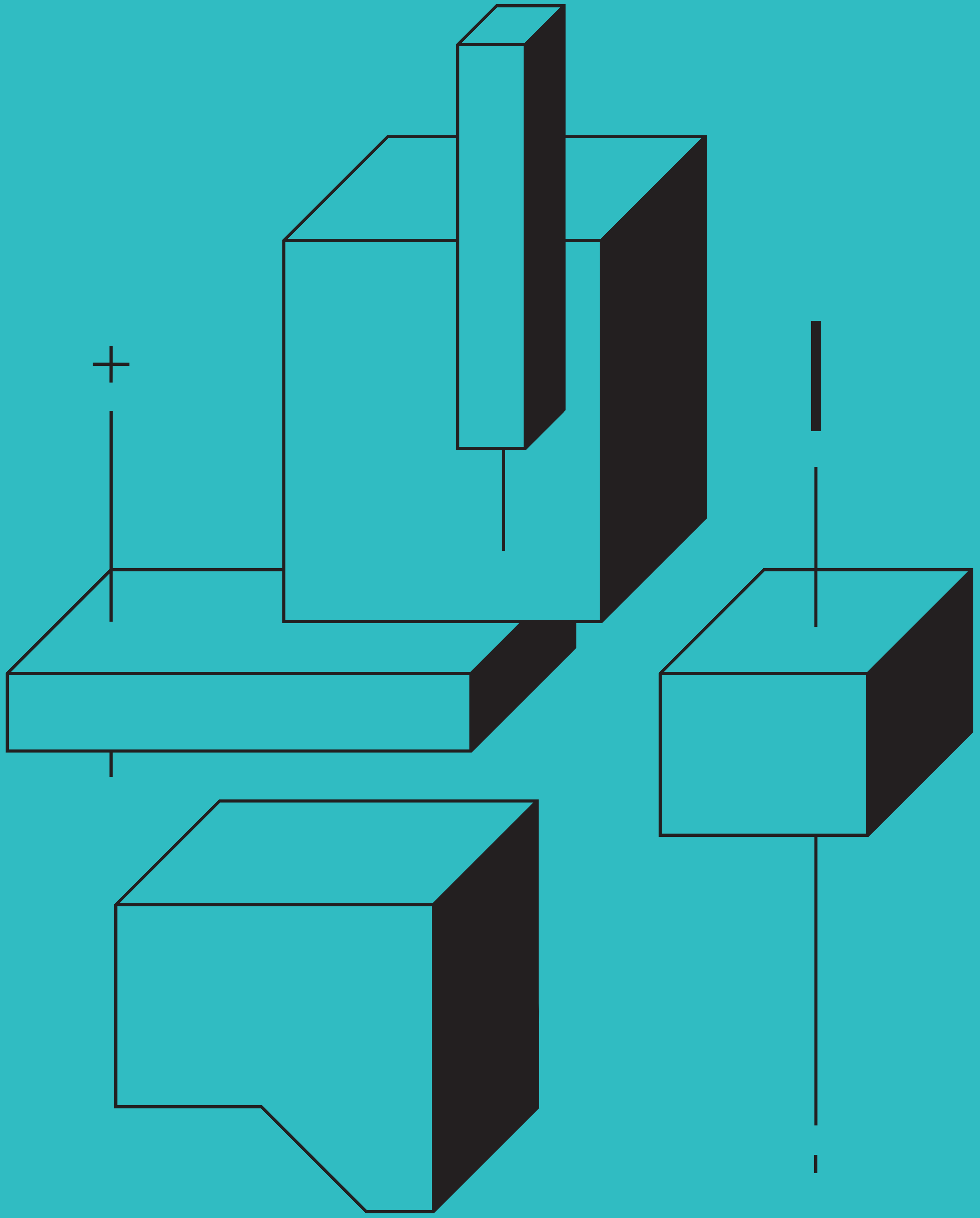
CI/CD Benefits Proposal

Created by
Khoa Nguyen Minh



Is it important?

CI/CD automates your builds, testing, and deployment so you can ship code changes faster and more reliably.

