

# CONTINUOUS INTEGRATION (CI)



Learning Objectives:

By the end of this discussion, students should be able to:

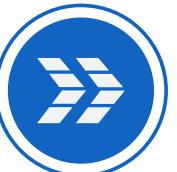
- Understand the concept of Continuous Integration (CI) in software engineering.
- Explain the benefits of CI in modern software development.
- Describe best practices for implementing CI.
- Identify common challenges and how to address them.

# Definition:

- **Continuous Integration (CI)** is a software development practice in which developers frequently integrate their code into a shared repository, ideally multiple times a day. Each integration is automatically verified by running tests to detect errors as early as possible.
- enables faster feedback loops, better collaboration, and higher software quality.



# KEY COMPONENTS

-  **Version Control System (VCS)**
-  **Automated Build System**
-  **Automated Testing**
-  **CI Server**
-  **Feedback Mechanism**



# Version Control System (vcs)

A Version Control System (VCS) manages and tracks changes to the source code. It allows multiple developers to collaborate, merge changes, and revert to previous versions when needed.



# Version Control System (vcs)

Examples:

- Git – A widely used distributed VCS that tracks changes and allows developers to work independently before merging changes.
- GitHub, GitLab, Bitbucket – Platforms that host Git repositories and provide collaboration feature



# Version Control System (vcs)

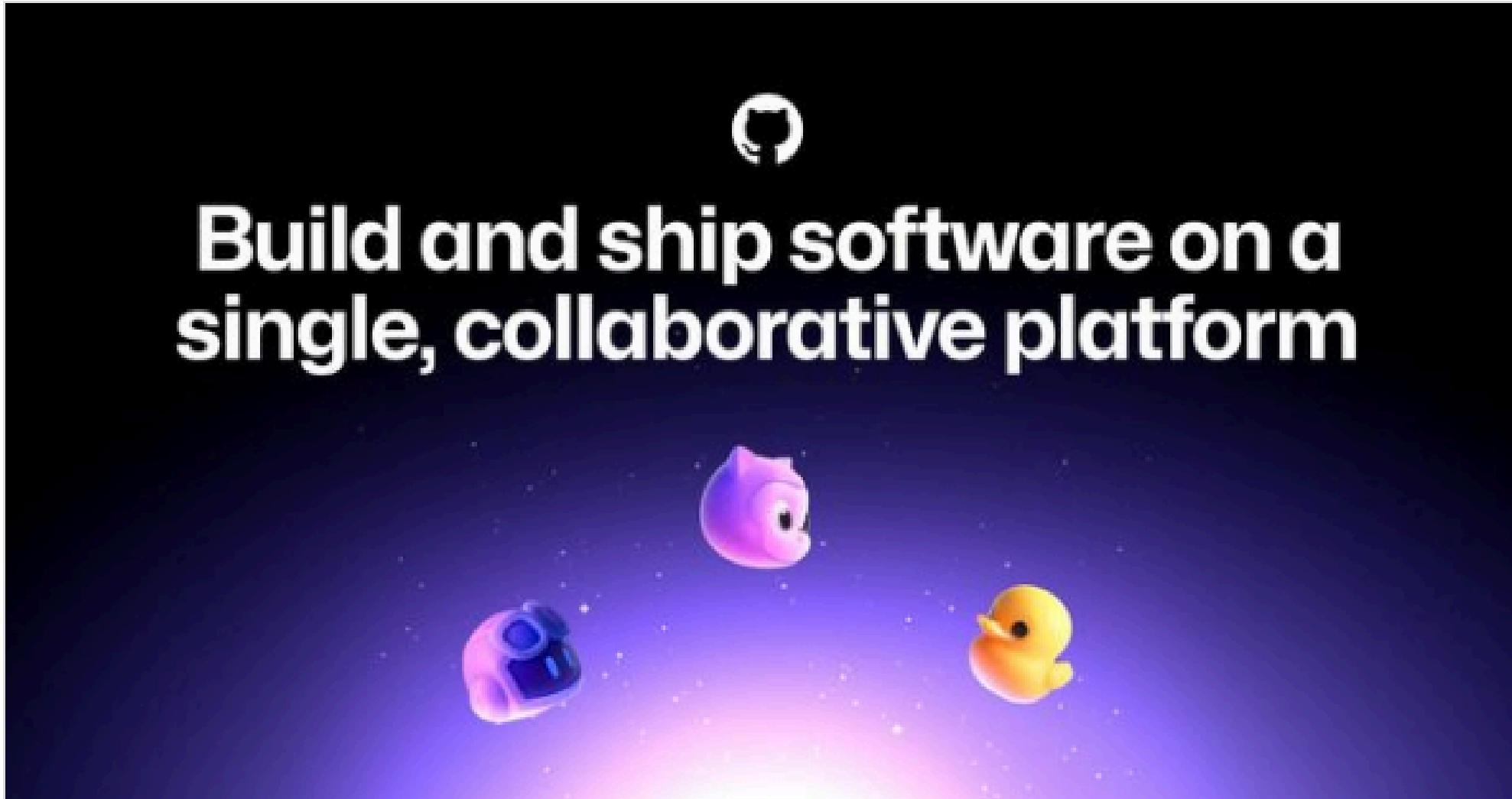
Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It has features like cheap local branching, convenient staging areas, and multiple workflows.

