

DevOps



Caltech

Center for Technology &
Management Education

Postgraduate Program in DevOps



Automated Deployment

Learning Objectives

By the end of this lesson, you will be able to:

- Implement automated and continuous deployment
- Launch a simple web application using Tomcat
- Deploy scripting-based PHP application

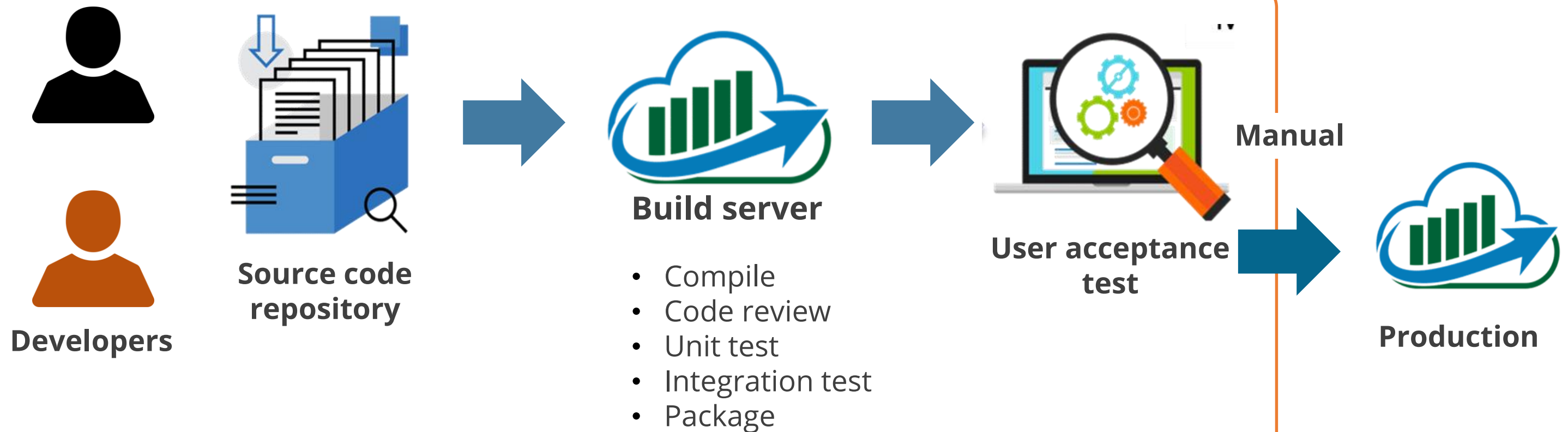


Introduction to Automated Deployment and Continuous Delivery

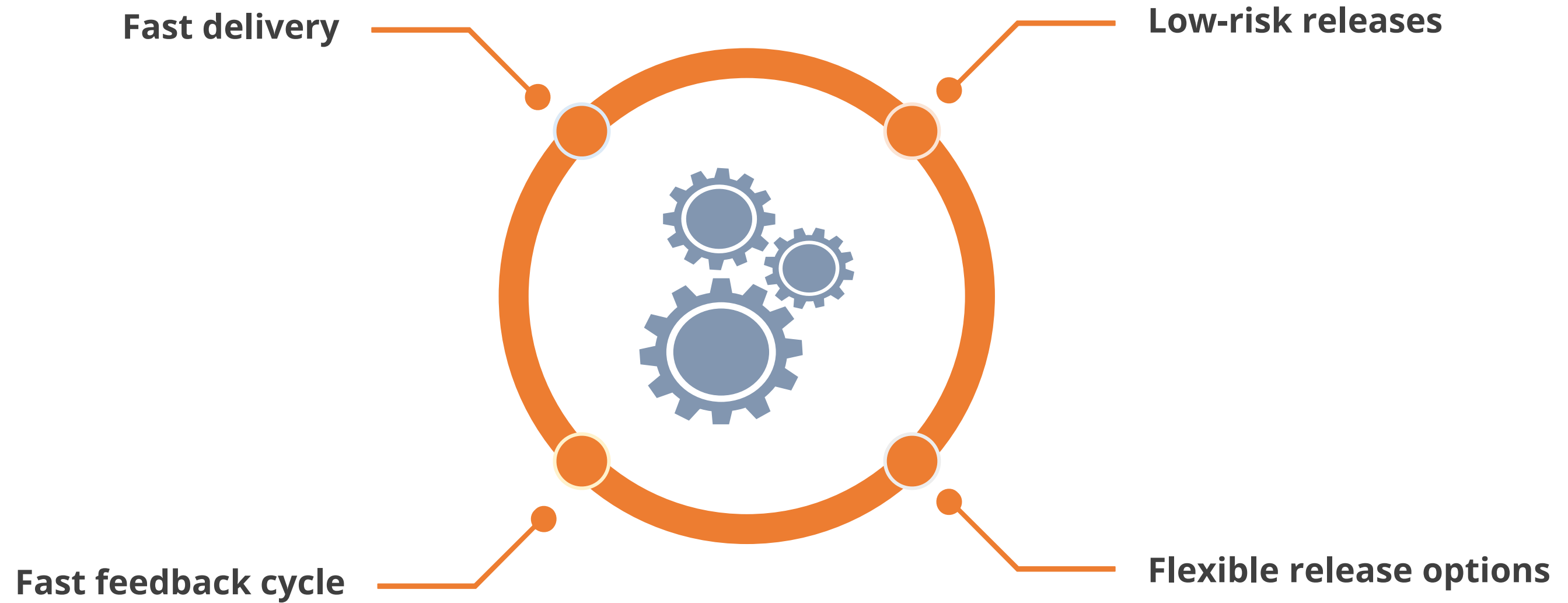
Continuous Delivery

Continuous Delivery is the ability to get changes of all types, including new features, configuration changes, bug fixes, and experiments into production, or into the hands of users, safely and quickly in a sustainable way.

