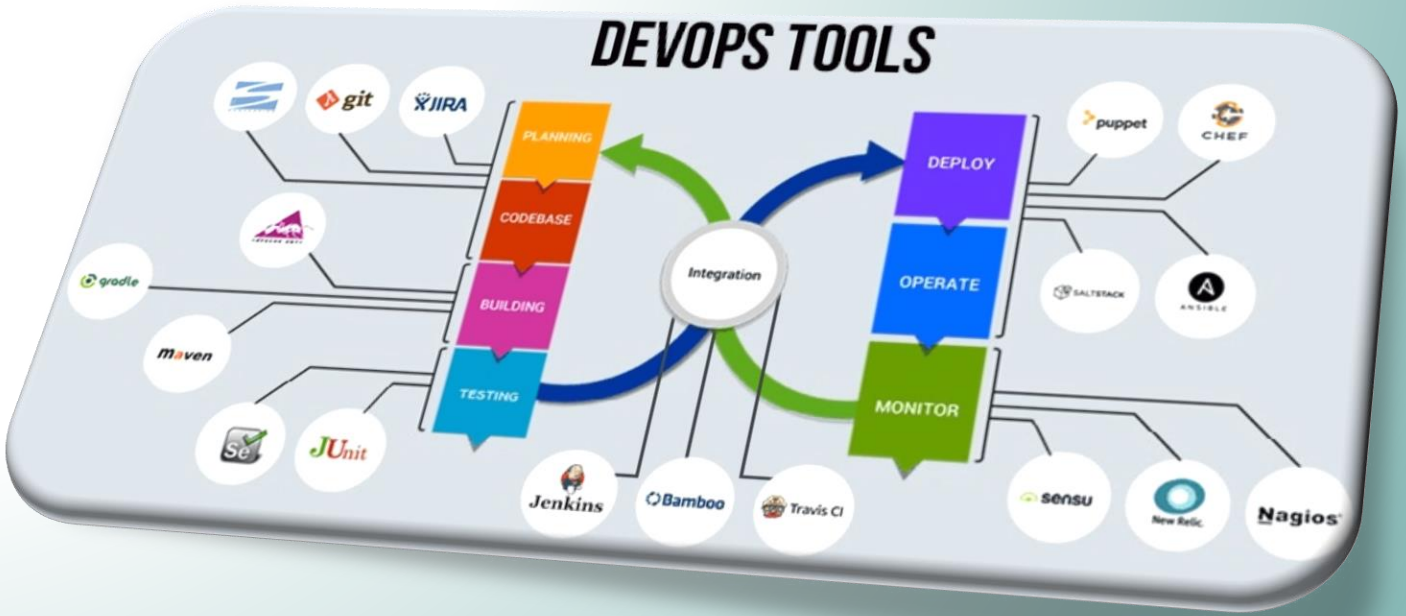
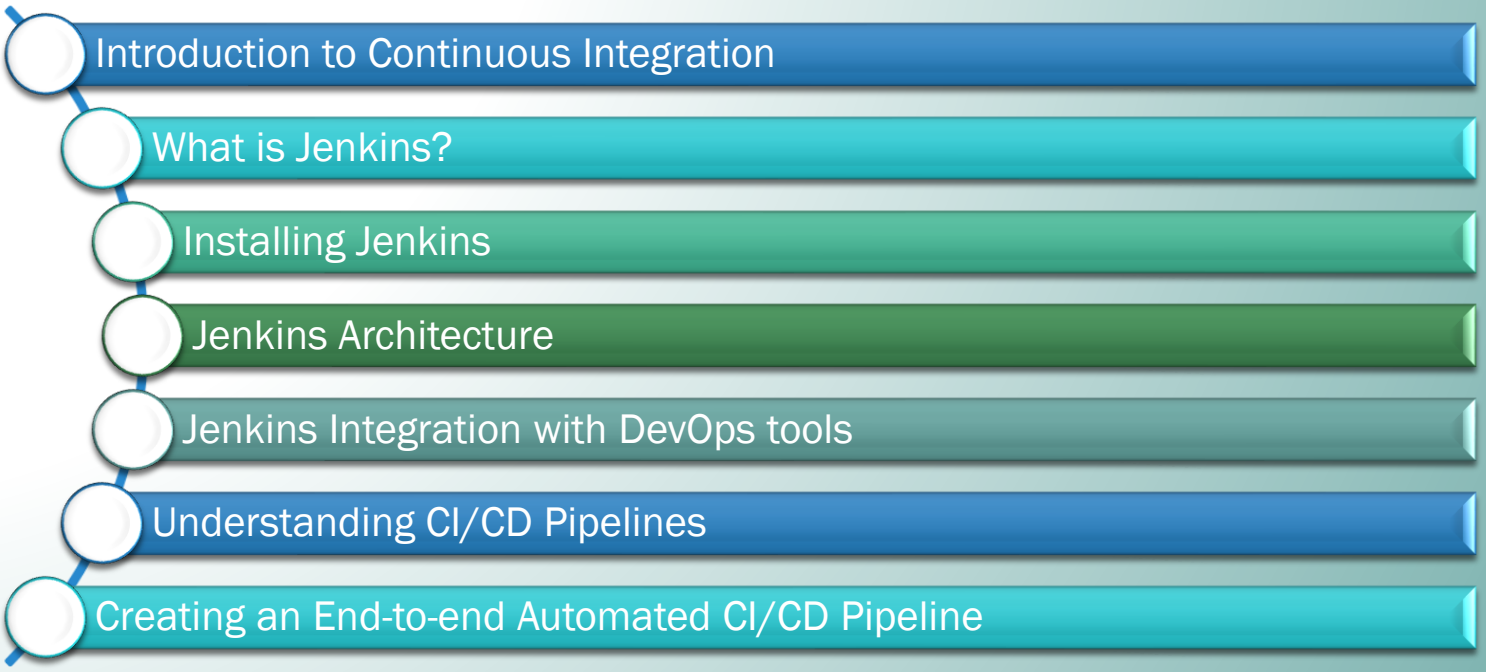
Continuous Integration [Jenkins]



