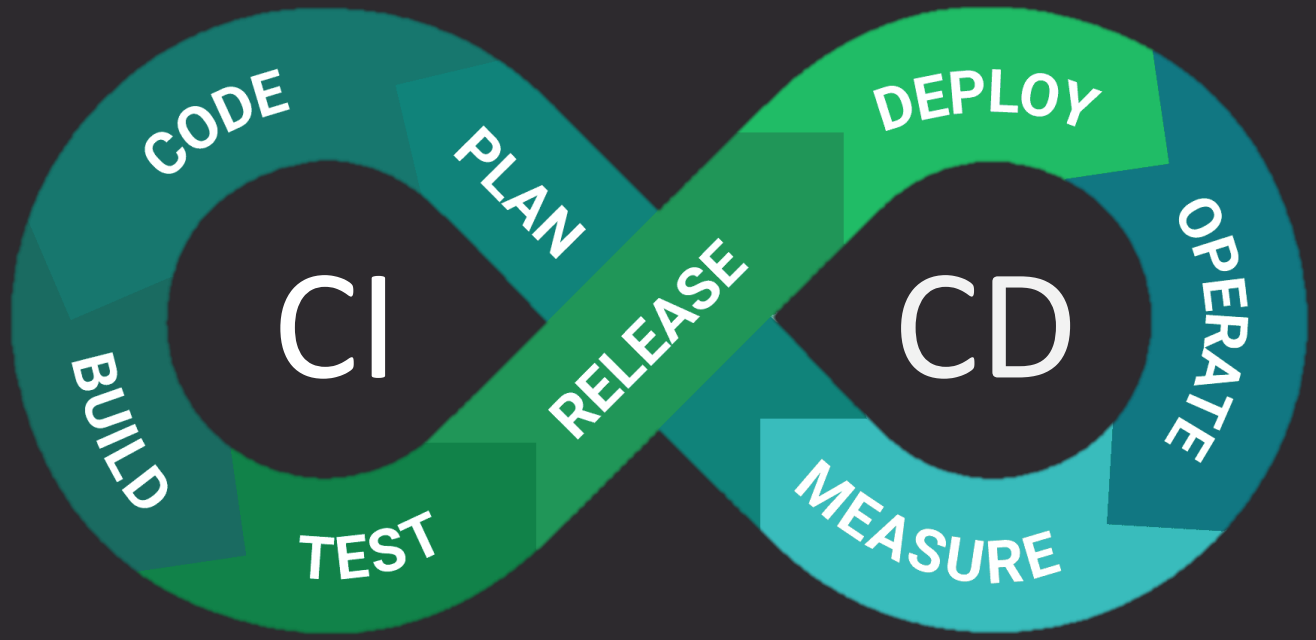

CODE
PLAN
BUILD
TEST
RELEASE
DEPLOY
OPERATE
MEASURE
CI
CD

What Is CI/CD?

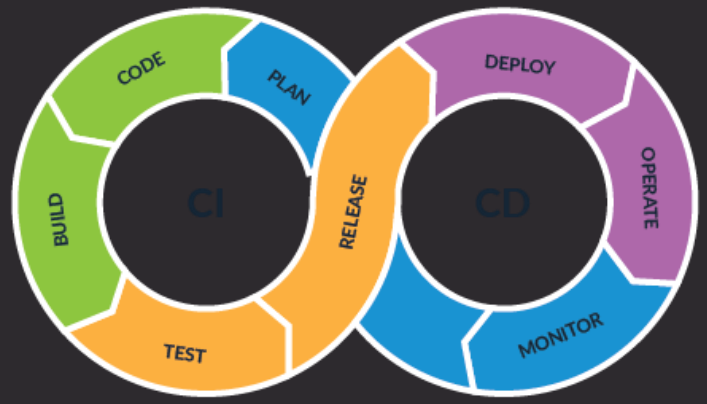
CI/CD is a shortened term for Continuous Integration and Continuous Delivery. CI/CD is the combination of principles, practices, and capabilities that allow for software changes of all kinds to get users in a quick, repeatable, and safe manner. This allows for software developers to integrate their feature or service changes continuously and for IT and operations teams to deliver with the standards, security, and confidence businesses need.

Continuous Integration

A major prerequisite to CI/CD is Continuous Integration (CI). CI is an automated process for continually integrating software development changes. CI processes automate the building, testing, and validation of source code. By working with CI capabilities, developers can accelerate their code release cycles, making it less likely to run into long feature development cycles and the challenges that come with it, such as merge conflicts.

