Potential Questions from the CI Presentation are

2 Marks Questions:

1. What is Continuous Integration?
2. What is the DevOps methodology?
3. What are the benefits of Continuous Integration?
4. Name one real-world example of Continuous Integration.
5. What is the role of Continuous Integration in DevOps?

5 Marks Questions:

1. How does Continuous Integration fit into the Software Development Lifecycle?
2. Explain the Continuous Integration process flow.
3. What are some of the key components of Continuous Integration?
4. What are the best practices of Continuous Integration?
5. What are the challenges of implementing Continuous Integration?

10 Marks Questions:

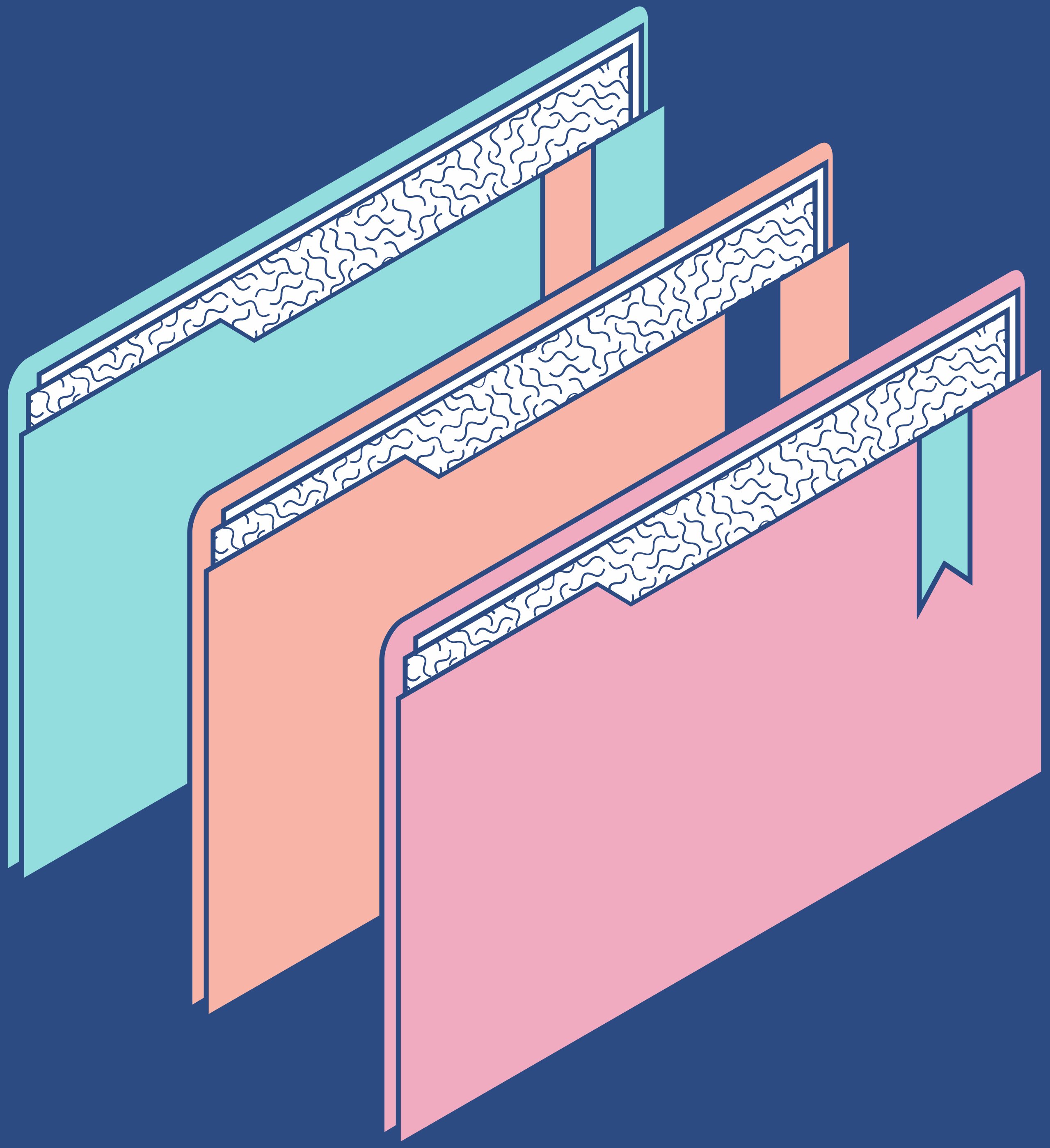
1. Describe the DevOps methodology and explain the role of Continuous Integration in DevOps.
2. Compare and contrast Continuous Integration with Continuous Delivery and Continuous Deployment.
3. Explain the benefits and drawbacks of Continuous Integration.
4. Discuss some of the tools used in Continuous Integration and their features.
5. Analyze the real-world examples of companies using Continuous Integration and identify their best practices, challenges, and solutions.