<https://git-scm.com>



# Version Control System (VCS)



**GitHub · Build and ship software on a single, collaborative platform**

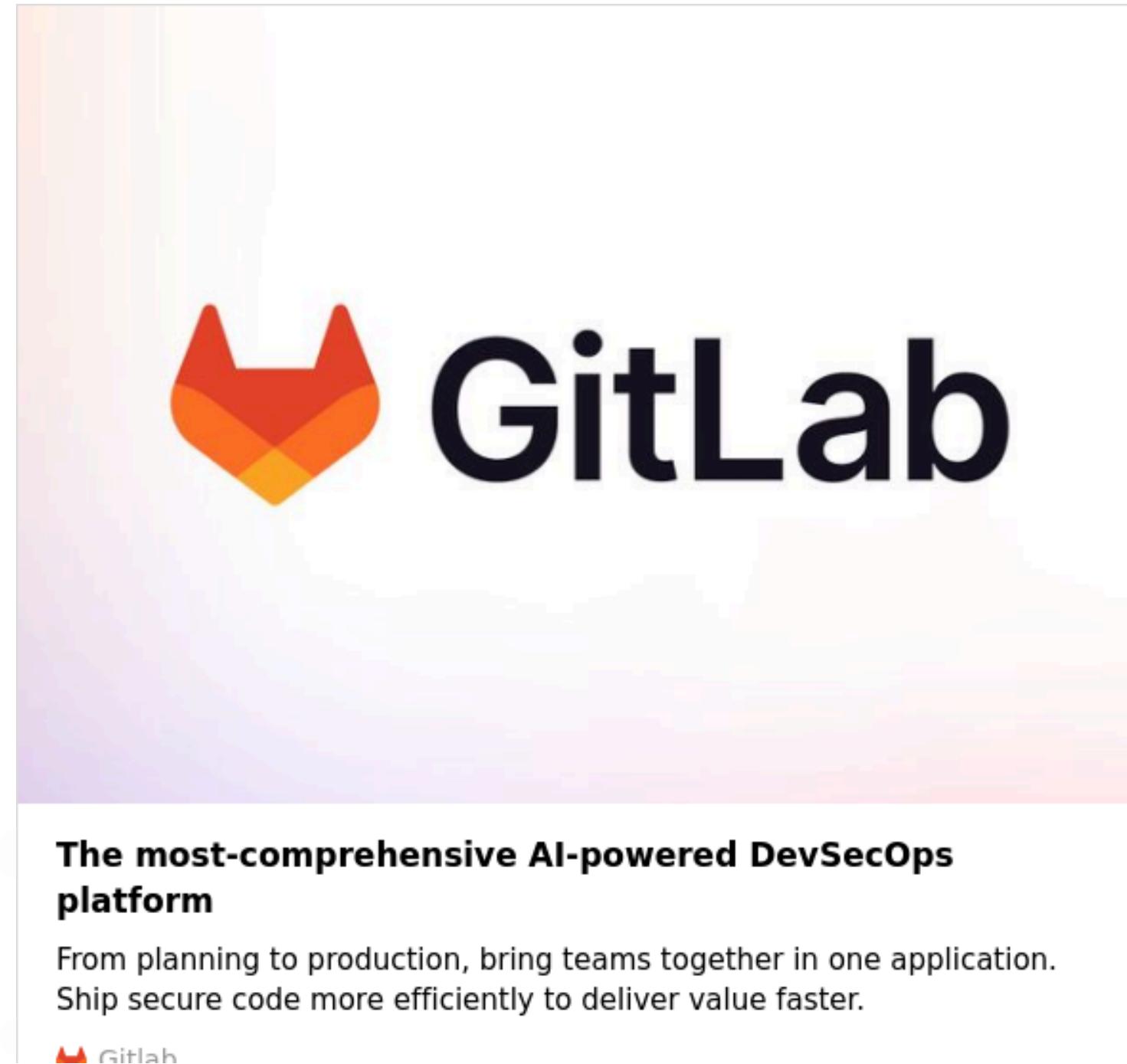
Join the world's most widely adopted, AI-powered developer platform where millions of developers, businesses, and the largest open source community build software that advances humanity.



