Udapeople By CI/CD

Done By: Nourhan Mohamed

Definition for CI/CD

Continuous Integration

- Developers practicing continuous integration merge their changes back to the main branch as often as possible.
- The developer's changes are validated by creating a build and running automated tests against the build. By doing so, you avoid integration challenges that can happen when waiting for release day to merge changes into the release branch.
- Continuous integration puts a great emphasis on testing automation to check that the application is not broken whenever new commits are integrated into the main branch.

Continuous Delivery

- Continuous delivery is an extension of continuous integration since it automatically deploys all code changes to a testing and/or production environment after the build stage.
- This means that on top of automated testing, you have an automated release process and you can deploy your application any time by clicking a button.

Continuous Deployment

- Continuous deployment goes one step further than continuous delivery. With this practice, every change that passes all stages of your production pipeline is released to your customers. There's no human intervention, and only a failed test will prevent a new change to be deployed to production.
- Continuous deployment is an excellent way to accelerate the feedback loop with your customers and take pressure off the team as there isn't a Release Day anymore. Developers can focus on building software, and they see their work go live minutes after they've finished working on it.

Benifets for CI/CD

- **Smaller code** changes are simpler (more atomic) and have fewer unintended consequences.
- **Fault isolation** is simpler and quicker.
- **Mean time to resolution (MTTR)** is shorter because of the smaller code changes and quicker fault isolation.
- **Testability** improves due to smaller, specific changes. These smaller changes allow more accurate positive and negative tests.
- **Elapsed time** to detect and correct production escapes is shorter with a faster rate of release.
- **The backlog** of non-critical defects is lower because defects are often fixed before other feature pressures arise.
- The product improves rapidly through **fast feature introduction** and **fast turn-around** on feature changes.

Benefits Contd's

- **Upgrades** introduce smaller units of change and are less disruptive.
- **CI-CD product feature velocity is high**. The high velocity improves the time spent investigating and patching defects.
- **Feature toggles and blue-green deploys** enable seamless, targeted introduction of new production features.
- You can introduce **critical changes during non-critical** (regional) hours. This non-critical hour change introduction limits the potential impact of a deployment problem.
- **Release cycles** are shorter with targeted releases and this blocks fewer features that aren't ready for release.
- **End-user involvement** and feedback during continuous development leads to usability improvements. You can add new requirements based on customer's needs on a daily basis.