–Jez Humble

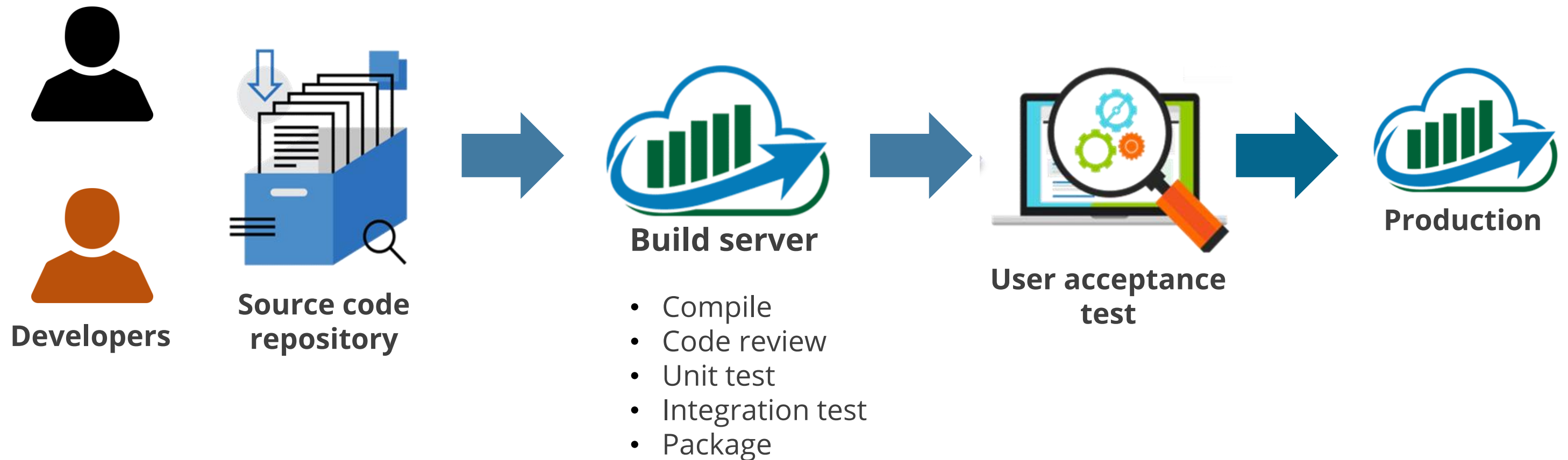


Benefits of Continuous Delivery

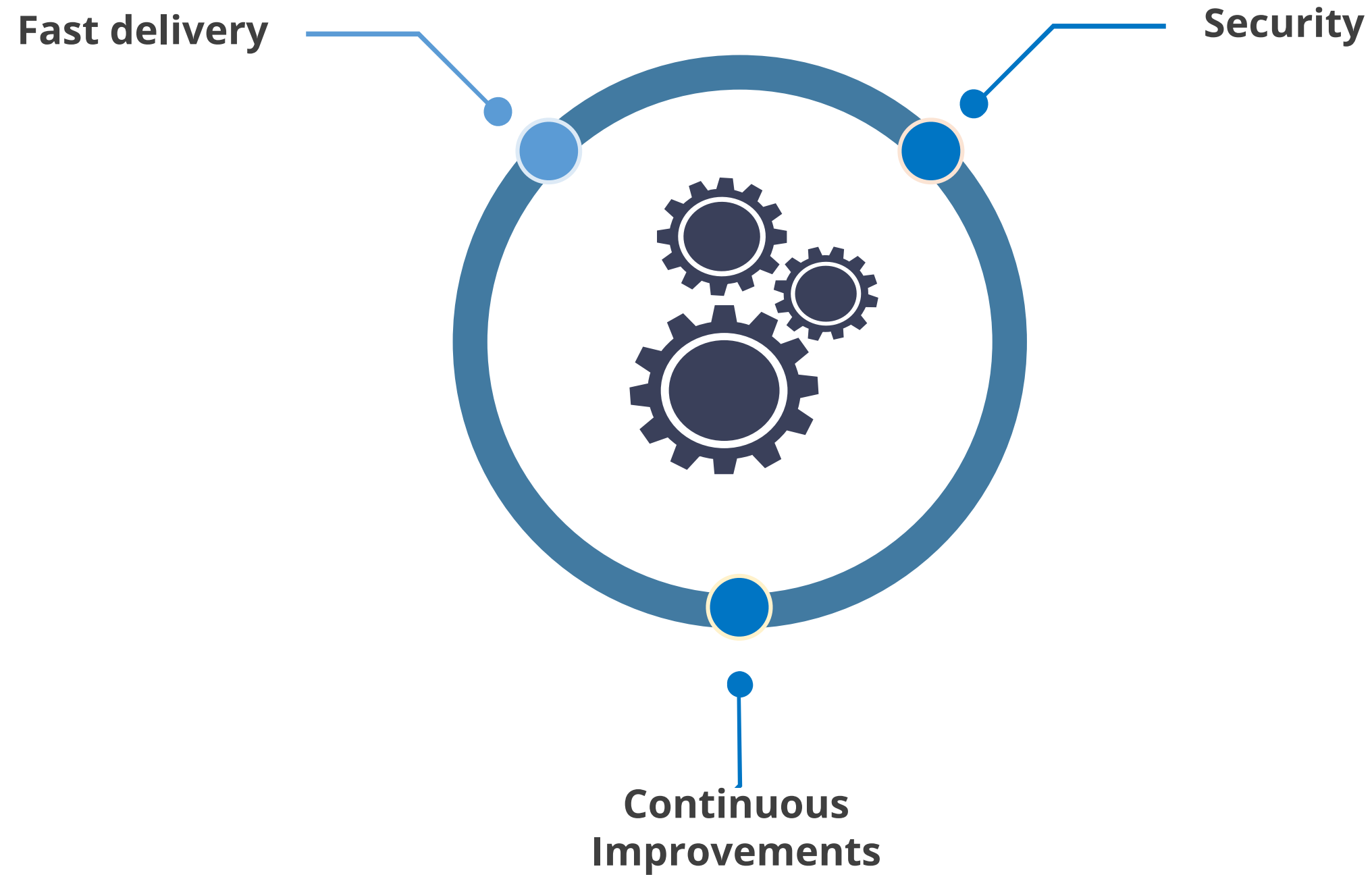


Automated Deployment

The process wherein any code change subject to automated tests and other appropriate verifications is immediately deployed into production.

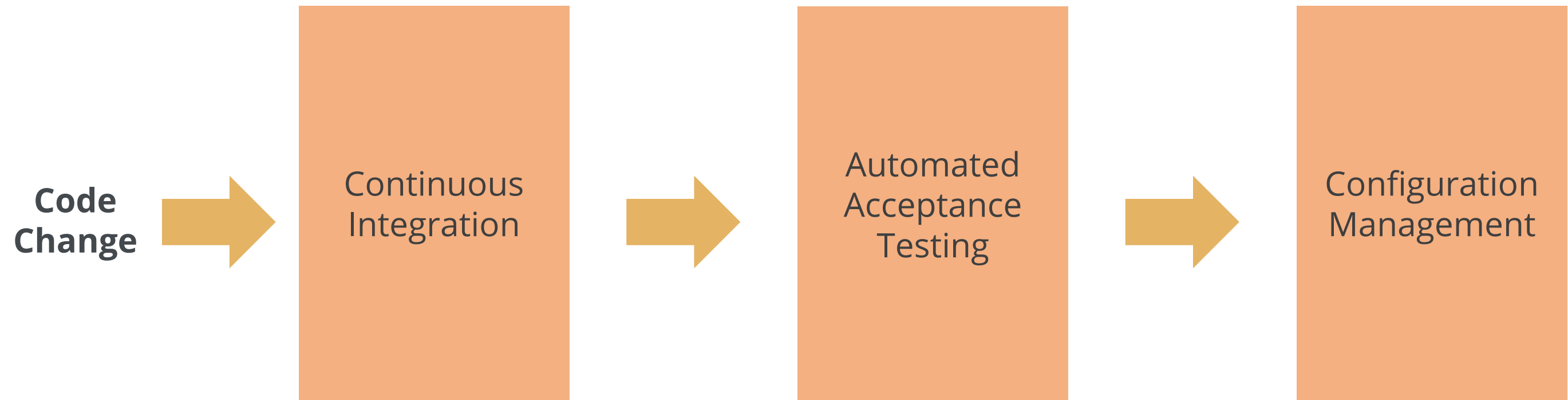


Benefits of Automated Deployment



Automated Deployment Pipeline

The automated pipeline deployment is a sequence of scripts executed following each change of code committed to the repository. If the process succeeds, the deployment ends in the production environment.