<https://github.com>



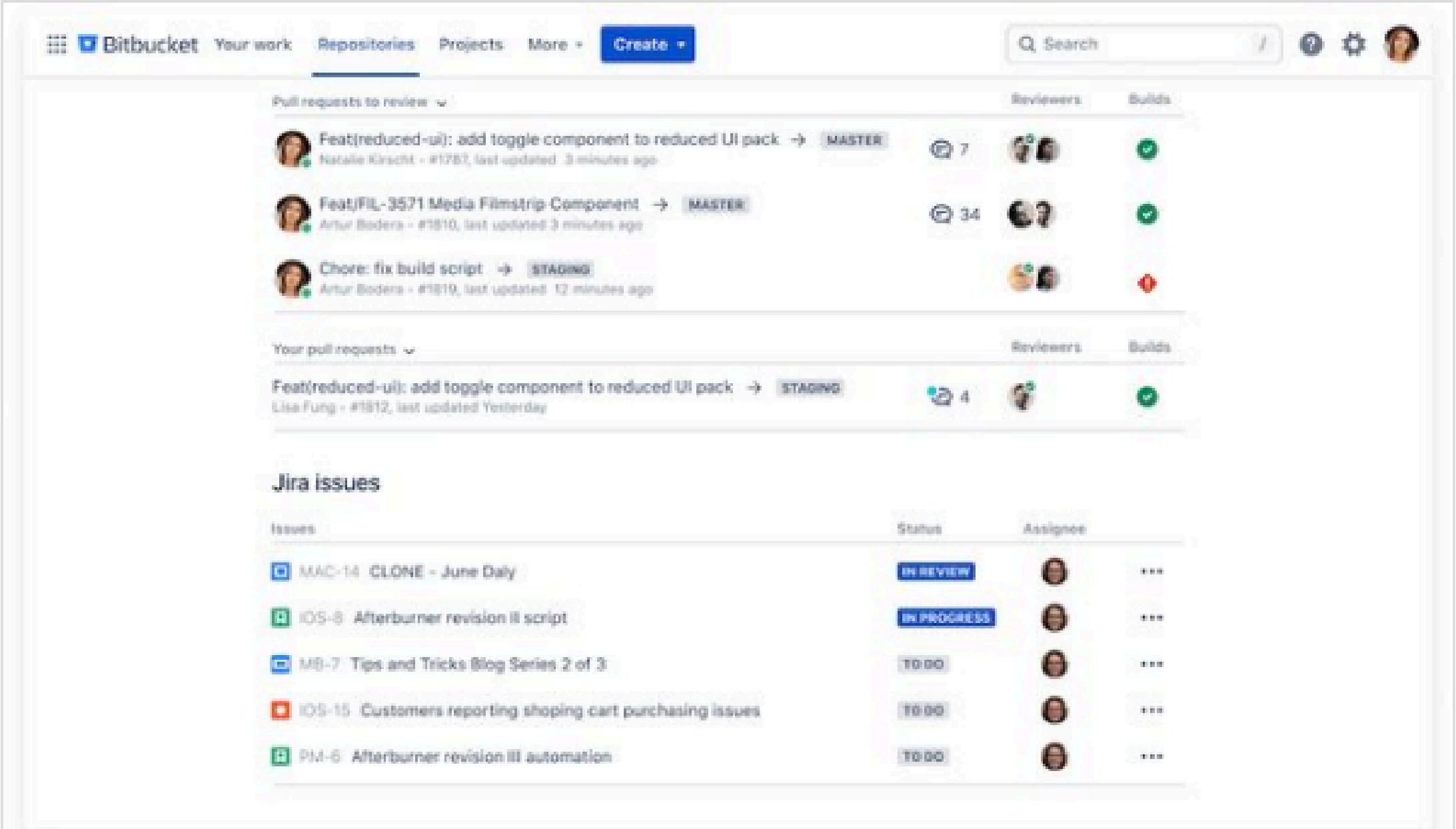
# Version Control System (VCS)