ITPM PRESENTATION

Continuous Integration

Varun Khadayate A016

Rugved Kulkarni A017



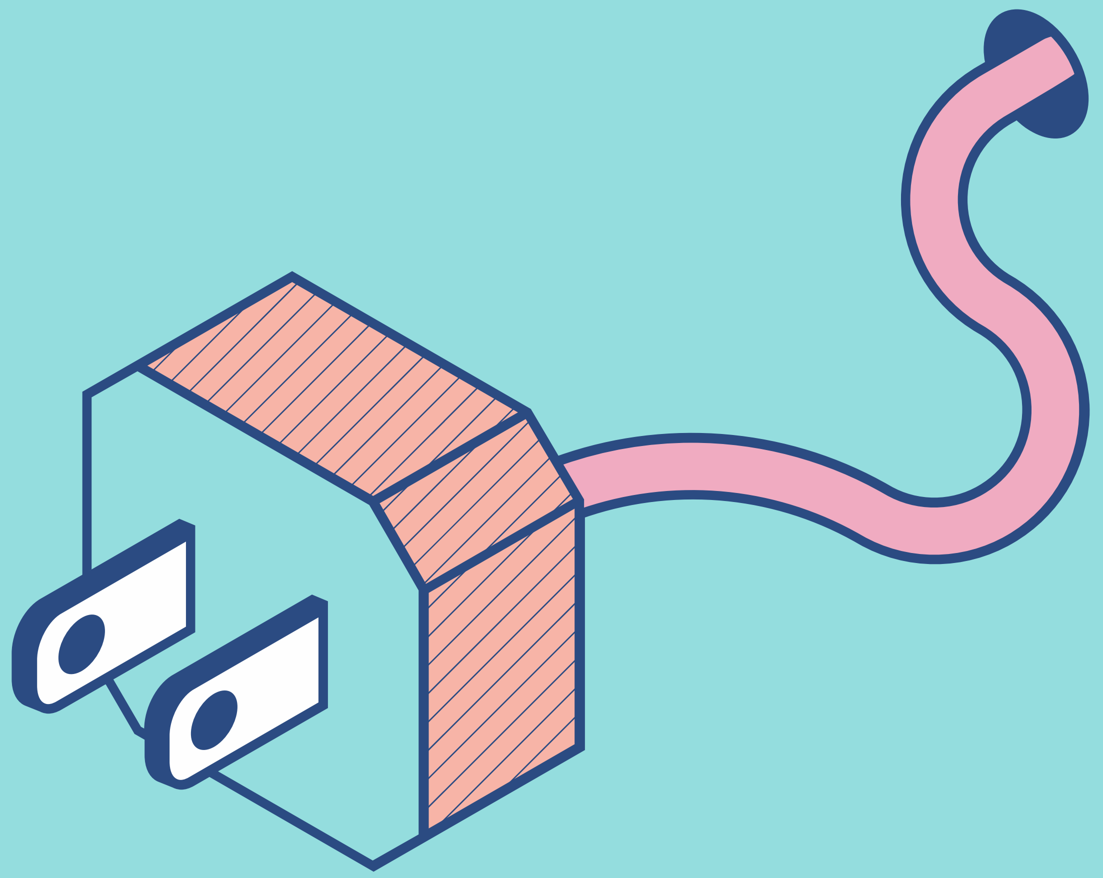
Agenda

KEY TOPICS DISCUSSED IN
THIS PRESENTATION

- What is Continuous Integration?
- DevOps methodology
- Continuous Integration in Software Delivery
- CI real-world examples
- Conclusion