Phases of Automated Deployment Pipeline

Each step below corresponds to a phase in the traditional delivery process:

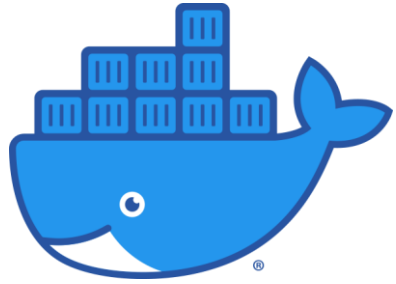
Continuous Integration: Makes sure that the code written by different developers integrates together.

Automated Acceptance Testing: Replaces the manual QA phase and checks if the features implemented by developers meet the client's requirements.

Configuration Management: Replaces the manual operations phase, configures the environment, and deploys the software.

Building the Continuous Delivery Process

Tools Used



Docker ecosystem



Jenkins



Ansible



Java



GitHub



Gradle



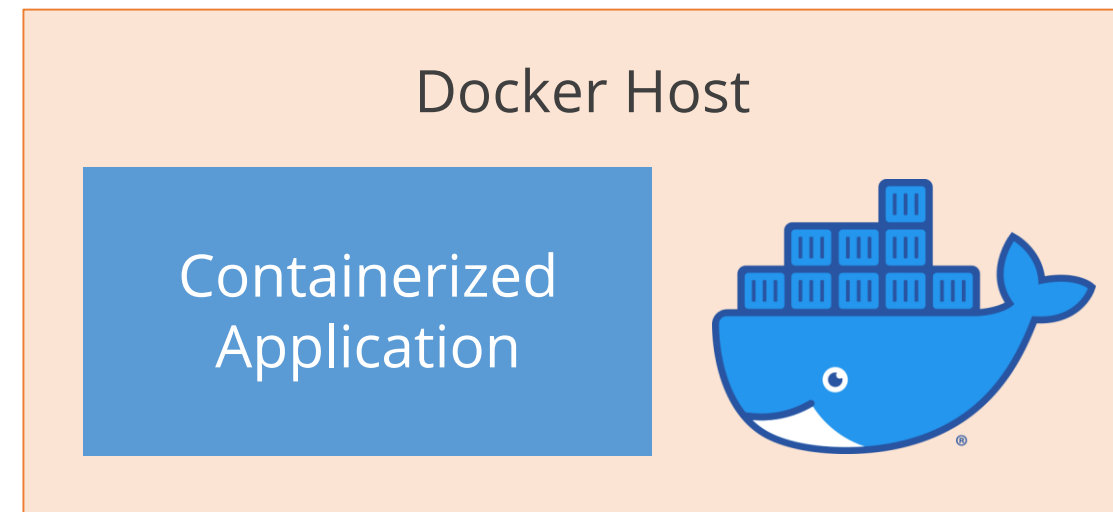
Spring boot



Cucumber

Automated Deployment System

A dockerized application (web service) is made to run as a container on a Docker Host. This is reachable as it would run directly on the host machine.



Dockerized application

Jenkins configuration

Continuous integration

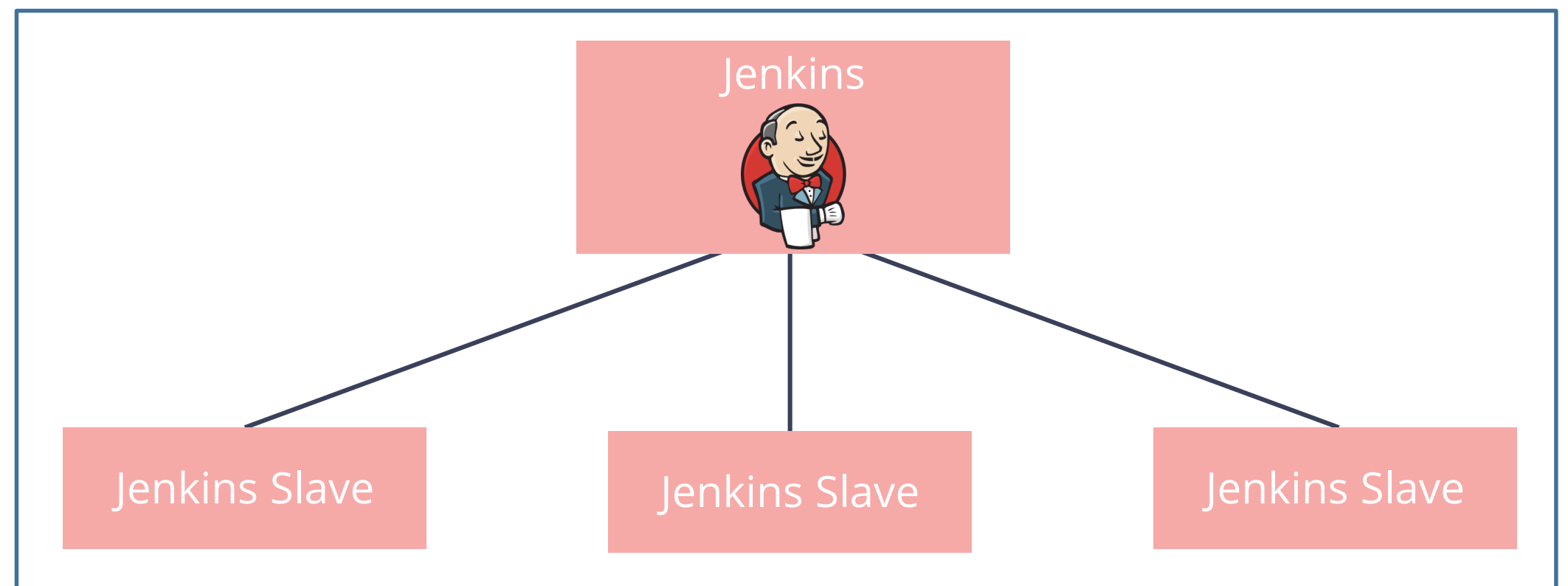
Automated acceptance testing

Configuration management

Advanced continuous delivery

Automated Deployment System

The Jenkins master accepts a build request, but the execution is started at one of the Jenkins Slave (agent) machines. This provides horizontal scaling of the Jenkins environment.



