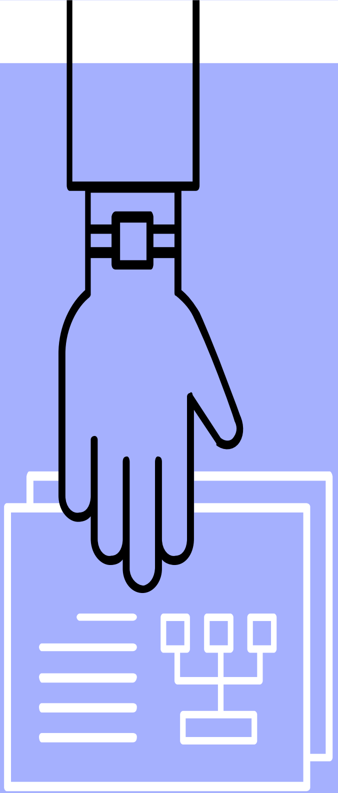
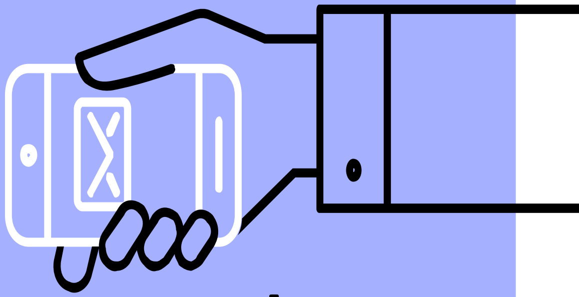
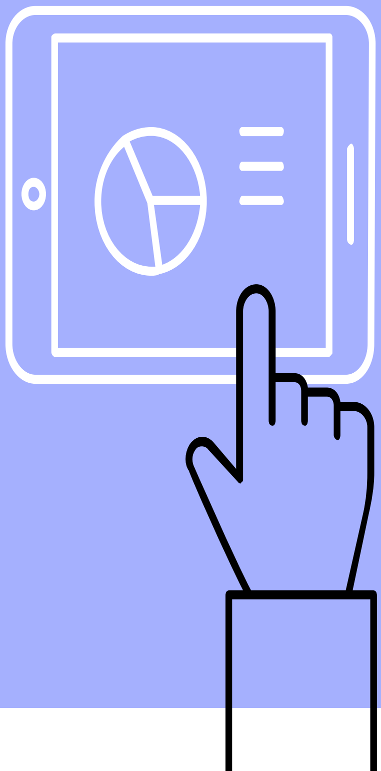
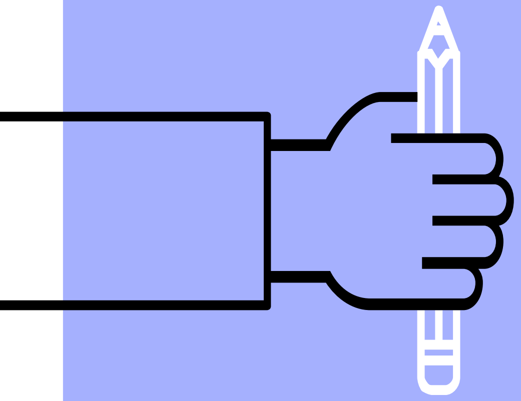



