CI/CD Proposal

1. What is CI/CD?

* CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development. The main concepts attributed to CI/CD are continuous integration, continuous delivery, and continuous deployment. CI/CD is a solution to the problems integrating new code can cause for development and operations teams (AKA "integration hell").
* Specifically, CI/CD introduces ongoing automation and continuous monitoring throughout the lifecycle of apps, from integration and testing phases to delivery and deployment. Taken together, these connected practices are often referred to as a "CI/CD pipeline" and are supported by development and operations teams working together in an agile way with either a DevOps or site reliability engineering (SRE) approach.

1. Benefit

* 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.
* 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.
* Get feedback continuously
  + Timely feedback is the ultimate benefit of the CI/CD pipeline. The CI/CD pipeline incorporates automated testing across the software development phases to immediately trigger feedback when faults are identified. The feedback, along with user behavior data and the key performance metrics, helps glean insights into what works for your organization and how to make further improvements to the product.
* Simplify rollback
  + Easy rollback is one of the key benefits offered by CI/CD. A CI/CD pipeline empowers development teams to fail fast and recover even faster. Simply put, the CI/CD pipeline enables your developers to easily push code into production and, if any issues arise, simply roll it back. This ultimate ability to roll back code saves time, resources, and expenses by helping teams to fix the problem code at a faster pace.
* Enhance transparency and accountability
  + A robust CI/CD pipeline provides real-time transparency and visibility into the entire software development process, courtesy of continuous testing and feedback. The feedback reports and test results allow everyone in the team to track the project status and immediately understand the build failures, code integration problems, and architectural bottlenecks, among others. The project managers and stakeholders can easily check the project status and track accountability as needed.
* Improve performance metrics
  + Before DevOps and CI/CD, application performance or monitoring metrics are often absent, so teams failed to understand how the code is working in the real world. With CI/CD pipeline, the teams can easily monitor the health, performance, and reliability of the application. This metrics data help in gleaning actionable insights to improve the product further.
* Reduce backlog
  + A CI/CD pipeline gives the development team the time to work on the things they won't be able to do with traditional approaches. They can now go back to fix older code and improve its quality and make it more efficient. This not only enables your developers to tackle the backlog but also addresses the non-critical defects. In this way, the non-critical issues are fixed before they become critical and make it to production.

1. Stages when apply CI/CD:

|  |  |  |
| --- | --- | --- |
| Stage | Not Apply | Apply |
| Coding | Person | Person |
| Code Review | Person  Subjective | CI – Static Analysis |
| Compile/Lint | Person | CI |
| Merge/Intergrate | Person | CI |
| Run Unit Test | Person | CI |
| Verify Dependency | Person | CI |
| Deploy to Test Environment | Person | CD |
| Team Test | Person | CD |
| Create Infrastructure | Person | CD |
| Rollbacks | Person | CD |