Dockerized application

Jenkins configuration

Continuous integration

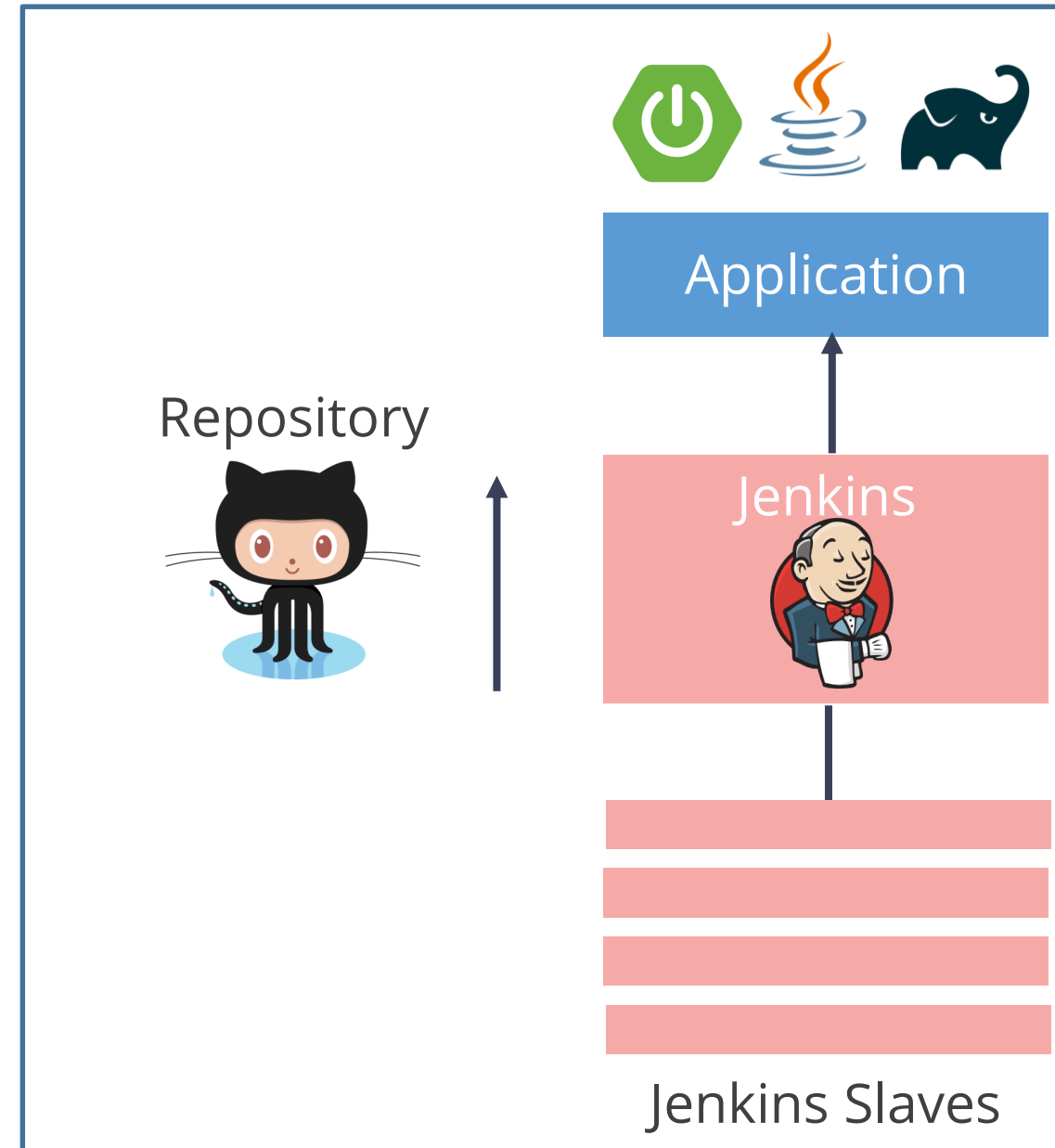
Automated acceptance testing

Configuration management

Advanced continuous delivery

Automated Deployment System

The application is a simple web service written in Java with the Spring Boot framework. Gradle is used as a build tool and GitHub as the source code repository.



Dockerized application

Jenkins configuration

Continuous integration

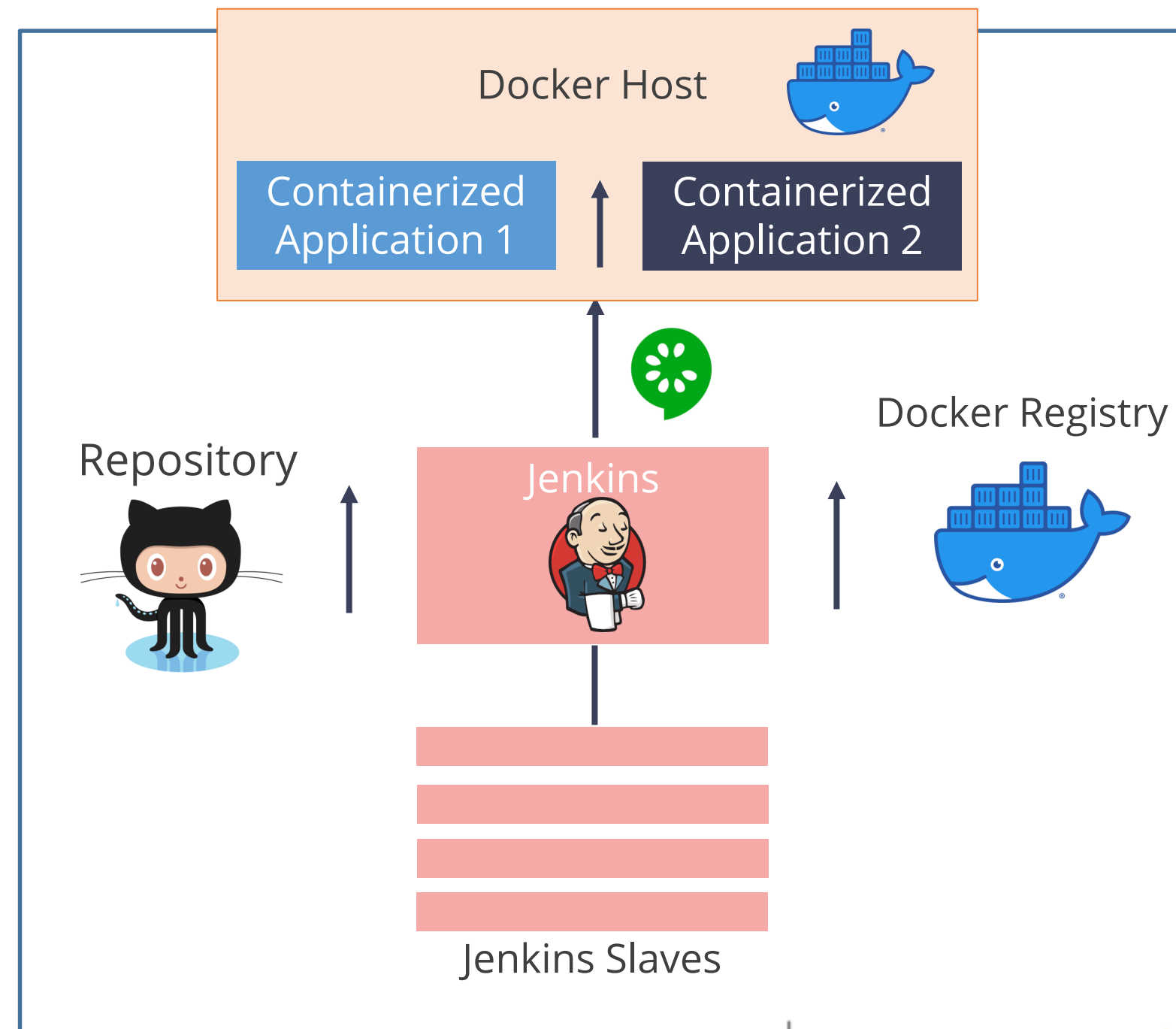
Automated acceptance testing

Configuration management

Advanced continuous delivery

Automated Deployment System

After the application is started on the Docker Host, Jenkins runs a suite of acceptance tests written in the Cucumber framework.