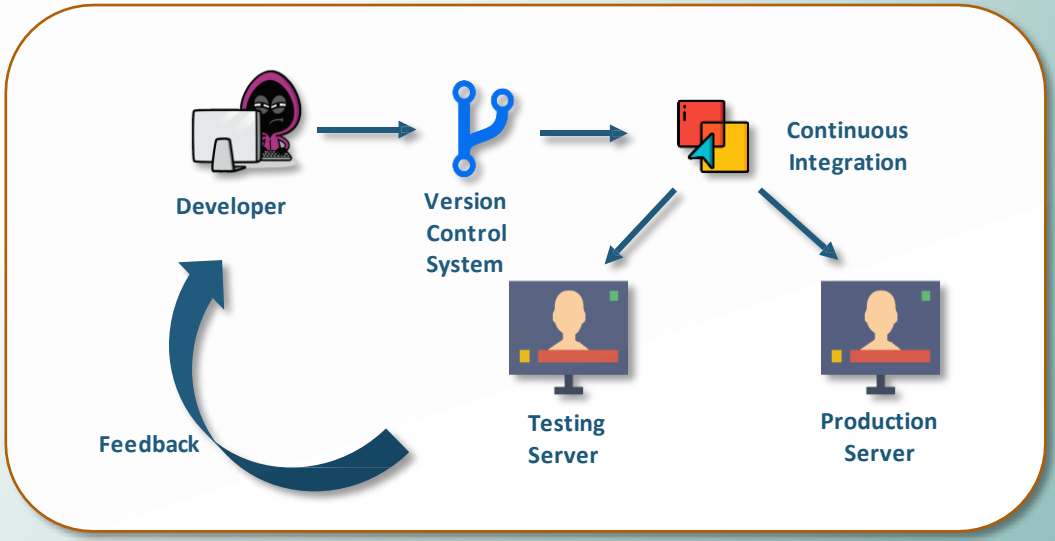
AGENDA


Introduction to Continuous Integration
What is Jenkins?
Installing Jenkins
Jenkins Architecture
Jenkins Integration with DevOps tools
Understanding CI/CD Pipelines
Creating an End-to-end Automated CI/CD Pipeline

WHAT IS CONTINUOUS INTEGRATION?

WHAT IS CONTINUOUS INTEGRATION?

The process of having shorter release cycles (sometimes, several times a day), i.e., creating small features and integrating them to the source code and employing automated build and test processes for quicker feedback is called Continuous Integration.



ADVANTAGES OF CONTINUOUS INTEGRATION

- Frequent Commits, hence small feature release
- Automated Build and Testing
- Instant feedback to the developer
- Low risk and faster delivery

WHAT IS JENKINS?

What is Jenkins?

Jenkins is an open-source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery.

FEATURES OF JENKINS

Adoption: Jenkins is extremely popular among the open-source community; hence, there are more than 147,000 active installations throughout the world and 1 million people are using it.

Plugins Support: With an extremely active open-source community, Jenkins has around 1000 plugins that allow it to integrate with most of the development, testing and deployment tools.

