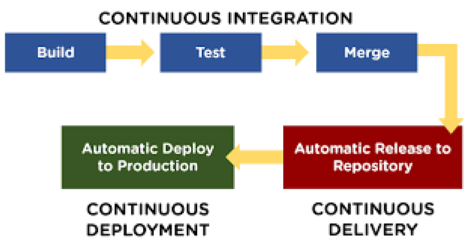
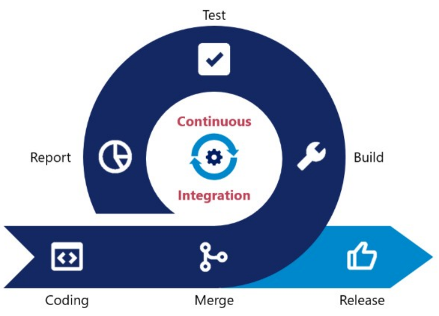
INTRODUCTION TO CONTINUOUS INTEGRATION AND DEPLOYMENT

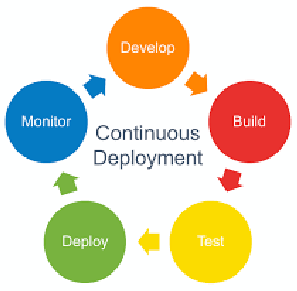
Continuous integration and continuous delivery (CI/CD) are modern ways of implementing software. They are often confused with each other. Continuous integration (CI) is a procedure in which software developers integrate their code at orderly intervals. Continuous delivery (CD) involves having the capability to deploy an application at any moment, without manual interferance. CI/CD refers to specific development practices that promote increased productivity, quicker software output and better quality. CI/CD can help change and transform an organization, but there is a lot to conisder.



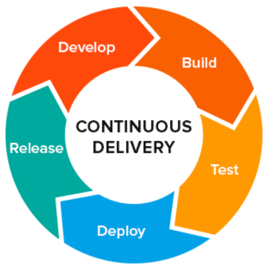
Continuous integration (CI) focuses on the early part of a software development cycle where the code is built and goes through initial testing. The Number of the build can be daily or you can have multiple builds per day. These minute, re-current builds allow easy and low-risk experimentation, inclduing the ability to easily push for a roll back or discard unwanted results. In time, CI is completed when a build successfully completes initial testing and is ready to move to more comprehensive testing.



Continuous deployment (also CD) goes through the same phases as continuous delivery. The Only difference is that continuous deployment automatically deploys each validated build to production. By comparison, continuous delivery typically just stages the validated build for manual deployment or other human authorization.



Continuous delivery is a process of developing an application to aid software developers rollout software efficently and quicker. The main goal of Continuous Delivery is to ehance the quick and reliable rollout of new features, updates, and fixes to your users. This can be achieved by automating the entire software development process from Beginning to end. The process involves automating the entire software delivery pipeline so that changes are made to the source code in an ordely manner and repeatable way. This allows for swifter and more reliable rollout, which then leads to increased customer satisfaction and increased revenue for the business.



BENEFITS OF CI/CD:

Fast delivery to market: The focus of a CI/CD pipeline is to deploy working application to users faster and regularly. Understanding your users' wants, coming up with innovative features, and turning them into a working application is not necessarily enough if your competition is moving more quickly. With an automated CI/CD pipeline you can make changes daily, hourly, weekly and monthly. New features and requirements can be launched quicker, with deployment strategies giving you the option to experiment and collect useful feedback, which you can then factor into the next build. Being able to rollout changes faster and with confidence means you can respond to new trends and new changes in the business requirement, which can put you ahead of the competition.

More efficient software development: Little iterations allow for less difficult and more efficient testing. The limited scope of code in each new iteration, as well as the scope to test it, makes it less difficult to find and fix bugs. Features are more readily evaluated for usefulness and user acceptance, and less useful features are easily adjusted or even discarded before further development is wasted.

Improved software maintenance: Bugs can take days, weeks or even months to debug in traditional application development, but the regular flow of a CI/CD pipeline makes it less difficult to address and fix bugs on time and with better confidence. The product is more stable and reliable over time, which increases turn over for the business.

Improved operations support: Regular software releases keep operations staff in the know of software's requirements and monitoring requirements. Devops Engineers are better able to deploy software upgrades, new features and handle rollbacks with fewer deployment errors and troubleshooting. Similarly, IT automation technologies can make deployments faster while reducing setup or configuration issues.

Improved project collaboration and quality: Teams can identify software quality issues quicker with smaller code packages, instead of bigger ones created later along project timelines. Also, when developers have shorter commit cycles, they probably won't edit the same code and need merges.

Conclusion:

Benefits of CI/CD pipeline range from quality of code and quick debugging and fixes, ensuring you're building based on business and user requirement, to improving your entire software development process as buiness requirement changes and evolve, making the business more competitve in the market.

Building a CI/CD pipeline provides an opportunity for collaboration across a whole range of functions. By reducing the steps to rollout your product, you can provide your team with more know how on how your product is used and free up individuals' time so they can focus on other things.