Dockerized application

Jenkins configuration

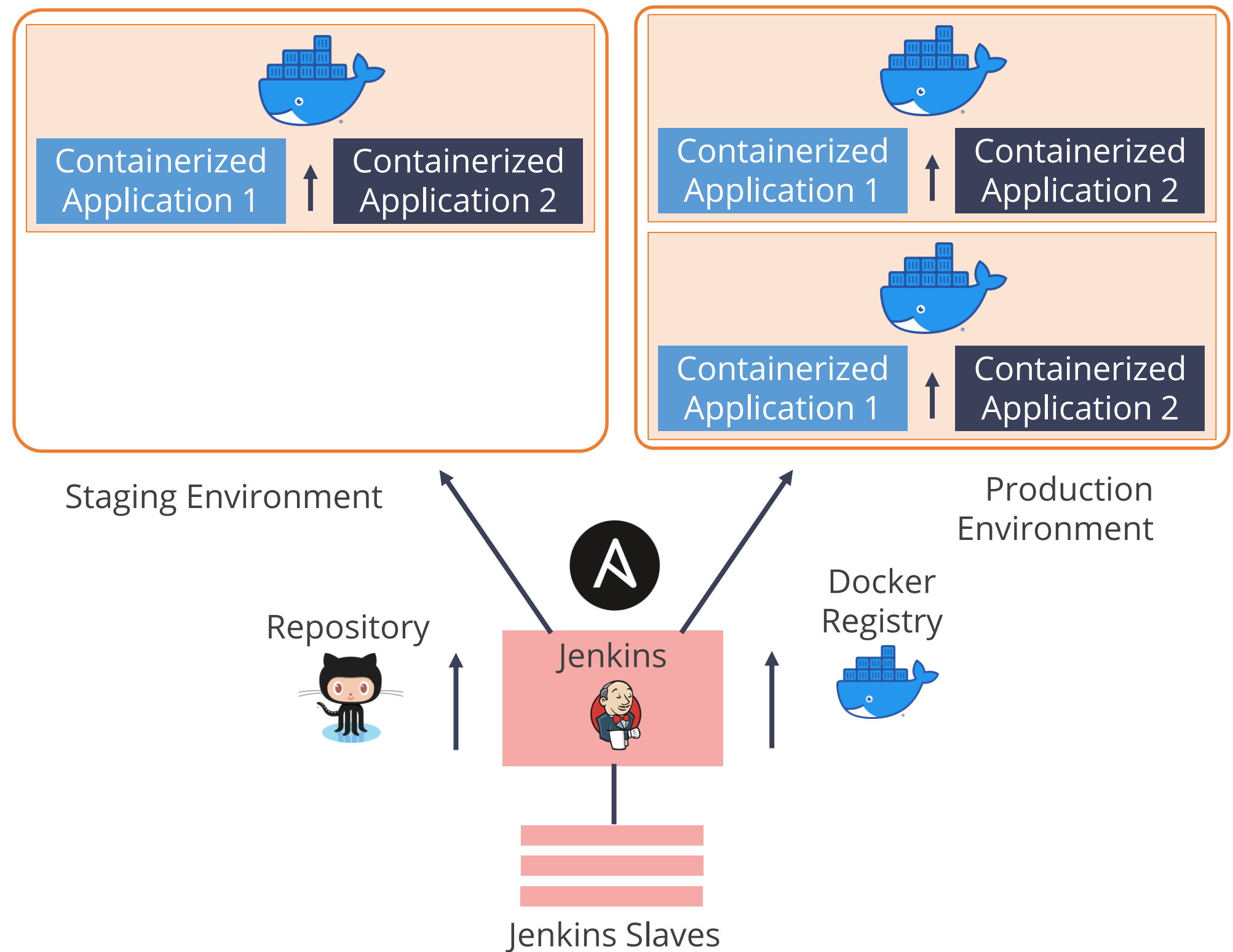
Continuous integration

Automated acceptance testing

Configuration management

Advanced continuous delivery

Automated Deployment System



Dockerized application

Jenkins configuration

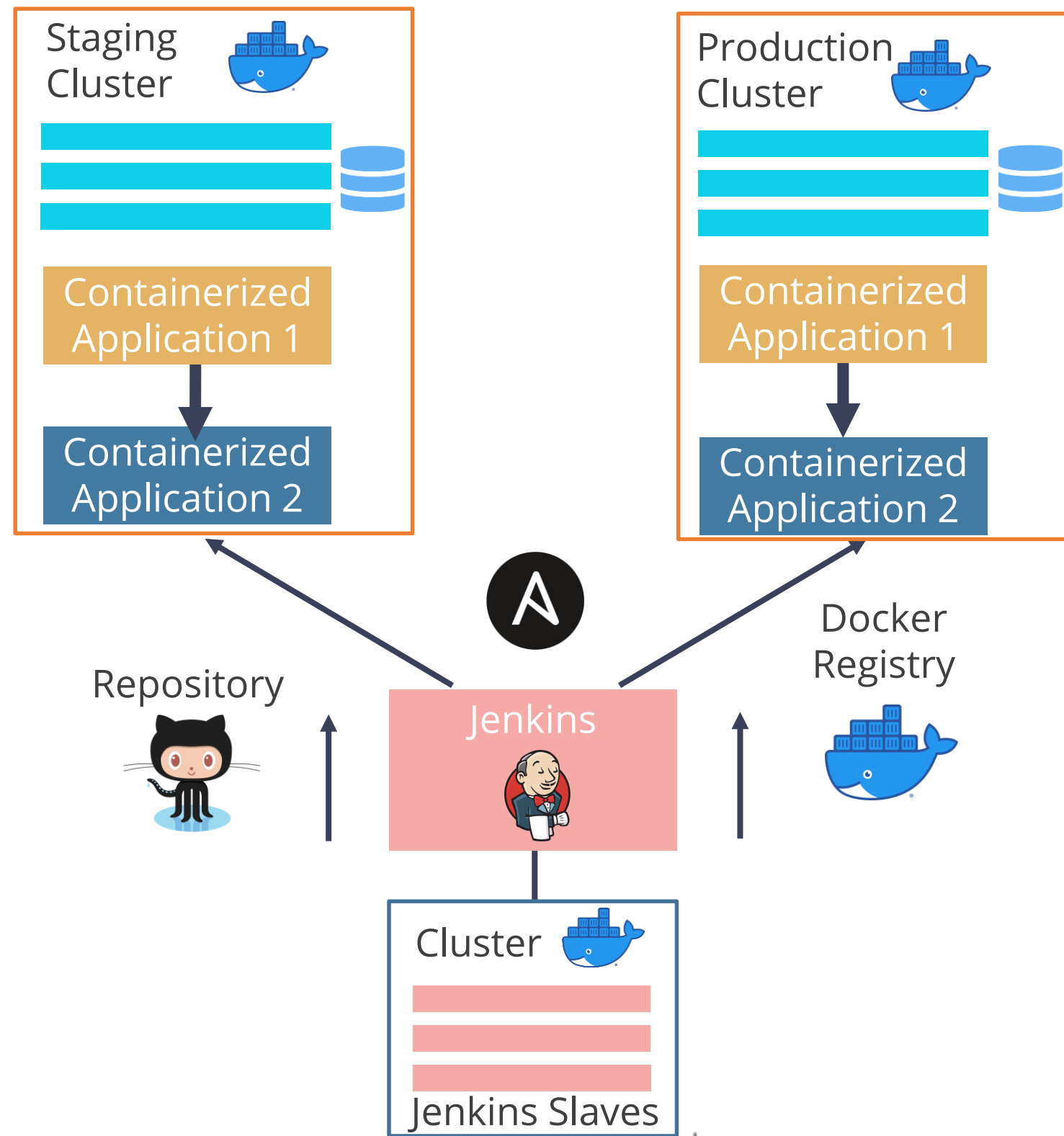
Continuous integration

Automated acceptance testing

Configuration management

Advanced continuous delivery

Automated Deployment System



Dockerized application

Jenkins configuration

Continuous integration

Automated acceptance testing

Configuration management

Advanced continuous delivery

Implementing Automated and Continuous Deployment

The Deployment Script

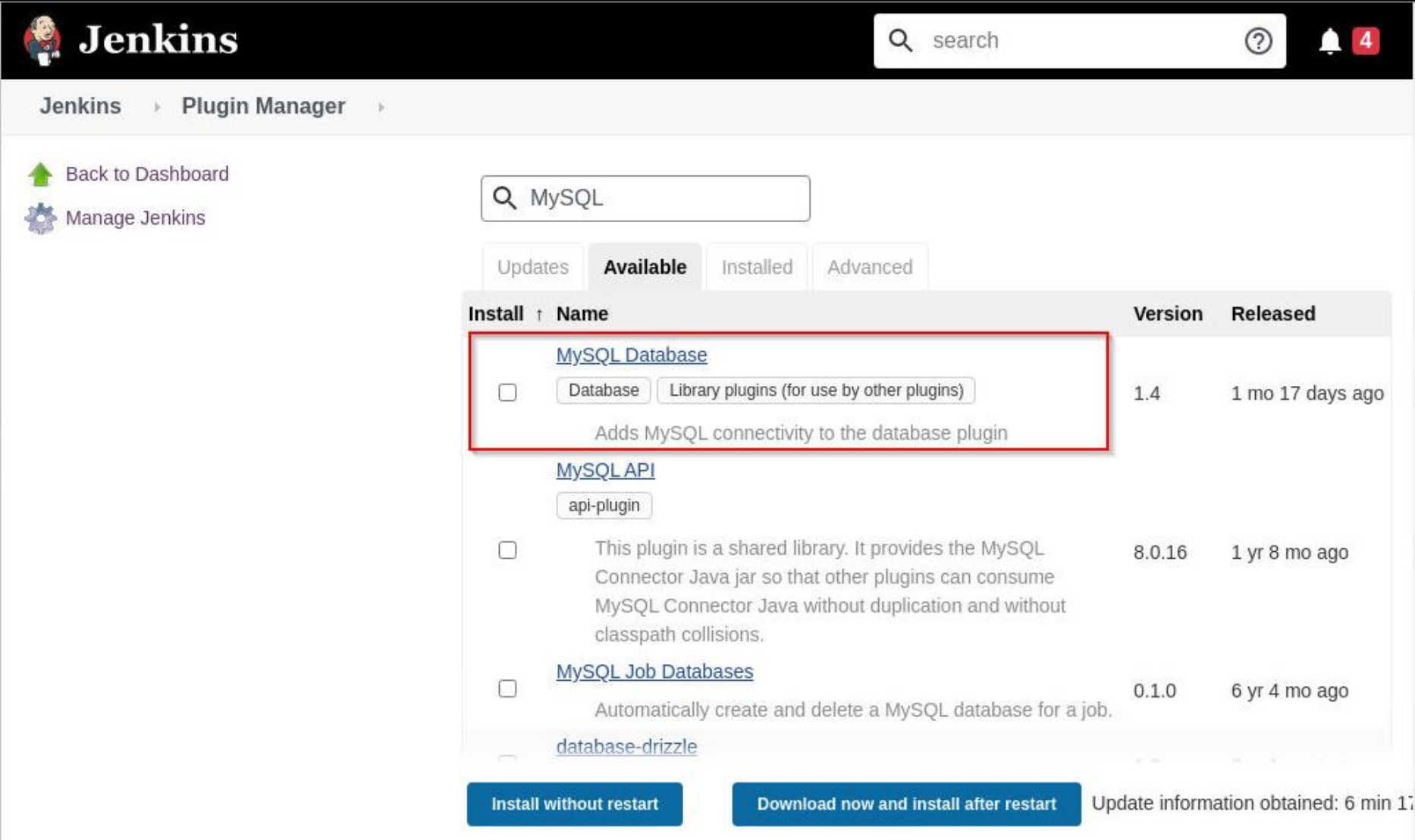
Scriptable deployment process is an important part in any automated deployment. Essentially, the deployment stage executes only after the successful completion of the **Build** and **Test** stages.

```
Jenkinsfile (Declarative Pipeline)
pipeline {
    agent any

    stages {
        stage('Deploy') {
            when {
                expression {
                    currentBuild.result == null || currentBuild.result ==
'SUCCESS'
                }
            }
            steps {
                sh 'make publish'
            }
        }
    }
}
```


Database Updates: MySQL Plugin

Database plugin adds a system configuration entry to let the administrator configure the database used by Jenkins. MySQL Database plugin is a driver plugin for Database plugin that adds MySQL database driver.