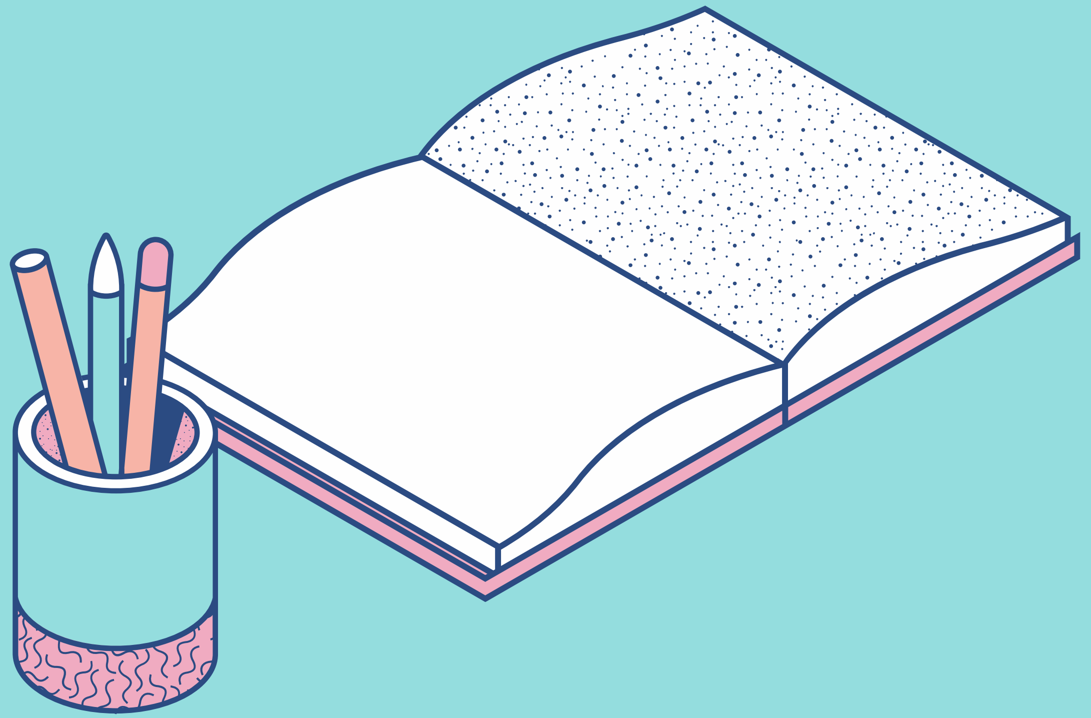
What is Continuous Integration

CI is a practice in software development where developers frequently integrate their code changes into a shared code repository. Each integration is verified by an automated build and test process to detect errors early in the development cycle.

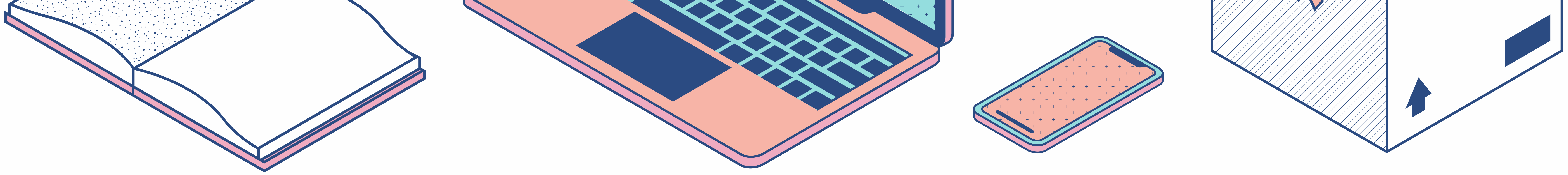


Importance of CI

Continuous integration (CI) is a software development practice that automatically and regularly integrates code changes into a central repository, providing early issue detection, faster development cycles, improved collaboration, higher code quality, and improved reliability.



DevOps methodology



Definition of DevOps

DevOps is a methodology that aims to improve collaboration between development and operations teams to deliver software faster, more reliably, and with better quality.