<https://about.gitlab.com>



# Version Control System (VCS)



The screenshot shows the Bitbucket Cloud interface. At the top, there's a navigation bar with 'Bitbucket', 'Your work', 'Repositories', 'Projects', 'More', 'Create', and a search bar. Below the navigation, there are two sections: 'Pull requests to review' and 'Your pull requests'. The 'Pull requests to review' section lists three pull requests: 'Feat(reduced-ui): add toggle component to reduced UI pack' (status: MASTER, 7 reviews, green build), 'Feat/FIL-3571 Media Filmstrip Component' (status: MASTER, 34 reviews, green build), and 'Chore: fix build script' (status: STAGING, 0 reviews, red build). The 'Your pull requests' section shows one pull request: 'Feat(reduced-ui): add toggle component to reduced UI pack' (status: STAGING, 4 reviews, green build). At the bottom, there's a 'Jira issues' section with a table:

Issue	Status	Assignee
MAC-14 CLONE - June Daily	IN REVIEW	[User]
iOS-8 Afterburner revision II script	IN PROGRESS	[User]
MO-7 Tips and Tricks Blog Series 2 of 3	TO DO	[User]
iOS-15 Customers reporting shopping cart purchasing issues	TO DO	[User]
PW-6 Afterburner revision III automation	TO DO	[User]

<https://bitbucket.org/product>

## Git solution for teams using Jira

Bitbucket Cloud is a Git-based code and CI/CD tool optimized for teams using Jira.

# Version Control System (VCS)

Example in Action:

- A development team uses Git with GitHub to store and manage their project.
- Developers create branches (e.g., feature-login for login functionality).
- They push their changes to GitHub and open a Pull Request (PR) for review.



# Automated Build System

An automated build system compiles the source code and ensures that all dependencies are properly included. It is responsible for checking whether the latest changes break the application.



# Automated Build System



## Gradle Build Tool

Accelerate developer productivity. Gradle helps teams build, automate and deliver better software, faster.

 Gradle

<https://gradle.org/>  
(Java-based projects)

## Gradle Build Tool accelerates developer productivity

Gradle is the open source build system of choice for Java, Android, and Kotlin developers. From mobile apps to microservices, from small startups to big enterprises, it helps teams deliver better software, faster.



# Automated Build System



Apache / Maven / Welcome to Apache Maven

Download | Get Sources | Last Published: 2025-03-28

## Welcome to Apache Maven

Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.

If you think that Maven could help your project, you can find out more information in the "About Maven" section of the navigation. This includes an in-depth description of [what Maven is](#) and a [list of some of its main features](#).

This site is separated into the following sections, depending on how you'd like to use Maven:

Use	Download, Install, Configure, Run Maven	Maven Plugins and Maven Extensions
Extend	<a href="#">Write Maven Plugins</a>	<a href="#">Improve the Maven Central Repository</a>
Contribute	<a href="#">Help Maven</a>	<a href="#">Develop Maven</a>

Information for those needing to build a project that uses Maven  
Information for developers writing Maven plugins.  
Information if you'd like to get involved. Maven is an open source community and welcomes contributions.

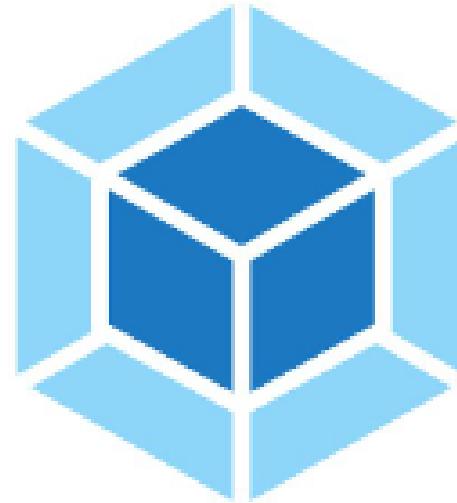
Lists of plugins and extensions to help with your builds.  
Information for those who may or may not use Maven, but are interested in getting project metadata into the [central repository](#).  
Information for those who are currently Maven developers, or who are interested in contributing to the Maven project itself.

