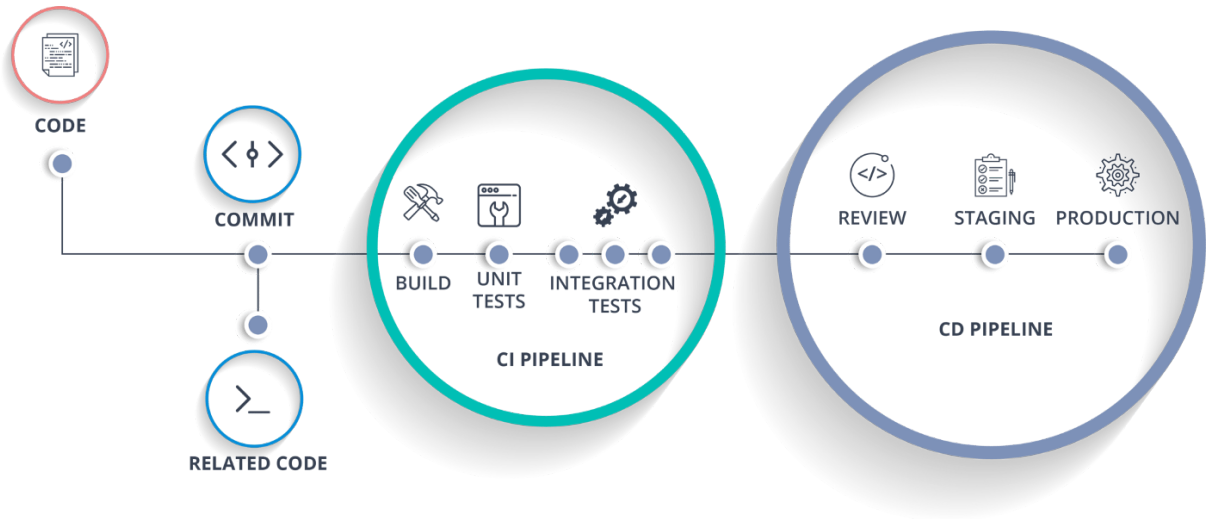
CI/CD for Non-Technical Stakeholder

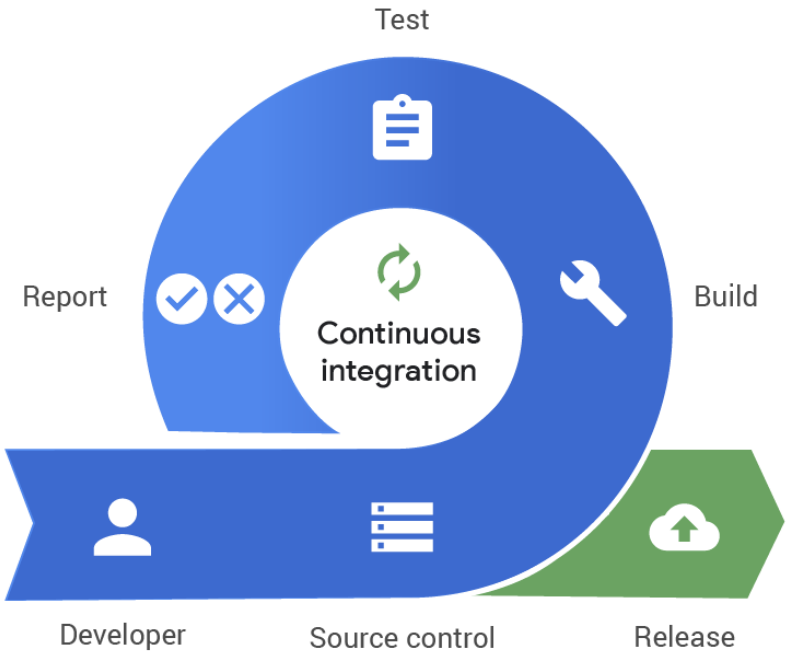


CI/CD Introduction
Before the CI/CD idea was birthed, development practices were majorly manual processes, this style of building and developing applications and software were accompanied with long months or even years of intense development practices before an application or a software was declared ready for production or release. This long development time has been known to have myriads of challenges, which include: cost of development (man power and finance required), manual code reviews on every commit before codes are merged, manual deployments, long time to markets, and in many cases, errors or failures are not caught on time then the process is also accompanied with inconsistency.