Key principles of DevOps

The key principles of DevOps include automation, collaboration, continuous feedback, and continuous improvement.

Importance of CI in DevOps

CI is a critical component of the DevOps methodology because it enables faster and more frequent deployments, which is essential for delivering software at the speed of business.

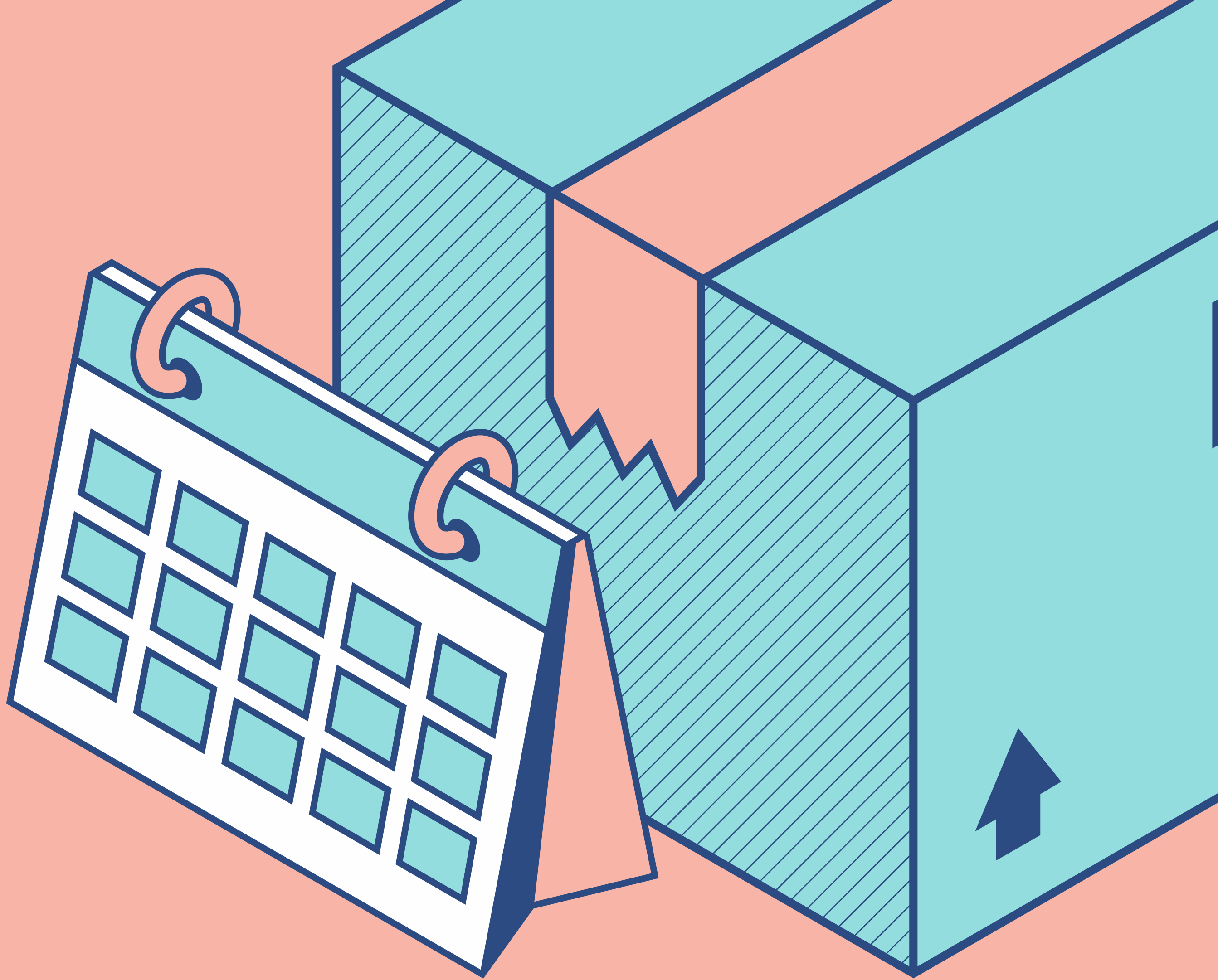
Continuous Integration use in Software Delivery

Faster feedback

CI allows developers to get immediate feedback on their code changes, making it easier to identify and fix bugs early in the development process.

