CICD

CI/CD allows organizations to ship software quickly and efficiently

CONTINUOUS INTEGRATION

- continuous integration is the practice of merging all developers' working copies to a shared mainline several times a day. (Think of Code)
- The key goals of continuous integration are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates.
- Some common CI-related phases might include:
  - Compile
  - Unit Test
  - Static Analysis
  - Dependency vulnerability testing
  - Store artifact

CONTINUOUS DEPLOYMENT

- a software engineering approach in which software functionalities are delivered frequently and through automated deployments. (Think of Deployment)
- It allows you to deploy revisions to a production environment automatically without explicit approval from a developer.
- Some common CD-related phases might include:

  - Creating infrastructure
  - Provisioning servers
  - Copying files
  - Promoting to production
  - Smoke Testing
  - Rollbacks in case if any failure

CONTINUOUS DELIVERY

- It's a mindset more than a tool, It's an engineering paradigm in which teams produce and release value in short cycles.
- Principles of continuous delivery:
  - Repeatable reliable process
  - Automate everything
  - Version control everything
  - Bring the pain forward
  - Build in quality
  - Done means "Released"
  - everyone is responsible

BENEFITS OF CI/CD AT THE BUSINESS LEVEL

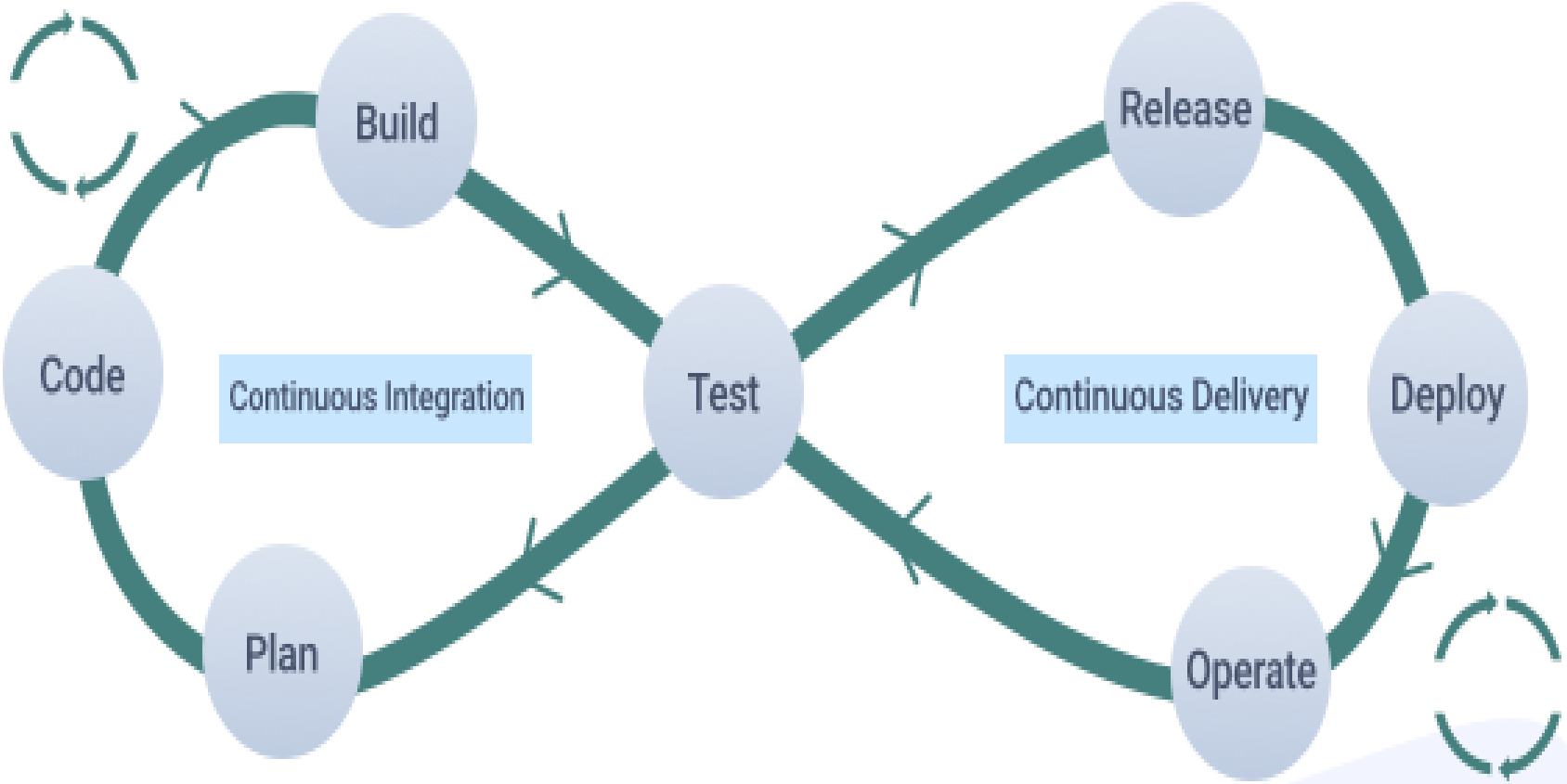
- **Deploy faster:** A CI/CD pipeline is like a turbo engine when it comes to accelerating the code deployment pace and time-to-market of the final product. the automation the CI/CD pipeline brings to the software development lifecycle reduces manual labor and time required for creating and maintaining deployment scripts and tools.
- **Avoid outages:** A successful CI/CD pipeline enables your DevOps teams to continuously integrate small batches of code instead of the entire application. So, you can avoid significant outages and other key issues by flagging bugs and vulnerabilities before they make it to production and disrupt the entire application.
- **Improve visibility across development:** Before DevOps methodologies and CI/CD approaches the developers used to struggle to know where exactly the problem was happening. Now, the automated testing practices of the CI/CD pipeline improved the visibility across the software lifecycle. Developers can easily spot and isolate code issues. This, in turn, significantly improved productivity.
- **Reduce costs of delivery:** CI/CD pipeline reduces human intervention across the DevOps lifecycle by automating. This significantly saves the time and money required to develop and deliver high-quality software. Moreover, with a successful CI/CD pipeline in play, the development teams aren't plagued with endless 'code fix' requests, so they can keenly focus on the next projects, maximizing the overall ROI for the company.

- **Enhance code quality:** A CI/CD pipeline enables dev teams to integrate code into a shared repository in small batches. the collaborative work environment enables us to easily identify critical bugs, before code issues make it to production.Thus CI/CD pipeline improves code quality and helps deliver superior software.
- **Get feedback continuously:** Timely feedback is the ultimate benefit of the CI/CD pipeline. 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. 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:** The feedback reports and test results allow everyone in the team to track the project status and immediately understand the build failures, and code integration problems. 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, and performance of the application.

CI/CD

(Continuous Integration/Continuous Delivery)

XENONSTACK



Thank you!