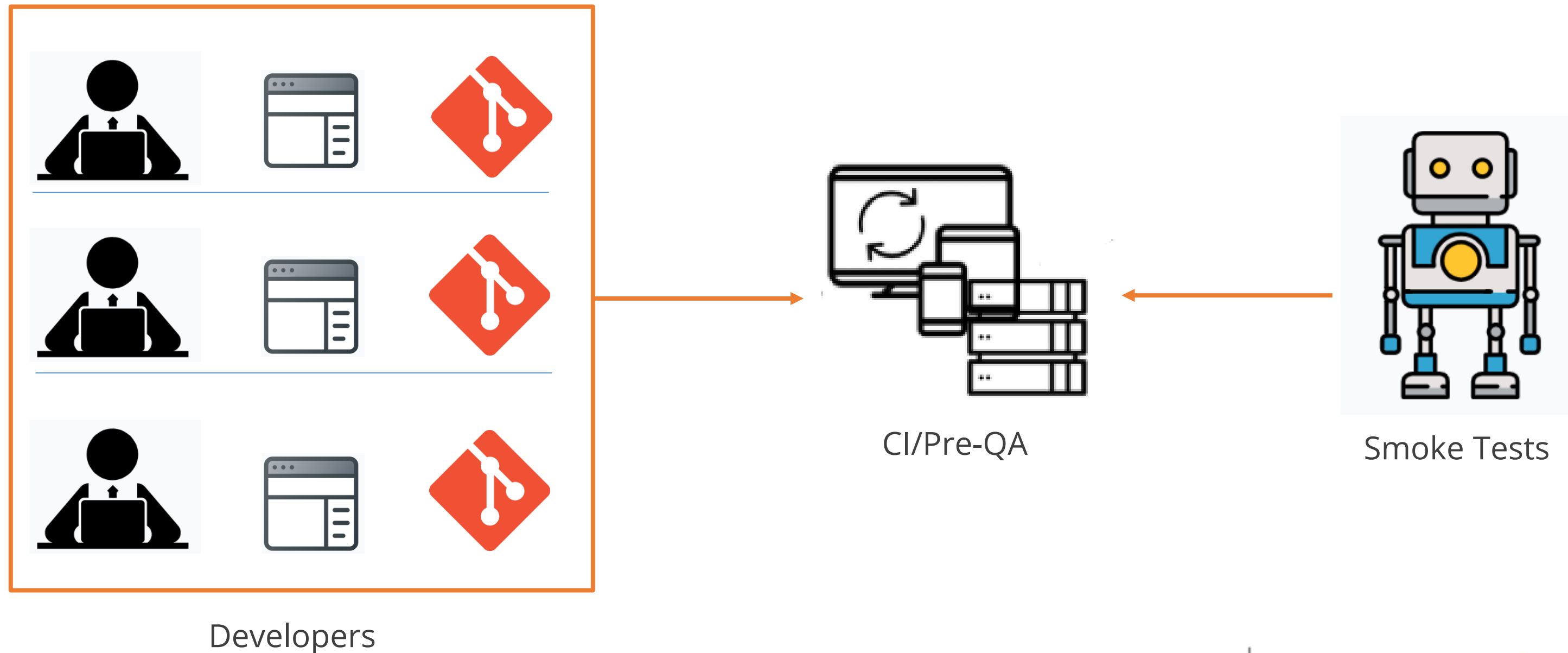


The screenshot shows the Jenkins Plugin Manager interface. The top navigation bar includes the Jenkins logo, a search bar, and a notification bell with 4 alerts. The main content area is titled 'Jenkins > Plugin Manager'. On the left, there are links for 'Back to Dashboard' and 'Manage Jenkins'. The central part of the interface features a search bar with 'MySQL' entered and tabs for 'Updates', 'Available', 'Installed', and 'Advanced'. Below these tabs is a table of available plugins. The first plugin, 'MySQL Database', is highlighted with a red box. It has a checkbox for installation, a description 'Adds MySQL connectivity to the database plugin', and a version of 1.4 released 1 month and 17 days ago. Other visible plugins include 'MySQL API' (version 8.0.16, released 1 year and 8 months ago) and 'MySQL Job Databases' (version 0.1.0, released 6 years and 4 months ago). At the bottom, there are buttons for 'Install without restart' and 'Download now and install after restart', along with a note that update information was obtained 6 minutes and 17 seconds ago.

Install	Name	Version	Released
<input type="checkbox"/>	MySQL Database Database Library plugins (for use by other plugins) Adds MySQL connectivity to the database plugin	1.4	1 mo 17 days ago
<input type="checkbox"/>	MySQL API api-plugin This plugin is a shared library. It provides the MySQL Connector Java jar so that other plugins can consume MySQL Connector Java without duplication and without classpath collisions.	8.0.16	1 yr 8 mo ago
<input type="checkbox"/>	MySQL Job Databases Automatically create and delete a MySQL database for a job.	0.1.0	6 yr 4 mo ago
<input type="checkbox"/>	database-drizzle		

Smoke Tests

A series of automated smoke checks must be executed for any significant automated implementation.
An automated acceptance test subset can be a good smoke test candidate.

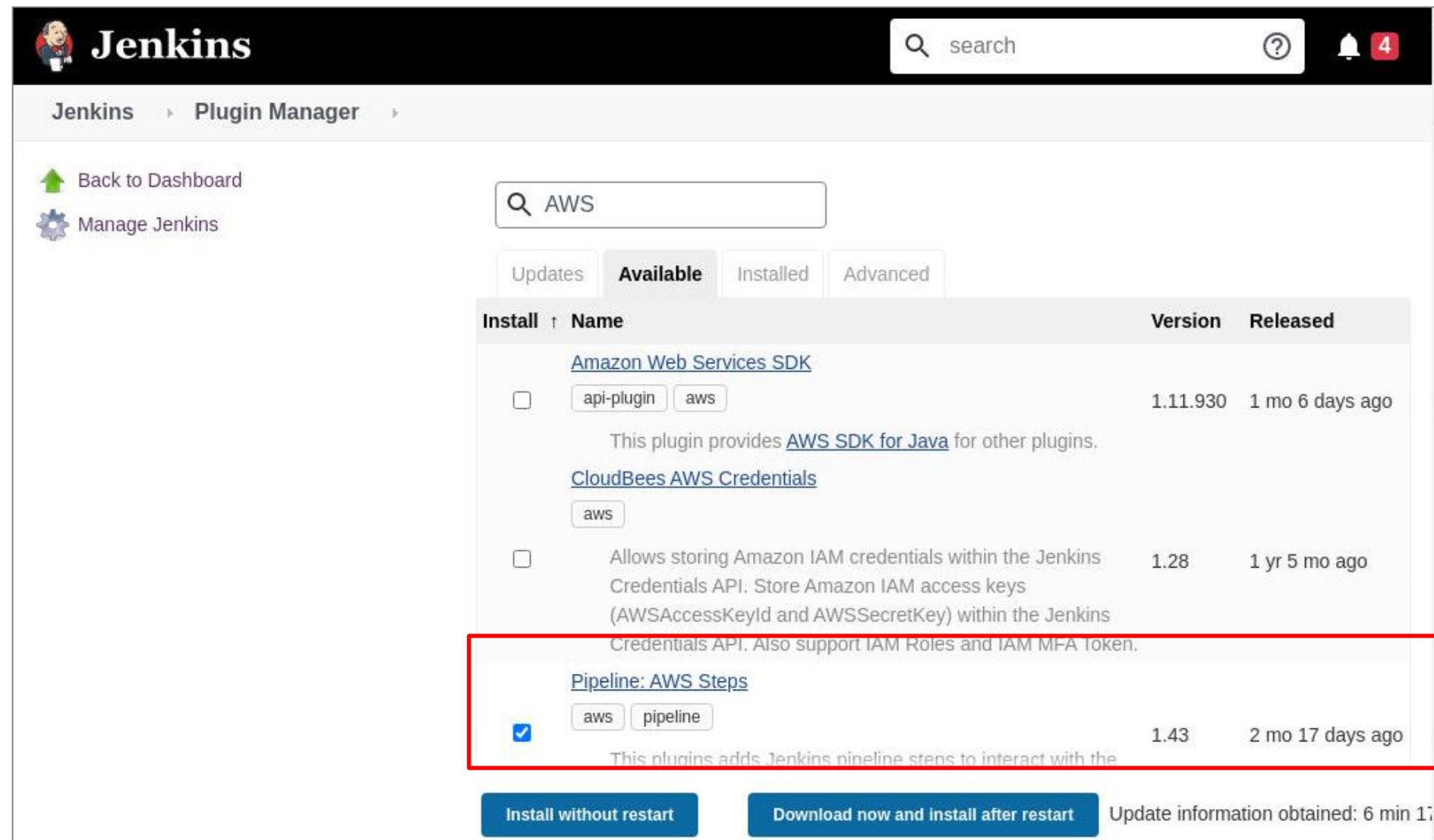


Hot Deploy

Hot deployment is the process of adding new components (such as WAR files, EJB Jar files, enterprise Java beans, servlets, and JSP files) to a running server without having to stop the application server process and start it again.