Each guide is divided into a number of trails to get you started on a particular topic, and includes a reference area and a "cookbook" of common examples.



<https://maven.apache.org>  
(Java, Spring Boot applications)

# Automated Build System



## webpack

webpack is a module bundler. Its main purpose is to bundle JavaScript files for usage in a browser, yet it is also capable of transforming, bundling, or...

webpack

<https://webpack.js.org/>  
(JavaScript applications)



# Automated Build System

## Example in Action:

- A developer commits a new feature.
- The CI pipeline triggers a build using Gradle.
- If there's a missing dependency or syntax error, the system fails the build and notifies the developer.



# Automated Testing

Automated testing runs various types of tests to identify errors early, before deployment. This includes:

- Unit Tests – Check individual components.
- Integration Tests – Validate interactions between modules.
- Functional Tests – Ensure the application works as expected.



# Automated Testing



JUnit 5



JUnit 4

The 5th major version of the programmer-friendly testing framework for Java and the JVM

User Guide

Javadoc

Code & Issues

Q & A

Support JUnit

<https://junit.org/junit5/>

Unit testing framework.



# Automated Testing

## Selenium

Selenium automates browsers. That's it!



Selenium

<https://www.selenium.dev/>

Automated browser testing.



# Automated Testing

pytest: helps you write better programs  
The pytest framework makes it easy to write small, readable tests, and can scale to support complex functional testing for applications and libraries. pytest requires: Python 3.8+ or PyPy3.

Framework for writing unit and integration tests.



# Automated Testing

## Example in Action:

- A developer updates a function in a banking application.
- Automated tests run to verify that the function does not break other features.
- If a test fails, the developer is notified, preventing bugs from reaching production.



# CI Server

A CI server automates the entire CI process by:

- Detecting changes in the repository.
- Running builds and tests automatically.
- Providing feedback to developers.



# CI Server

Examples:

- Jenkins – Open-source automation server.
- GitHub Actions – Built-in CI/CD for GitHub.
- Travis CI – Cloud-based CI/CD tool.
- CircleCI – Fast, scalable CI/CD tool.



# CI Server

## Example in Action:

- A developer pushes code to GitHub.
- GitHub Actions detects the change and triggers a build and test.
- The CI server provides immediate feedback if something breaks.



# Feedback Mechanism

A feedback mechanism provides real-time feedback to developers if a build fails or tests break. It helps in:

- Identifying issues quickly.
- Reducing debugging time.
- Keeping the project stable.



# Feedback Mechanism

Examples:

- Email notifications – Jenkins sends an email if a build fails.
- Slack integration – GitHub Actions posts updates in a team's Slack channel.
- Dashboards – CircleCI provides real-time build status in a web dashboard.



# Feedback Mechanism

Example in Action:

- A developer commits code.
- Jenkins runs the build and detects an issue.
- Jenkins sends an email alert to the developer with logs explaining the failure.



# Summary Table of CI Components

CI Component	Purpose	Examples	Example in Action
<b>Version Control System (VCS)</b>	Tracks and manages code changes	Git, GitHub, GitLab, Bitbucket	Developers push changes to GitHub and open a pull request for review.
<b>Automated Build System</b>	Ensures code compiles successfully	Gradle, Maven, Webpack	CI pipeline triggers a build after a developer pushes code.
<b>Automated Testing</b>	Runs unit, integration, and functional tests	JUnit, Selenium, PyTest	A new feature is added, and automated tests verify that it doesn't break other features.
<b>CI Server</b>	Automates builds and tests	Jenkins, GitHub Actions, Travis CI, CircleCI	GitHub Actions detects code changes and runs automated tests.
<b>Feedback Mechanism</b>	Provides real-time feedback to developers	Email alerts, Slack notifications, Dashboards	Jenkins emails developers when a build fails, showing the error logs.

- CI is an essential practice in modern software development.
- It allows teams to integrate code frequently, detect errors early, and deliver stable software faster.
- By using version control, automated builds, testing, CI servers, and feedback mechanisms, teams can maintain high-quality code and improve development efficiency.