Fundamentals and benefits of CICD

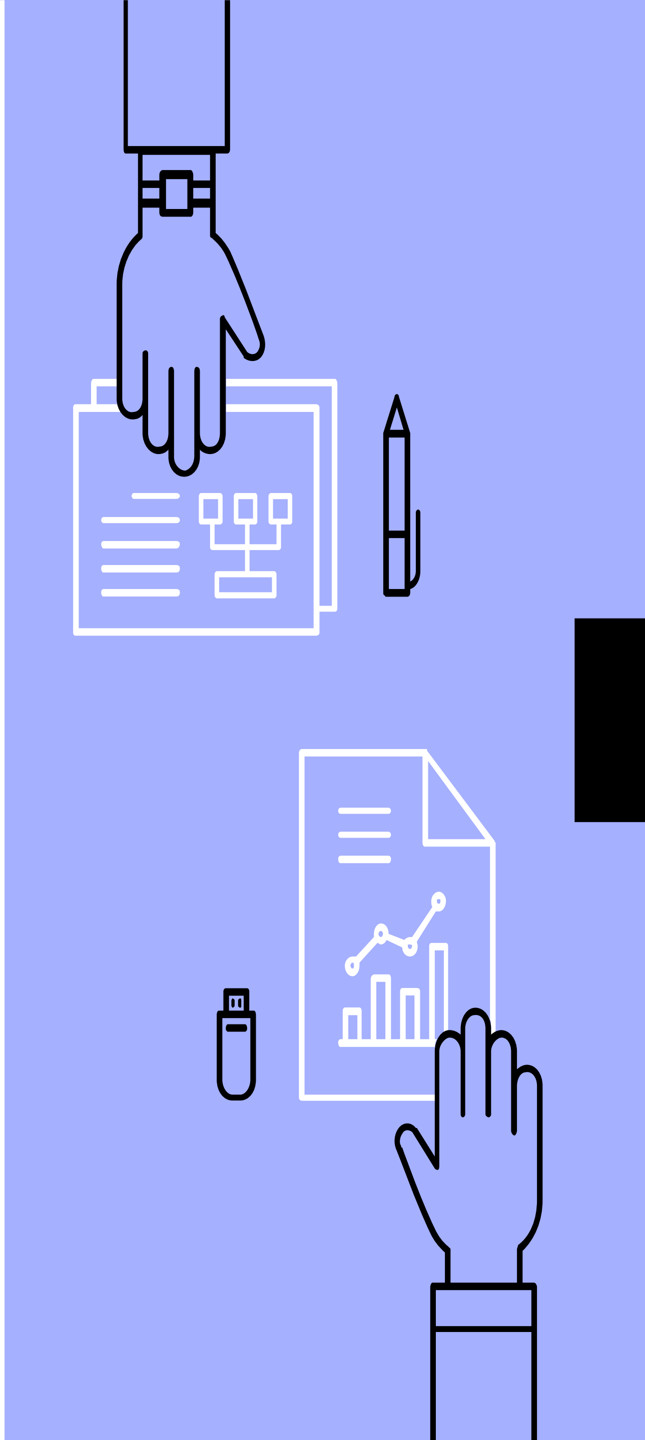



Continuous integration

- CI as it’s often known, is the practice of having everyone working on the same software project share their changes to the codebase regularly and then checking that the code still works as it should after each change.

Continuous Deployment

- CD as it’s often known, If a change to the code successfully passes all previous stages of the pipeline, that change is automatically deployed to production without any manual intervention.



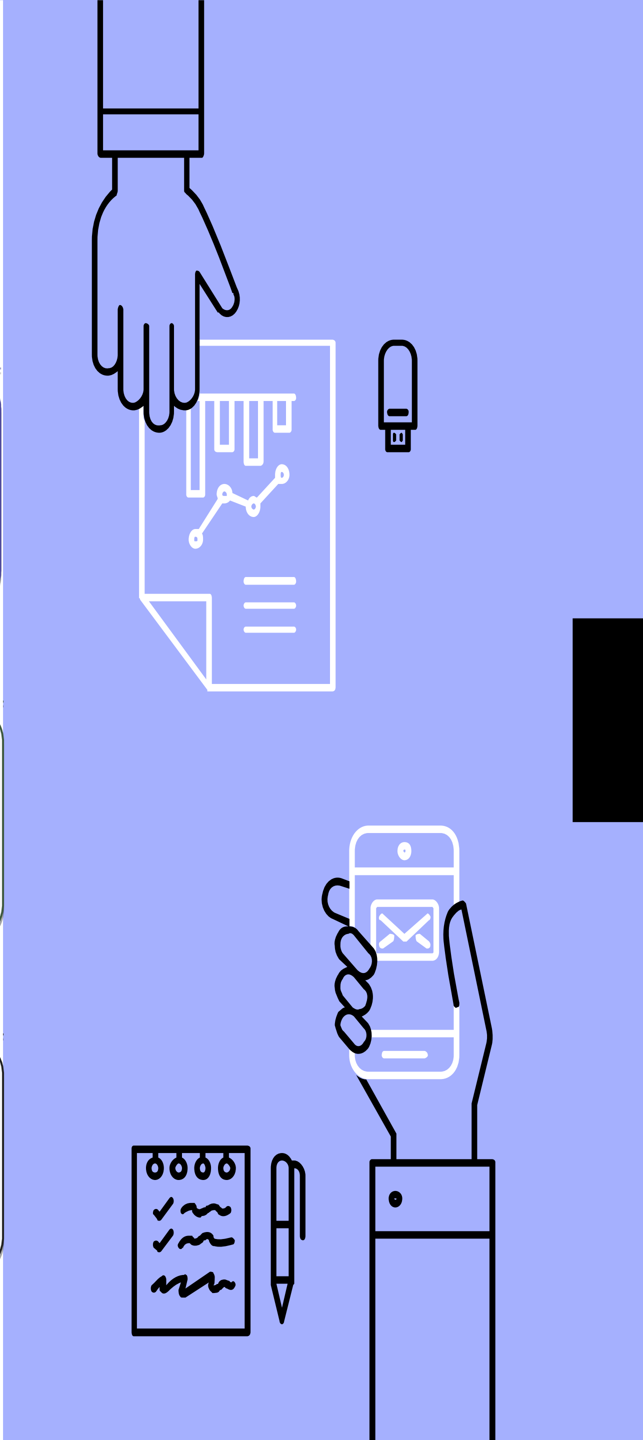
CICD | What for?

Benefits of CI/CD

1
- **Faster time to production and market:** The delivery of working software to users quickly and frequently which is led to publish and deliver improvement to users faster

2
- **Enhancement the code quality:** automated tests performed consistently and extending the test coverage

3
- **Early Fail and fix bugs:** If you're committing changes regularly and shipping frequently, each release to production will contain a relatively small number of code changes, making it much easier to identify the cause of an issue



CICD | What for?

Benefits of CI/CD

4

- **a collaborative culture:** It provides the oversight and reporting needed to be able to share high-level objectives and strategy in business terms with key decision makers and executives.

5

- **Automating the creation of Infrastructure as a code:** their configuration is scripted and stored in version control so that new environments can be brought online quickly without the risk of inadvertent changes and inconsistencies.

6

- **Rapid Feedback:** It starts with automated build and test steps to inform you of immediate problems, helping you to work more efficiently and effectively than if there is a long delay between the original work and the results.

CICD | What for?

Benefits of CI/CD

7

- **Faster Code review:** With continuous integration, developers are encouraged to commit their code changes more frequently – at least once a day as a rule of thumb. Sharing code with the rest of the team regularly not only ensures everyone is building on the same foundation

8

- **Rapid releases:** Being able to test your innovations with users early and often – either with test participants in a pre-production environment or with real users in live – means you can validate your approach before investing months or even years working on a feature that doesn't actually solve a problem for your users.

9

- **Smoke tests automation:** process came after deployment to reduce downtime and risks.