Early bug detection

By integrating code changes frequently, CI can detect errors and conflicts early, preventing them from becoming more significant problems later in the development process.



Consistent builds

CI ensures that all code changes are automatically built, tested, and verified against a set of predefined standards, resulting in consistent and predictable builds.

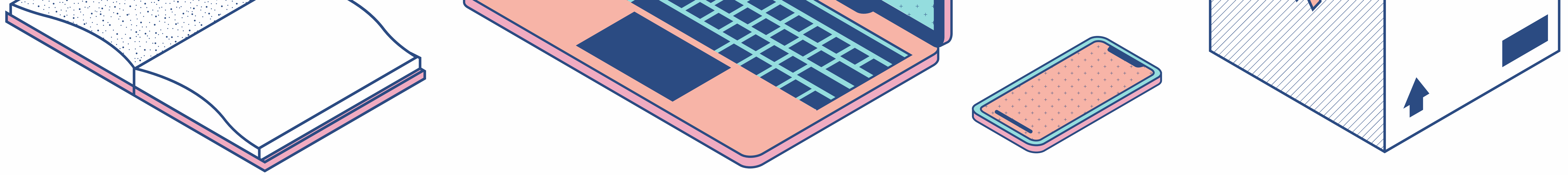
Improved collaboration

With CI, all team members can work on the same codebase simultaneously, without worrying about code conflicts or integration issues.

Continuous delivery

CI enables continuous delivery by automating the build, test, and deployment process. This makes it easier to release new features and updates to customers quickly and frequently.

How CI differs itself from traditional methods



Frequency of integration

With traditional methods, integration typically happens at the end of the development cycle, after all the code has been written. In contrast, CI emphasizes frequent integration, with code changes being integrated and tested as soon as possible.

Automation

CI relies heavily on automation tools and processes to build, test, and deploy code changes. This helps to speed up the development process and reduce the risk of errors and inconsistencies.

Emphasis on testing

In traditional methods, testing is often seen as a separate activity that happens after development is complete. In CI, testing is integrated into the development process, with automated tests being run every time a code change is made.