ADVANTAGES OF JENKINS

Before Jenkins

- Locating and fixing bugs in the event of build and test failure was difficult and time consuming.
- Tests were triggered manually.
- No central place for triggering jobs on remote systems.

After Jenkins

- Smaller and automated continuous build and testing make the task accurate and faster.
- Developers have to just commit the code to the remote repository, build, test and deployment happen automatically.
- All builds or tests on multiple remote systems can be controlled from one place.

INSTALLING JENKINS

INSTALLING JENKINS ON AWS

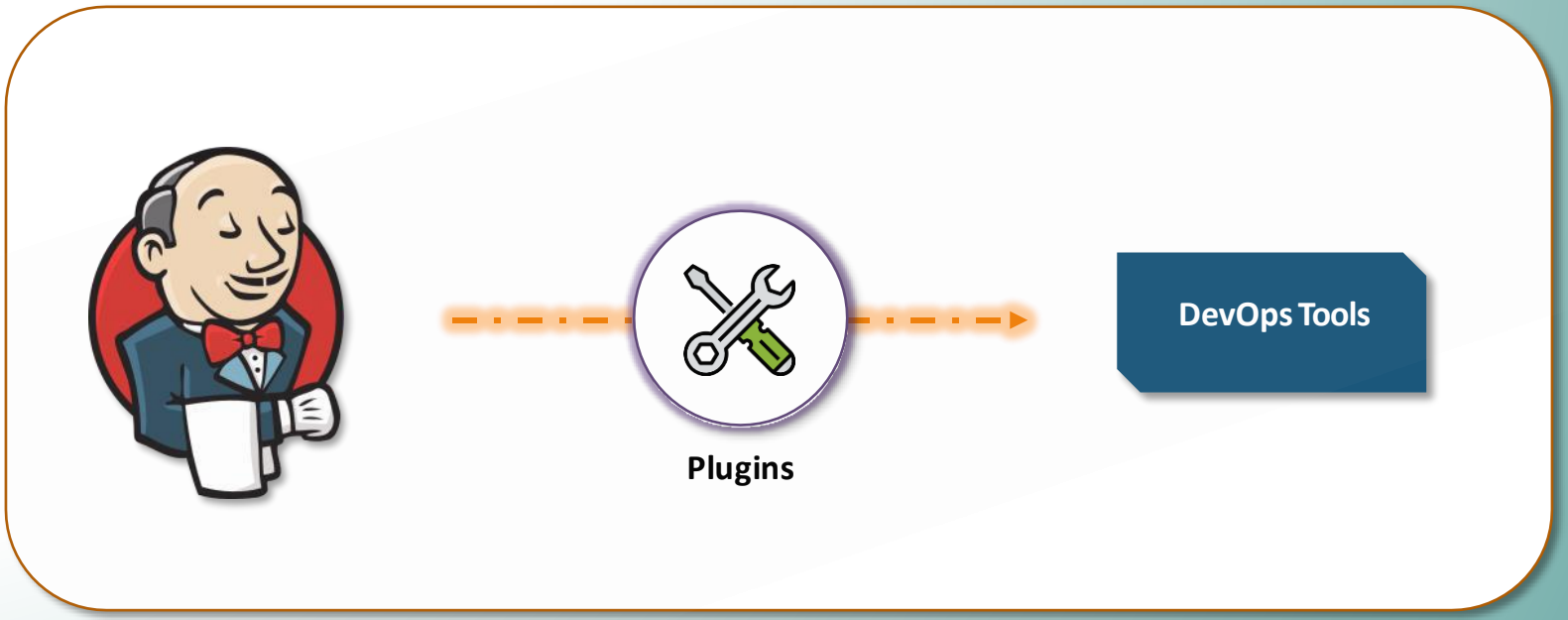
1. Launch an AWS Instance
2. Connect through SSH
3. Execute the following commands:

Jenkins Installation:

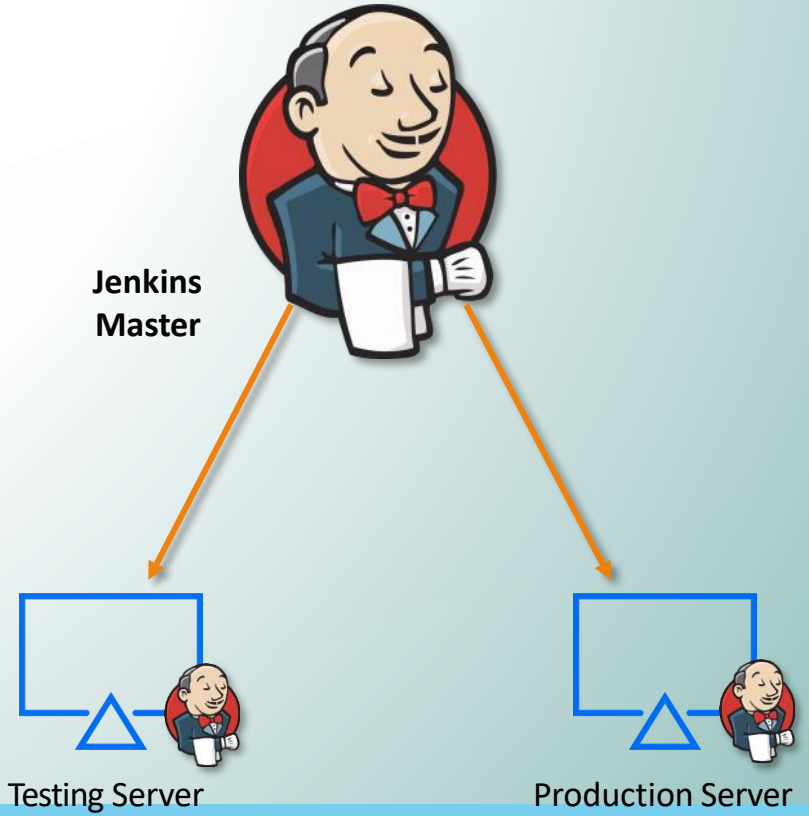
```
$> sudo apt-get update
$> sudo apt install openjdk-8-jdk
$> wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
$> sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ >
/etc/apt/sources.list.d/jenkins.list'
$> sudo apt update
$> sudo apt install jenkins
```

JENKINS ARCHITECTURE

JENKINS ARCHITECTURE

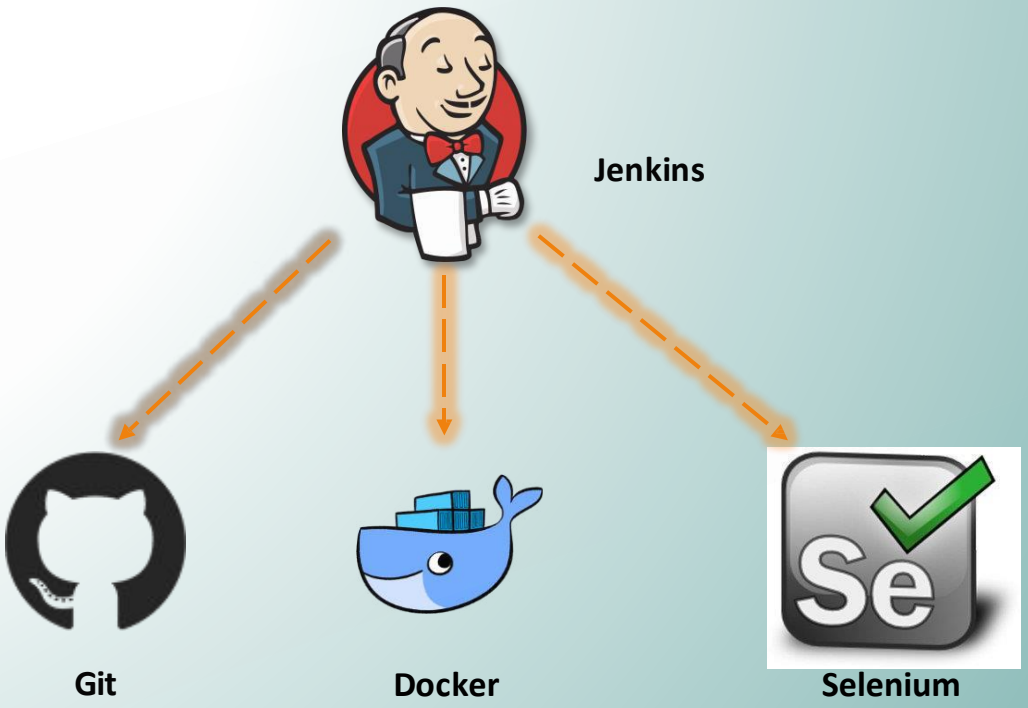


JENKINS ARCHITECTURE



JENKINS INTEGRATION WITH DEVOPS TOOLS

JENKINS INTEGRATION WITH DEVOPS TOOLS



JENKINS INTEGRATION WITH DEVOPS TOOLS

Git

Docker

Selenium

Copy a Git repository to the slave's filesystem from Jenkins master

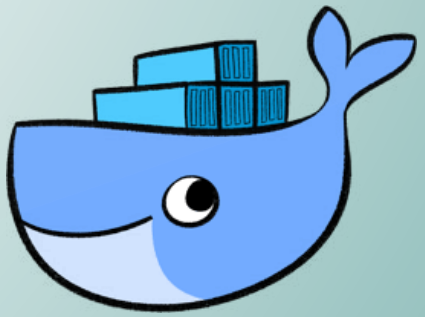
JENKINS INTEGRATION WITH DEVOPS TOOLS

Git

Docker

Selenium

Containerize the website in the previous step to a Docker Container using Jenkins



