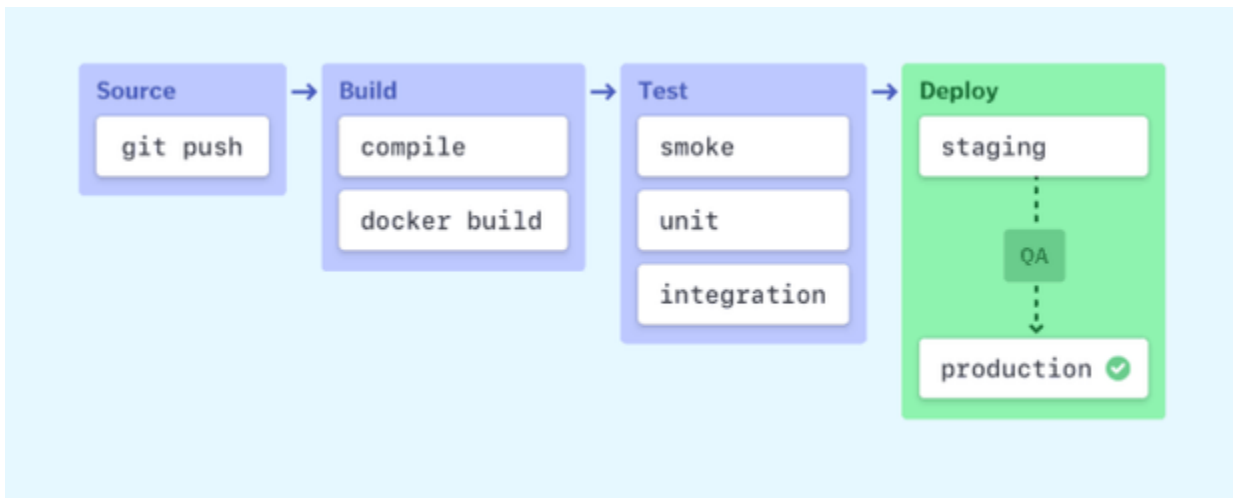


# How to - Standardize Onboarding process

## End To End Workflow using CI/CD Pipeline Bitbucket and Gitlab

Devops	Tools	Version
CI/CD	Jenkins	- 2.176
Version Control system	- Bitbucket - Gitlab	-v5.12.2 - 12.2.5-ee
Build	- Scala - R - Python	- 2.13.1 - 3.6.1 - 3.7.4
Containerization	Docker	- 17.03.1-ce-rc1
Configuration Management	Ansible	- 2.8.6
Artifacts Management	Nexus	- 3.15.2-01
Code Quality Checks	Sonar	- 6.7.5

Most software releases go through a couple of typical stages



Continuous Integration with stages:



## 1. Source stage

In most cases a pipeline run is triggered by a source code repository. A change in code triggers a notification to the CI/CD tool, which runs the corresponding pipeline.

We have a Bitbucket & Gitlab repository where the development team will commit the code.

### BitBucket Repo

#### 1.1 Repository creation steps for BitBucket:

Please refer document [How to- repository configuration for default branch& merge hook-bit bucket.](#)

#### 1.2 Sample HelloWorld project creation for Scala Code

Please refer document [How to - Bitbucket repo setup - Create sample scala repo](#)

### Gitlab Repo

#### 1.3 Repository Creation steps for Gitlab:

please refer document [How to - GitLab Repository Creation](#)

#### 1.4. Sample HelloWorld project creation for Python

Please refer document [GitLab: Setup sample Python repository](#)

## 2.Build stage

From Bitbucket/Gitlab, Jenkins pulls the code and then Jenkins moves it into the **commit phase**, where the code is committed from every branch. The **build phase** is where we compile the code.

### 2.1 Jenkins installation and configuration and Build scala code using Jenkins file

Please refer document [How to - Install jenkins and configuration on Cloud](#)

### 2.2 Jenkins integration for R code base

Please refer document [Jenkins R Build Support](#)

## 3.Test stage

In this phase we run automated tests to validate the correctness of our code and the behavior of our product. The test stage acts as a safety net that prevents easily reproducible bugs from reaching the end users.

### 3.1 Integration of sonar for scala code

please refer document [How To - jenkins -Sonar integration for Scala](#)

### 3.2 Integration of sonar for Python

please refer document [How To - PYTHON ONBOARDING](#)

## 4. Package

In this phase pushing artifacts in to nexus repository

### 4.1 Pushing the artifacts in to nexus/Jfrog repository

please refer documents [Nexus Artifactory Configurations With Basic Maven Repository](#)

[Jfrog Artifactory- Jenkins Integration:-](#)

## 5. Deploy stage

Once we have a built a runnable instance of our code that has passed all predefined tests, we're ready to deploy it

There are usually multiple deploy environments, for example Test, Staging & UAT