• Continuous Integration & Continuous Delivery

Continuous Integration (CI) allows you to continuously integrate code into a single shared and easy to access repository. Continuous Delivery (CD) allows you to take the code stored in the repository and continuously deliver it to production. CI/CD creates a fast and effective process of getting your product to market before your competition as well as releasing new features and bug fixes to keep your current customers happy.

Benefits of Continuous Integration and Continuous Delivery

- Smaller Code Changes
- Fault Isolations
- Faster Mean Time To Resolution (MTTR)
- More Test Reliability
- Faster Release Rate
- Reduce Costs

- Smaller Code Changes :
- One technical advantage of continuous integration and continuous delivery is that it allows you to integrate small pieces of code at one time. These code changes are simpler and easier to handle than huge chunks of code and as such, have fewer issues that may need to be repaired at a later date.
- Using continuous testing, these small pieces can be tested as soon as they are integrated into the code repository, allowing developers to recognize a problem before too much work is completed afterward. This works really well for large development teams who work remotely as well as those in-house as communication between team members can be challenging.

- **Fault Isolations**
- Fault isolation refers to the practice of designing systems such that when an error occurs, the negative outcomes are limited in scope. Limiting the scope of problems reduces the potential for damage and makes systems easier to maintain.
- Designing your system with CI/CD ensures that fault isolations are faster to detect and easier to implement. Fault isolations combine monitoring the system, identifying when the fault occurred, and triggering its location. Thus, the consequences of bugs appearing in the application are limited in scope. Sudden breakdowns and other critical issues can be prevented from occurring with the ability to isolate the problem before it can cause damage to the entire system.

- # Faster Mean Time To Resolution (MTTR)

  MTTR measures the maintainability of repairable features and sets the average time to repair a broken feature. Basically, it helps you track the amount of time spent to recover from a failure.

- CI/CD reduces the MTTR because the code changes are smaller and fault isolations are easier to detect. One of the most important business risk assurances is to keep failures to a minimum and quickly recover from any failures that do happen. Application monitoring tools are a great way to find and fix failures while also logging the problems to notice trends faster.

- # Faster Release Rate

- Failures are detected faster and as such, can be repaired faster, leading to increasing release rates. However, frequent releases are possible only if the code is developed in a continuously moving system.
- CI/CD continuously merges codes and continuously deploys them to production after thorough testing, keeping the code in a release-ready state. It's important to have as part of deployment a production environment set up that closely mimics that which end-users will ultimately be using. Containerization is a great method to test the code in a production environment to test only the area that will be affected by the release.

- More Test Reliability

  Using CI/CD, test reliability improves due to the bite-size and specific changes introduced to the system, allowing for more accurate positive and negative tests to be conducted. Test reliability within CI/CD can also be considered Continuous Reliability. With the continuous merging and releasing of new products and features, knowing that quality was top of mind throughout the entire process assures stakeholders their investment is worthwhile.

- **Reduce Costs**

- Automation in the CI/CD pipeline reduces the number of errors that can take place in the many repetitive steps of CI and CD. Doing so also frees up developer time that could be spent on product development as there aren't as many code changes to fix down the road if the error is caught quickly. Another thing to keep in mind: increasing code quality with automation also increases your ROI.