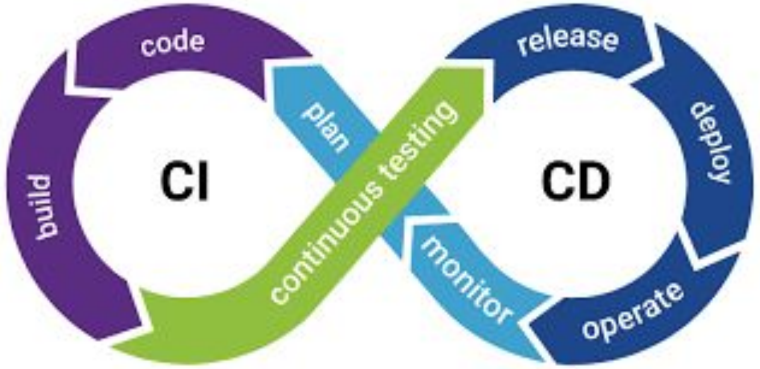
# CI / CD



CONTINUOUS INTEGRATION CONTINUOUS DEPLOYMENT

## What is CI/CD means?

**Continuous integration**

is the practice of automating the integration of code changes from multiple contributors into a single software project. it’s allowing developers to frequently merge code changes into a central repository where builds and tests then run. Automated tools are used to assert the new code’s correctness before integration.

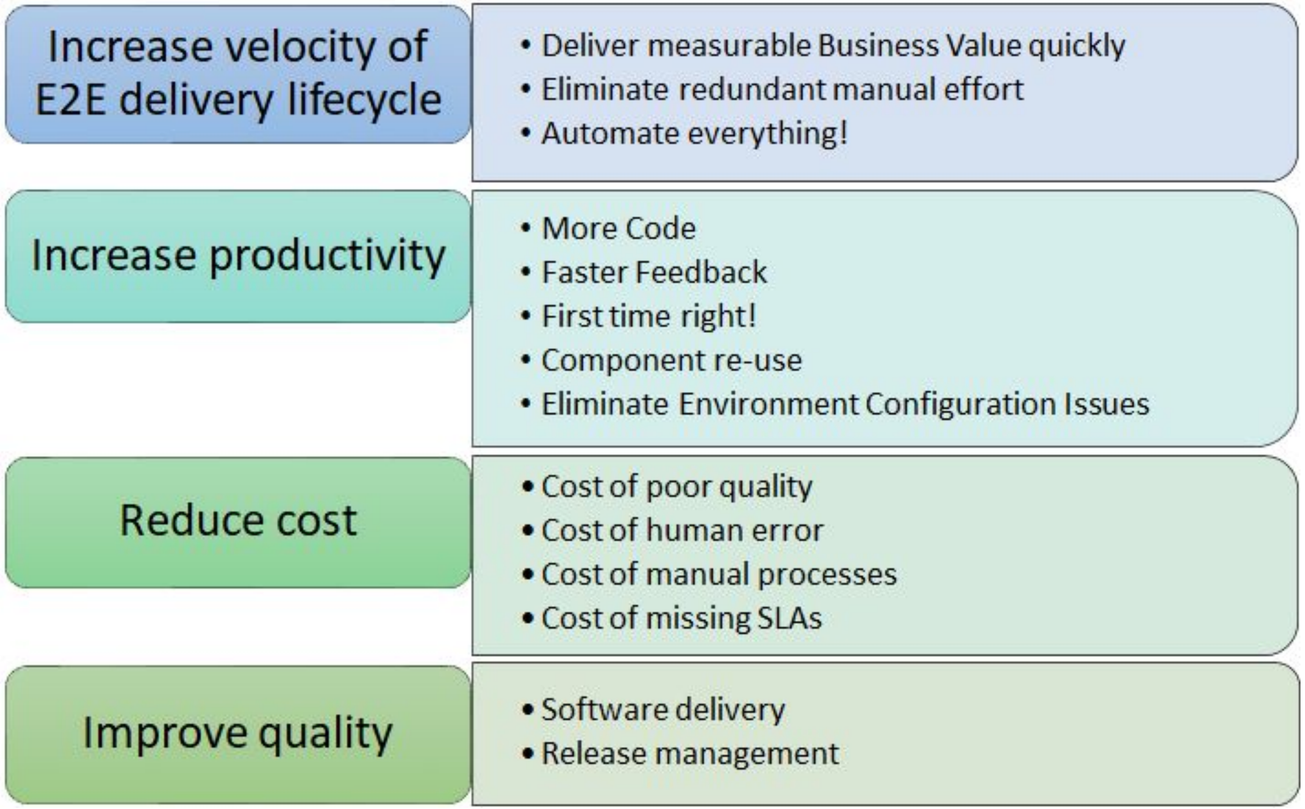
**Continuous delivery**

is a software development practice where code changes are automatically prepared for a release to production. A pillar of modern application development, continuous delivery expands upon integration  integration by deploying all code changes to a testing environment and/or a production environment after the build stage. When properly implemented, developers will always have a deployment-ready build artifact that has passed through a standardized test process.

**Continuous Deployment**

It is another step beyond Continuous Integration, similar to Continuous Delivery. The difference is that instead of deploying your application manually, you set it to be deployed automatically. Human intervention is not required.

# DevOps project Objective (CICD)



| | |
|---|---|
| Increase velocity of E2E delivery lifecycle | • Deliver measurable Business Value quickly<br>• Eliminate redundant manual effort<br>• Automate everything! |
| Increase productivity | • More Code<br>• Faster Feedback<br>• First time right!<br>• Component re-use<br>• Eliminate Environment Configuration Issues |
| Reduce cost | • Cost of poor quality<br>• Cost of human error<br>• Cost of manual processes<br>• Cost of missing SLAs |
| Improve quality | • Software delivery<br>• Release management |

# What is DevOps toolchain?

**Plan.** This phase helps define business value and requirements. Sample tools include Jira or Git to help track known issues and perform project management.

**Code.** This phase involves software design and the creation of software code. Sample tools include GitHub, GitLab, Bitbucket, or Stash.

**Build.** In this phase, you manage software builds and versions, and use automated tools to help compile and package code for future release to production. You use source code repositories or package repositories that also “package” infrastructure needed for product release. Sample tools include Docker, Ansible, Puppet, Chef, Gradle, Maven, or JFrog Artifactory.

**Test.** This phase involves continuous testing (manual or automated) to ensure optimal code quality. Sample tools include JUnit, Codeception, Selenium, Vagrant, TestNG, or BlazeMeter.

**Deploy.** This phase can include tools that help manage, coordinate, schedule, and automate product releases into production. Sample tools include Puppet, Chef, Ansible, Jenkins, Kubernetes, OpenShift, OpenStack, Docker, or Jira.

**Operate.** This phase manages software during production. Sample tools include Ansible, Puppet, PowerShell, Chef, Salt, or Otter.

**Monitor.** This phase involves identifying and collecting information about issues from a specific software release in production. Sample tools include New Relic, Datadog, Grafana, Wireshark, Splunk, Nagios, or Slack.

## What is Benefits of DevOps?

**Faster, better product delivery**

**Faster issue resolution and reduced complexity**

**Greater scalability and availability**

**More stable operating environments**

**Better resource utilization**

**Greater automation**

**Greater visibility into system outcomes**

**Greater innovation**