AWS Pipeline

AWS pipeline plugin adds Jenkins pipeline steps to interact with the AWS API.



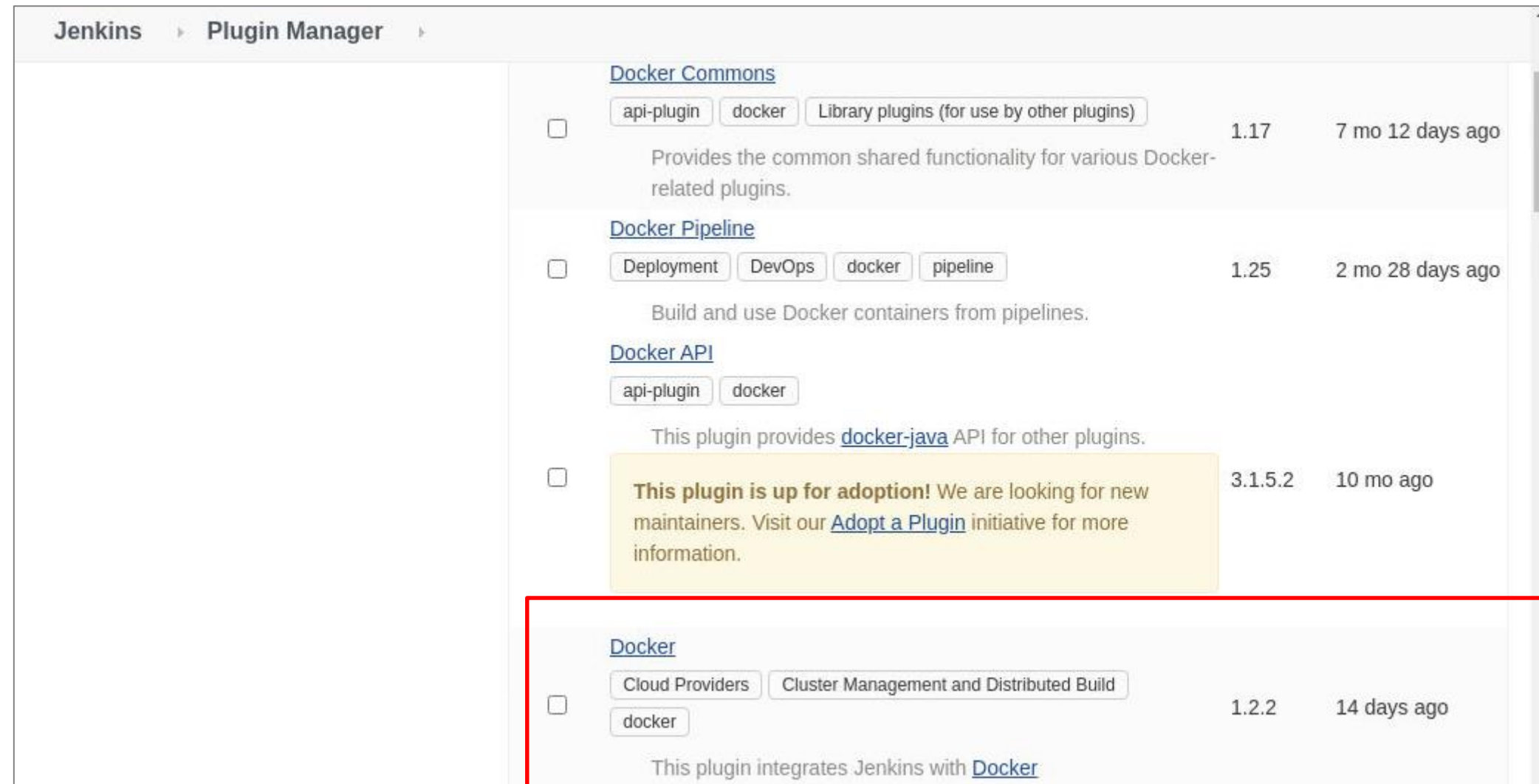
The screenshot shows the Jenkins Plugin Manager interface. The 'Available' tab is selected, and a search for 'AWS' has been performed. The following table lists the available plugins:

Install	Name	Version	Released
<input type="checkbox"/>	Amazon Web Services SDK api-plugin aws This plugin provides AWS SDK for Java for other plugins.	1.11.930	1 mo 6 days ago
<input type="checkbox"/>	CloudBees AWS Credentials aws Allows storing Amazon IAM credentials within the Jenkins Credentials API. Store Amazon IAM access keys (AWSAccessKeyId and AWSSecretKey) within the Jenkins Credentials API. Also support IAM Roles and IAM MFA Token.	1.28	1 yr 5 mo ago
<input checked="" type="checkbox"/>	Pipeline: AWS Steps aws pipeline This plugin adds Jenkins pipeline steps to interact with the	1.43	2 mo 17 days ago

At the bottom, there are two buttons: 'Install without restart' and 'Download now and install after restart'. The update information is 'Update information obtained: 6 min 17 s'.

Jenkins and Docker

Docker plugin allows to use a docker host dynamically to provision build agents, run single build, tear-down agent. This plugin integrates Jenkins with Docker.



Assisted Practice

Tomcat and Jenkins

Problem statement: You have been asked to connect and deploy a Java application on Tomcat using Jenkins.

Steps to perform:

1. Push code to GitHub repositories
2. Configure Tomcat
3. Log in to Jenkins
4. Create Jenkins job for deployment
5. Run the job and deploy the app

Key Takeaways

- Continuous Deployment is a development practice of releasing software on production servers continuously in an automated manner.
- Continuous delivery is a process, where code changes are automatically built, tested, and prepared for production.
- Smoke testing is a software testing method that determines whether the employed build is stable or not.



Lesson-End Project: Deploying Maven App to Tomcat Server

Problem Statement:

You're a DevOps engineer at Hooli, a computer vision company that started out as an image search platform. The company is celebrating its 25th anniversary and the CEO wants to celebrate the occasion by replacing the Hooli website with the original website that the company started out with for a day. You're required to build a Maven web app that says "Welcome to Hooli", set up a Tomcat server on an EC2 instance, set up a Jenkins pipeline to compile the app, and deploy it to Tomcat. The Jenkins job has to compile the app, generate a war file, and deploy the war file to the Tomcat server.

Requirements:

- The app should be built with Maven.
- The Tomcat server should allow remote deployment.
- The pipeline should be built with the Maven Project plugin.
- The build job should compile the app and do the deployment as a post-build action.