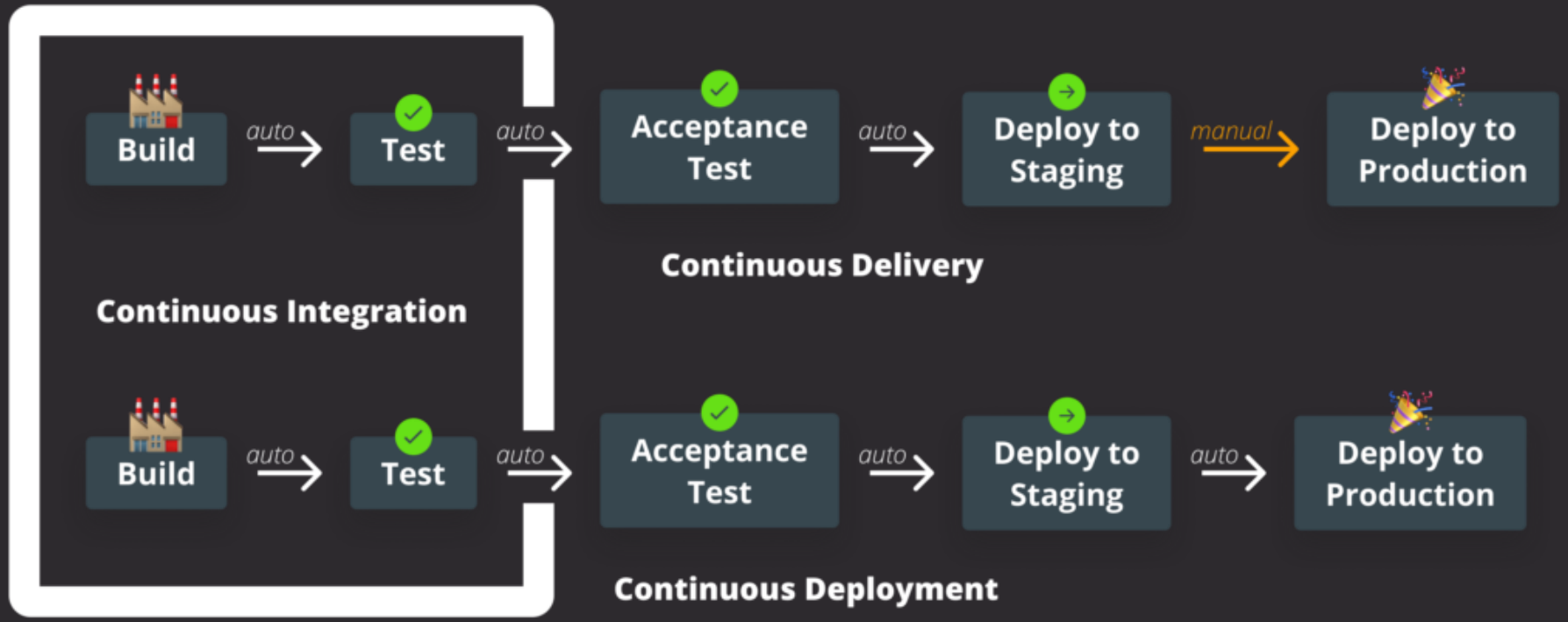
Continuous Delivery

The goal of continuous delivery is to deliver a packaged artifact into a production environment. CD automates the entire delivery process, including the deployment process. CD responsibilities can include provisioning infrastructure, managing changes (ticketing), deploying artifacts, verifying and monitoring those changes, and ensuring these changes do not happen if there are any issues.



How CI and CD Work Together

CI and CD work together to ensure we have the capabilities and visibility to catch and fix defects and potential production failures or incidents that can be otherwise detrimental to a business



Benefits of CICD

01 Automate Infrastructure Creation and clean up:
Eliminating human errors and avoid unnecessary cost of unused or invalid infrastructure

02 Faster to production:
By automating the pipeline to production this way, we can deploy features as soon as created which will help increase revenue

03 Automated Rollback Triggered by Job Failure:
Automate the process of rolling back and cleaning any infrastructure left which would help in reducing cost and lower down time
Benefits of CICD

04 Catch Compile Errors After Merge:
Discover errors as soon as the developer make his commit which will help reduce the time of developers and reduce cost

05 Catch Unit Test Failures:
Unit tests are not neglected with CICD which will increase code quality and catch errors early before production which would decrease cost

06 Automated Smoke Tests:
Automate smoke test after deployment and automatic rollback in case of failure which will decrease downtime and reduce cos