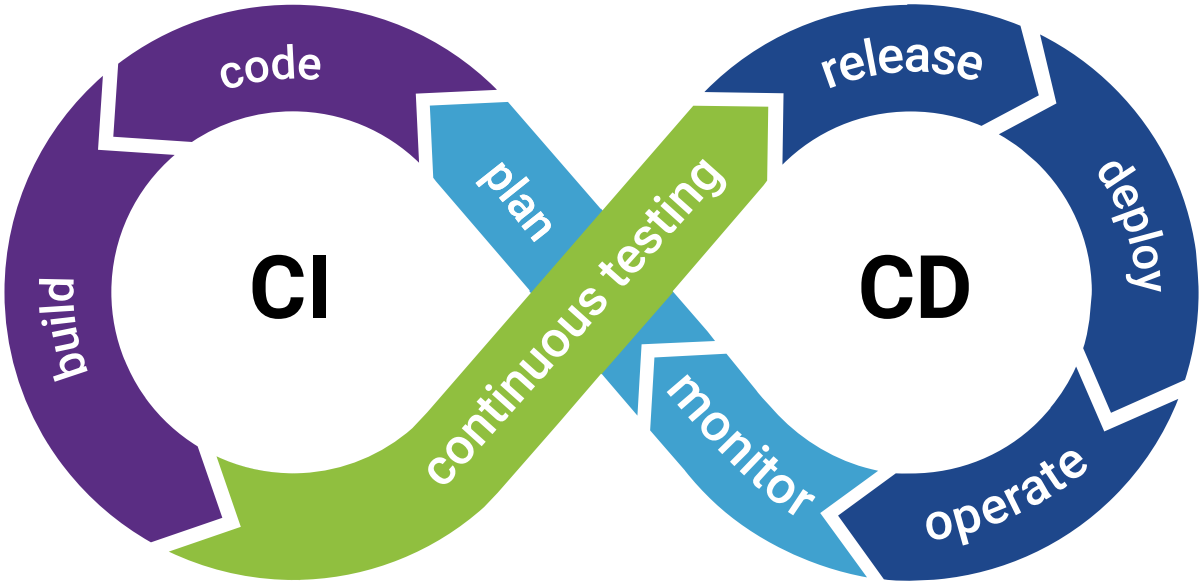
Why CICD?

It’s that thing you never thought you need...



by: Nada Dia

Embrace the reality...

Human is the most valuable resource on this planet, but let's get real.

Before we implement CI/CD almost everything requires human intervention.

Can you imagine a world without human error?

... Neither can we, but with CI/CD, we can reduce it! ... BUT HOW?

What is CICD (According to Gitlab)

CI/CD falls under DevOps (the joining of development and operations) and combines the practices of continuous integration and continuous delivery.

CI/CD automates much, or all of the manual human intervention traditionally needed to get new code from a commit into production such as build, test, and deploy, as well as infrastructure provisioning.

After implementing CICD into our company, we would almost immediately expect:

- Happier users and customers ☺
- Accelerated time-to-value ✓
- Hit dates more reliably ✓
- Reduce burnout ✓
- Recover faster ✓

Benefits of CICD

Less developer time on issues from new developer code	**Reduce Cost ↓**
Less bugs in production and less time in testing	**Avoid cost ✓**
Less human error, Faster deployments	**Avoid cost ✓**
Less infrastructure costs from unused resources	**Reduce Cost ↓**
New value-generating features released more quickly	**Increase revenue ↑**
Less time to market	**Increase revenue ↑**
Quick undo to return production to working state	**Protect revenue ↑**

Business impact

While your closest competitor is already taking advantage of CICD, let's look at this article $3 trillion loss. According to this article, failed IT projects cost businesses around the globe $3 trillion each year.

Failure takes many forms, including non-completion, inability to release, failure to effectively address business needs, cost overruns and missed timelines.

Therefore, the first measurable value of CI/CD pipelines is that they reduce the risk ↓ and cost of IT failure ↓ by producing better code and well-targeted, highly available products and systems.

CI/CD pipelines also automate and optimize provisioning and testing workflows, which reduces errors ↑ and lowers costs ↑.