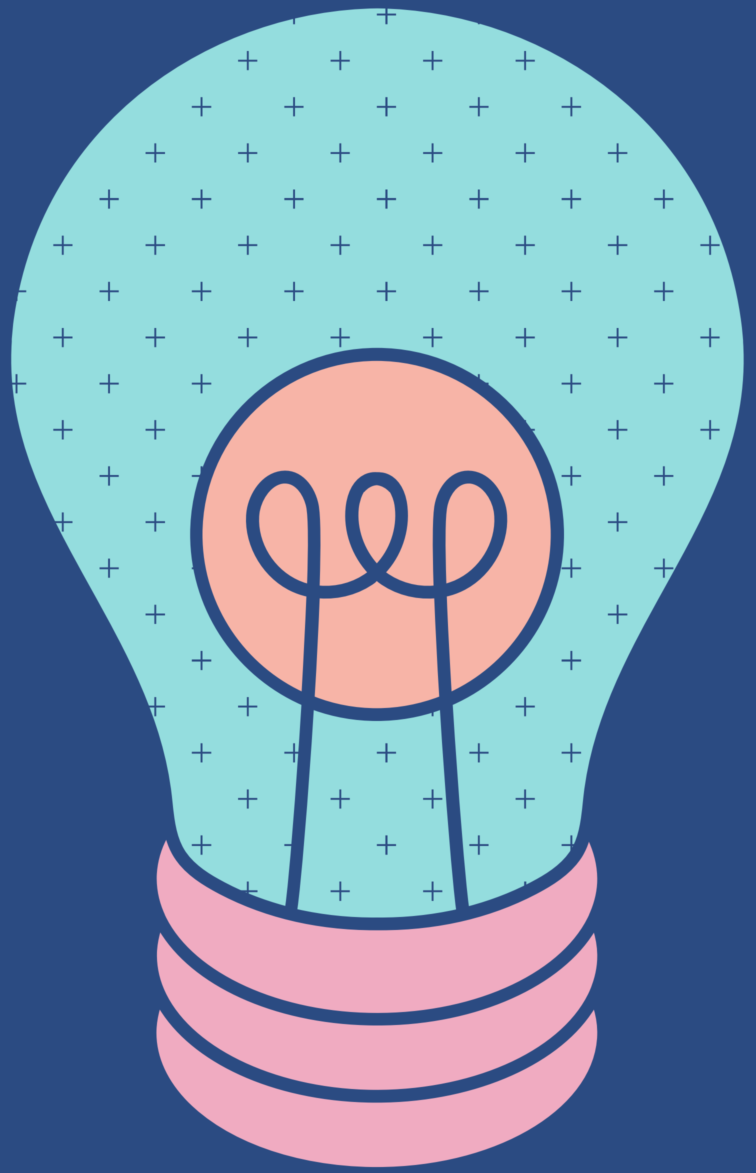
Collaborative development

CI emphasizes collaboration between developers, with all team members working on the same codebase and integrating their changes frequently. This helps to reduce the risk of code conflicts and integration issues.

Continuous delivery

CI enables continuous delivery, with code changes being released to production as soon as they are validated and tested. In traditional methods, release cycles can be slower and more cumbersome, with more extensive manual testing and deployment processes.

Real life Examples of Continuous Integration



Netflix

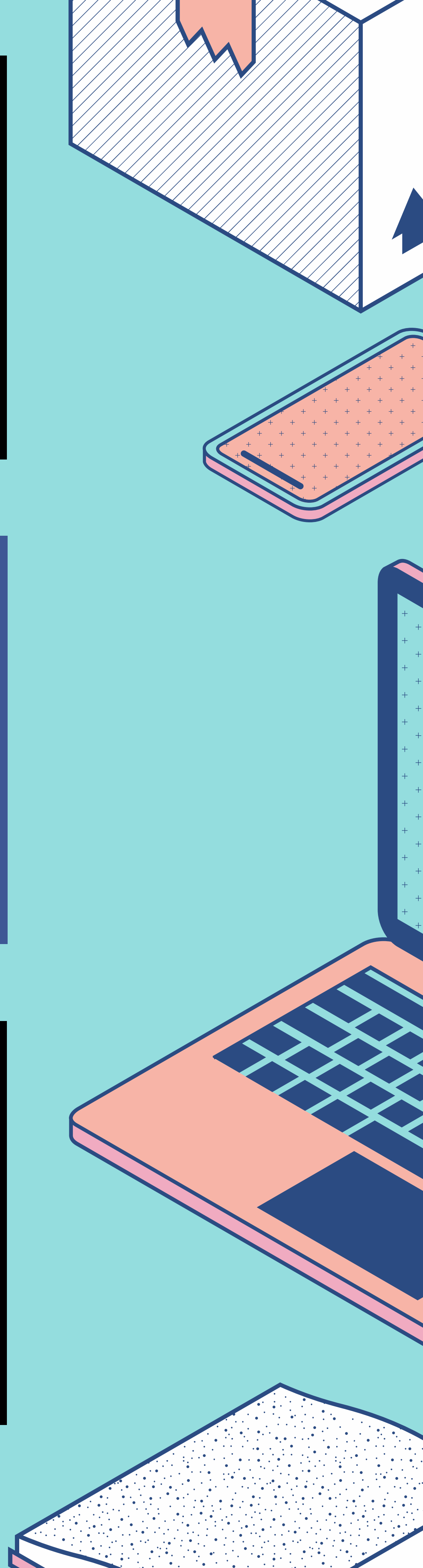
Netflix uses CI to deliver new features and updates to its streaming platform continuously. With CI, Netflix can test and verify changes to its codebase quickly and efficiently, ensuring that its service remains stable and reliable.

Facebook

Facebook's development teams use CI to ensure that code changes are thoroughly tested and reviewed before they are released to production. This helps to minimize the risk of bugs and other issues, ensuring that Facebook remains stable and reliable for its billions of users.

Spotify

Spotify uses CI to continuously integrate code changes from its development teams and automate its testing and deployment process. This allows Spotify to release new features and updates to its music streaming service frequently and consistently, keeping its customers engaged and satisfied.



Questions?

"Continuous integration: because who has time for manual toothbrushing or manual testing?"

Thank You