JENKINS INTEGRATION WITH DEVOPS TOOLS

Git

Docker

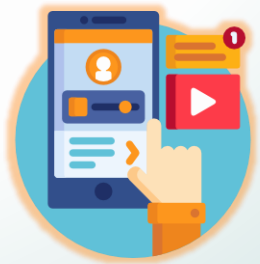
Selenium

Create a test case for the website in the previous step and execute the test on the slave using Jenkins

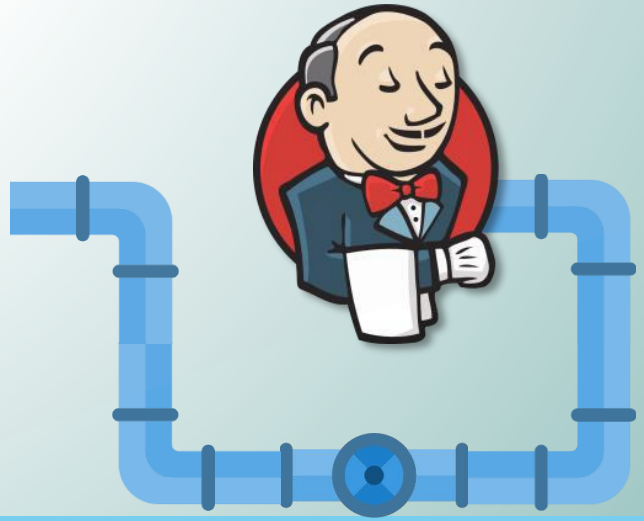
UNDERSTANDING CI/CD PIPELINES

WHAT ARE CI/CD PIPELINES?

CI/CD Pipelines, i.e., Continuous Integration, Continuous Delivery and Deployment pipelines, are a way of running Jenkins jobs in a sequence, which resembles a pipeline view.



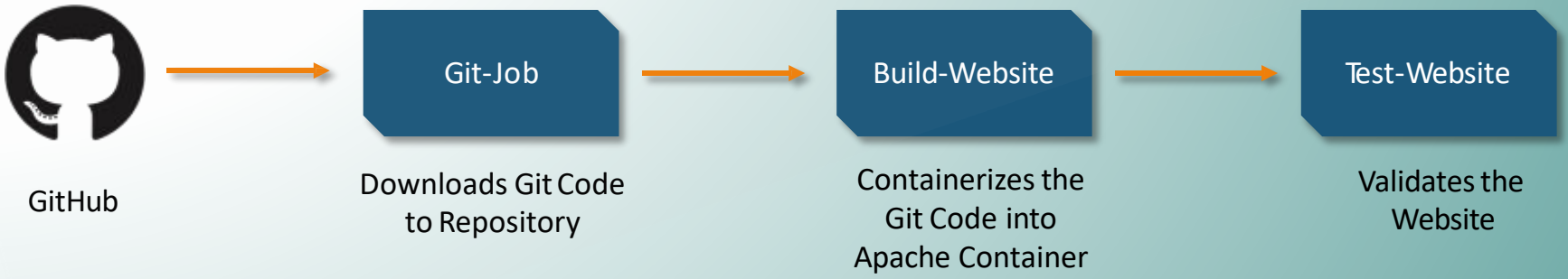
Finished Application



WHAT ARE CI/CD PIPELINES?

CI/CD Pipelines, i.e., Continuous Integration, Continuous Delivery and Deployment pipelines, are a way of running Jenkins jobs in a sequence, which resembles a pipeline view.

For Example:



CREATING AUTOMATED CI/CD PIPELINE

CREATING AN AUTOMATED CI/CD PIPELINE

1. Initiate a Git Webhook for the Jenkin's git-job repository
2. Trigger the jobs after the completion of previous jobs with the following map: Git-Job → Build-Website → Website-Test
3. Install the plugin for the pipeline view
4. Make changes to the website and commit the job to see the changes

Thank you