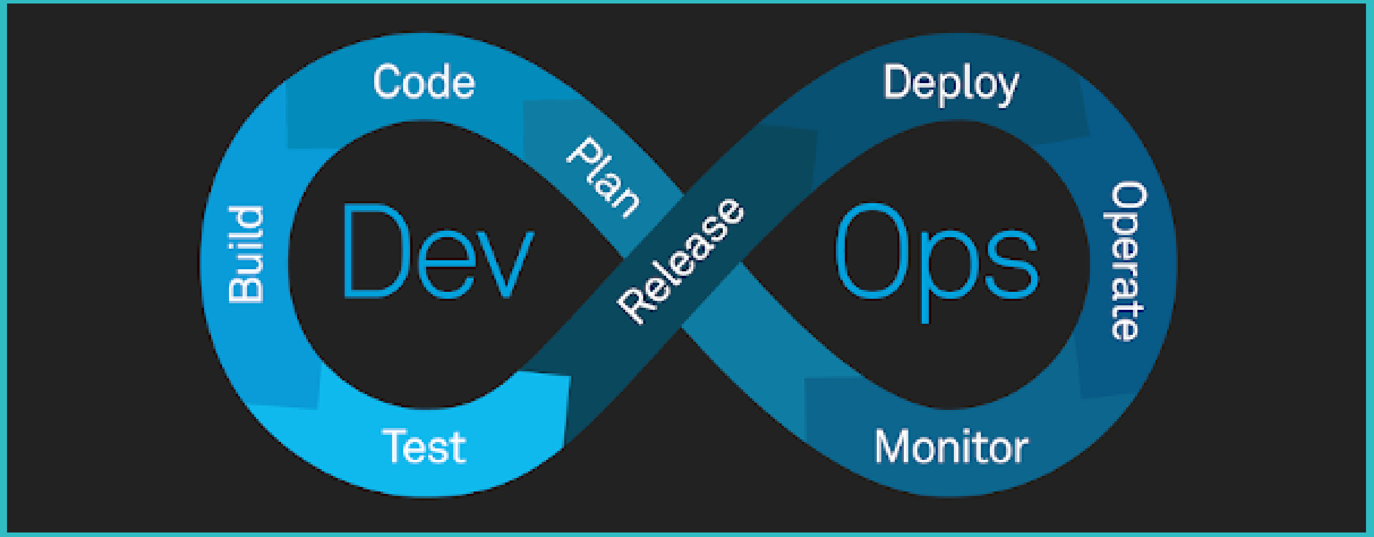
Automation is a core principle for achieving DevOps success and CI/CD is a critical component.

The State of DevOps report found organizations that have “mastered” CI/CD deploy 208 times more often and have a lead time that is 106 times faster than the rest.

The Fundamentals

Continuous Intergration

Continuous Integration (CI) is a development practice that involves frequently integrating code changes into a shared repository. These changes are automatically built and tested to ensure that they integrate well with the existing codebase.



Put together, they form a “CI/CD pipeline”—a series of automated workflows that help DevOps teams cut down on manual tasks

Continuous Deployment

Continuous Deployment (CD) is a development practice where code changes are automatically deployed to customers as soon as they pass all the required tests. It is the ultimate example of DevOps automation.

Best Practices

What you should do when integrating CI/CD

Fail Fast

Set up your CI/CD pipeline to find and reveal failures as fast as possible. The faster you can bring your code failures to light, the faster you can fix them.

Measure Quality

Measure your code quality so that you can see the positive effects of your improvement work (or the negative effects of technical debt).

Only Road to Production

Once CI/CD is deploying to production on your behalf, it must be the only way to deploy. Any other person or process that meddles with production after CI/CD is running will inevitably cause CI/CD to become inconsistent and fail.

Maximum Automation

If it can be automated, automate it. This will only improve your process!

Config in Code

All configuration code must be in code and versioned alongside your production code. This includes the CI/CD configuration files!

Benefits

- Less developer time on issues from new developer code
- Less bugs in production and less time in testing
- Prevent embarrassing or costly security holes
- Less human error, Faster deployments
- Less infrastructure costs from unused resources
- New value-generating features released more quickly
- Less time to market
- Reduced downtime from a deploy-related crash or major bug
- Quick undo to return production to working state