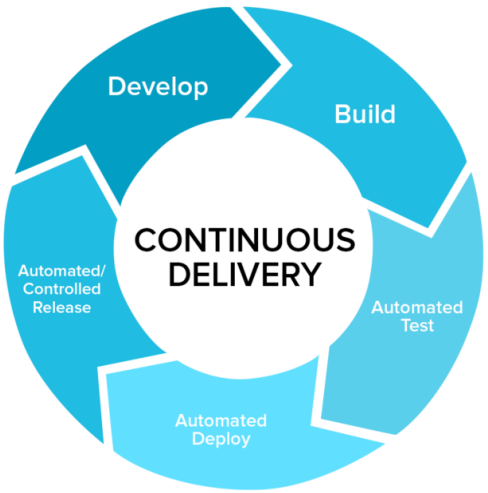
Continuous Integration and Continuous Delivery (CI/CD)
This is a development strategy whereby application development processes are automated as much as possible to simplify and reduce the whole software development life cycle and time to market of an application or a software. This enables developers to focus more on his code and build applications faster, with zero errors or few errors that are captured fast, this also helps the operations team to carry out all the required testing, analyze, build and deploy the highly quality codes.

Continuous Integration (CI)
Continuous integration is a software or application development pattern whereby developers commit and merge their working copies of codes into a central version control system several times a day. This phase consists of everything that has to do with code, which all adds up to produce a high-quality and deployable artifact.
Stages Here include:

- Code Compile
- Unit Test
- Static Code analysis
- Dependency, Vulnerability testing
- Artifact storage

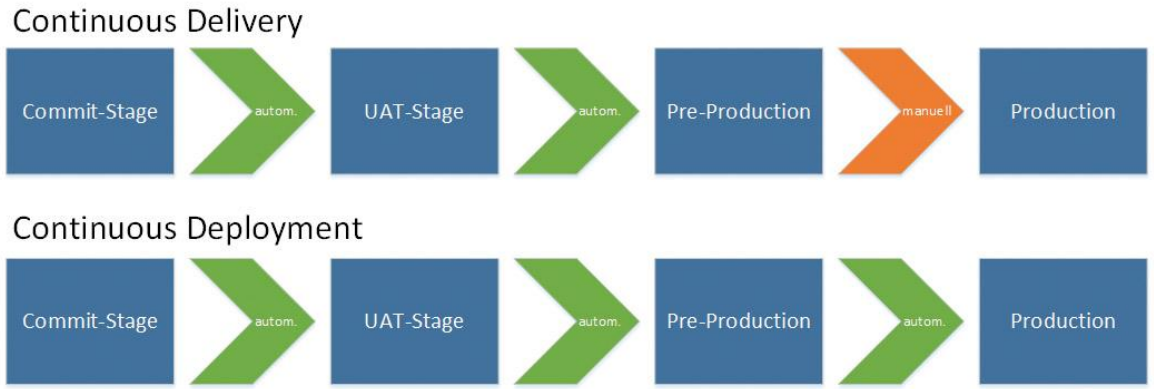


Continuous Delivery (CD)
Continuous Delivery continues from the CI stage, it is the practice of delivering updates into production and to end-users in a sustainable, safe and fast approach. Updates can include a new feature, bug fixes, configuration management, or experiments.
Continuous Delivery uses automation for all parts of its process but would require a manual third party to allow the change or release or accept the new features and updates.



Continuous Deployment
Continuous deployment on the other hand unlike continuous delivery does not require a manual check to release updates and changes to production. Everything is automated and the process kicks off after a successful Continuous integration stage. This helps to save time

and beats the human error factor in the development process when you have a trusted CI stage in place.



Benefits of CI/CD

- **Catch Compile Errors After Merge:** With quality CI in place compile errors can be detected on time, this helps to reduce developers' time on issues from new developers' code, and as a result, helps to reduce the cost of development.
- **Detect Unit Test Failures and Identify Security vulnerabilities:** this helps to prevent bugs from being introduced to the production environment, prevent costly security loopholes by checking code vulnerabilities, it also reduce testing time. This ultimately helps to avoid unwanted costs of debugging.
- **Automation of Infrastructure Creation and Cleanup:** with CD infrastructures are created in an automated fashion faster without human errors and can be destroyed too ask quickly, this helps to avoid the cost and reduce the cost of unused resources.
- **Faster and Frequent Production deployments:** new features and updates can be released faster and shipped to production with CD. This helps to increase revenue from the features and updates use.
- **Automated Smoke Test:** the smoke test is used as an extra test to check if the deployed product is working as expected. This serves as additional security to reduce downtime from deploy-related crashes or major bugs. This helps to protect revenue by keeping copies deployed till another working copy is available depending on the deployment strategy.
- **Rollbacks:** With CICD there is an automated rollback in place at each stage in the cycle. This quickly undoes changes and returns production to a working state. This also helps to protect revenue.

Conclusion

Continuous integration and continuous Delivery or Deployment is an excellent way to ship applications and software faster and securely, it also helps to take pressure or reduce pressure from your development team. With good CICD in place, you can start shipping applications on a daily basis.