



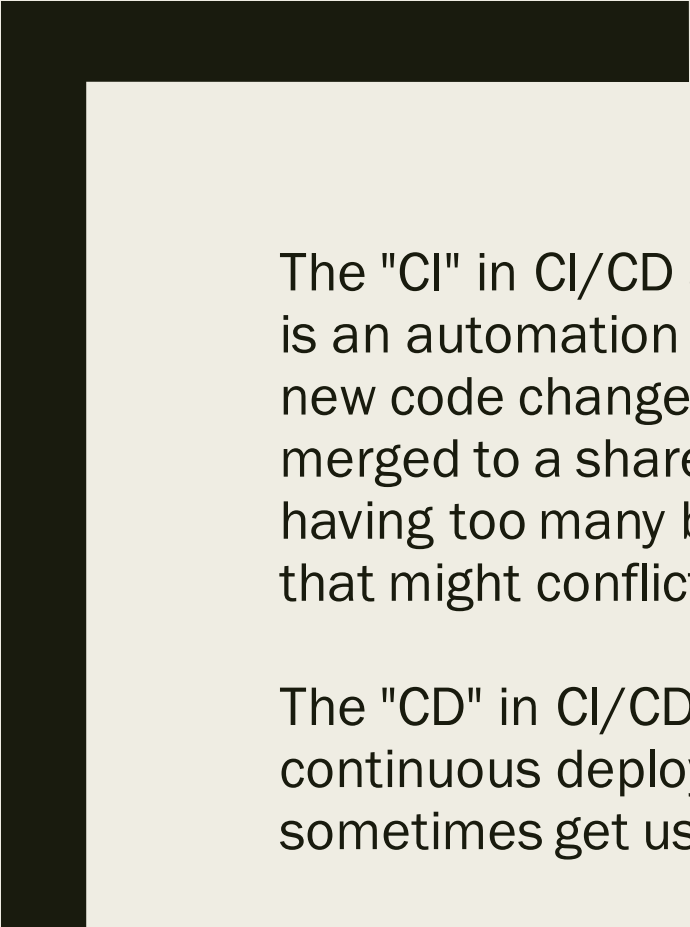
CICD

AN INTRODUCTORY GUIDE



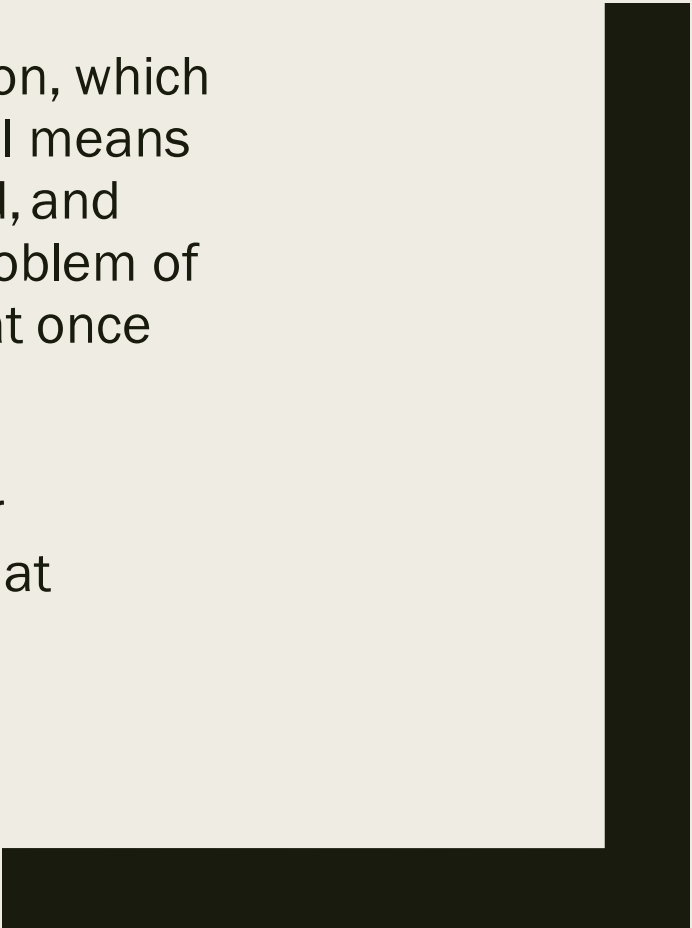
What is CI/CD

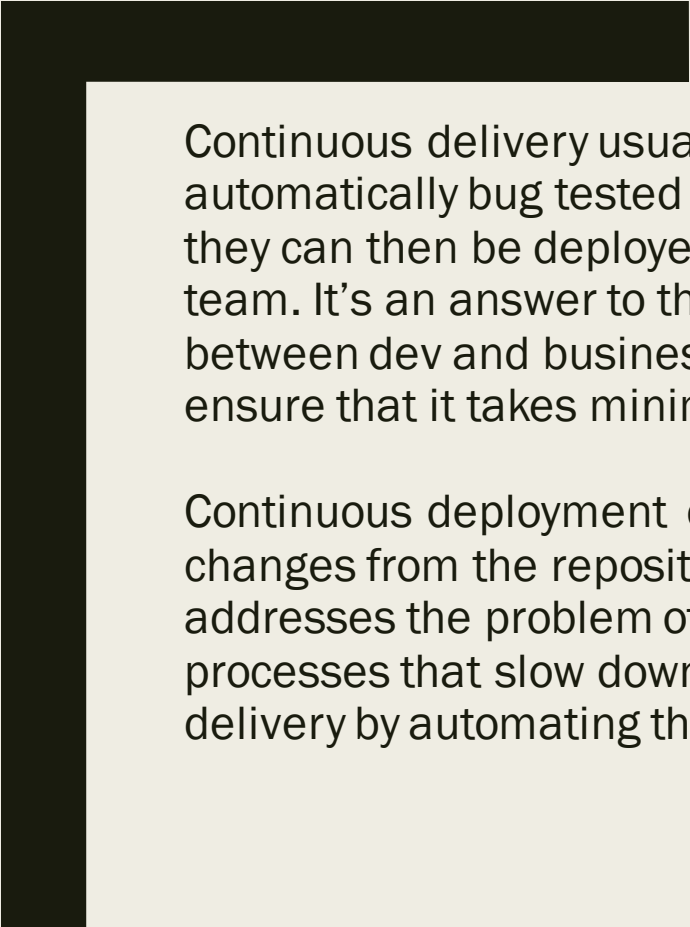
CI/CD is a method to frequently deliver apps to customers by introducing automation into the stages of app development. The main concepts attributed to CI/CD are continuous integration, continuous delivery, and continuous deployment. CI/CD is a solution to the problems integrating new code can cause for development and operations teams

A thick black L-shaped bar in the top-left corner of the slide, consisting of a horizontal segment and a vertical segment.

The "CI" in CI/CD always refers to continuous integration, which is an automation process for developers. Successful CI means new code changes to an app are regularly built, tested, and merged to a shared repository. It's a solution to the problem of having too many branches of an app in development at once that might conflict with each other.

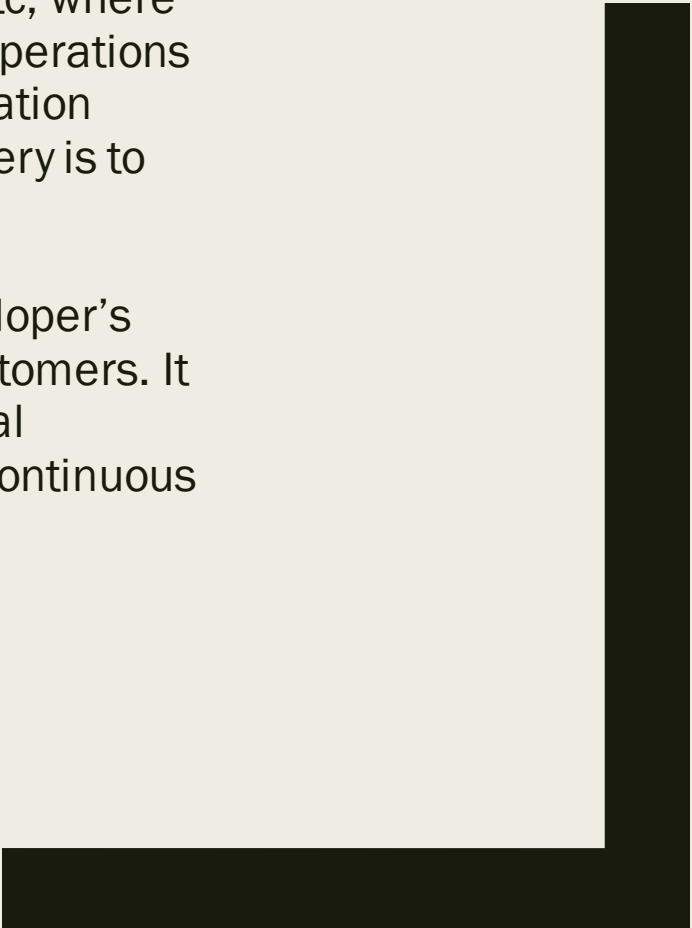
The "CD" in CI/CD refers to continuous delivery and/or continuous deployment, which are related concepts that sometimes get used interchangeably.

A thick black L-shaped bar in the bottom-right corner of the slide, consisting of a horizontal segment and a vertical segment.



Continuous delivery usually means a developer's changes to an application are automatically bug tested and uploaded to a repository like GitHub etc, where they can then be deployed to a live production environment by the operations team. It's an answer to the problem of poor visibility and communication between dev and business teams. The purpose of continuous delivery is to ensure that it takes minimal effort to deploy new code.

Continuous deployment can refer to automatically releasing a developer's changes from the repository to production, where it is usable by customers. It addresses the problem of overloading operations teams with manual processes that slow down app delivery. It builds on the benefits of continuous delivery by automating the next stage in the pipeline.



Benefits of CICD

1. **Faster release cycles and better business advantage:** Speeding up the build and deploy cycle will increase speed to market.
2. **Reduced risk and higher-quality products:** The ultimate goal of a continuous delivery process is to make each release a painless experience for both the QA teams and customers. By releasing features continuously, the risk of bugs ending up in production is drastically reduced.
3. **Lower costs:** Adopting a continuous development model eliminates many of the fixed costs incurred in building and testing changes to the application. For example, automated environment provisioning will reduce the costs associated with maintaining several testbeds and test infrastructure.
4. **Better business advantage:** Moving to a continuous development model gives teams the flexibility to make changes on the go to meet new market trends and user needs. Application monitoring tools are a great way to find and fix failures while also logging the problems to notice trends faster. It also aids in understanding the feedback we receive from customers to ensure our products meet rapidly changing demands.