- What are CI and CD, and why are they both important?

Continuous Integration (CI) and Continuous Delivery (CD) enhance the development process of software development by streamlining collaboration and ensuring the delivery of high-quality, reliable software.

CI is a development practice that involves developers frequently integrating their code changes into a shared repository, such as version control, multiple times throughout the build. The process typically involves automated builds and tests to verify that new changes don't break the existing codebase. This helps identify and fix problems early, making projects cheaper, more stable and reliable - whilst encouraging teams to work collaboratively by promoting a culture of shared responsibility and collective ownership of the codebase.

CD is an extension of CI, which allows developers to ensure the software is always ready to be deployed at all times, even as many developers are making regular changes. It includes automated deployment to production-like environments, enabling quick, reliable releases, thus reducing the risk of errors as smaller changes are easier to comprehend and resolve than bigger ones.

CI ensures that code changes are consistently integrated and tested, while CD extends this to automated, ready-to-deploy states, enabling rapid and reliable releases. Both practices enhance collaboration, reduce integration issues, and contribute to delivering high-quality software efficiently and effectively.

- What are the advantages of using CI and CD tools during the development process?

Using CI and CD tools in the development process offers several advantages that enhance the efficiency, quality, and reliability of the software being developed.

CI enables early bug detection and encourages teamwork, as working together daily means everyone knows what they are all doing and avoids confusion. Offers immediate feedback and provides confidence with the integration as the work is constantly merging.

Advantages of CD is fast delivery, as everything is prepared and tested. Confidence in any changes and amendments as the strong history of the project is known. Ability to quickly adapt to market requests,requirements or enhancements, which offers a happier user experience, as users get to see new, improved versions of projects sooner.

Using these tools significantly improves software development. CI ensures early bug spotting, promotes collaboration, and provides quick feedback, fostering a confident integration process. Whilst CD guarantees swift, well-prepared deliveries, boosting confidence in changes and facilitating adaptability to market demands, leading to an enhanced user experience with faster access to improved project versions.

- How could you use CI and CD practices during the development of your Meet app's development?

I can utilise CI practices by using a version control system like Git. Making small, frequent commits and tracking all changes. Set up tools that automatically build my app as well as run tests whenever new code is added. Monitoring the automated build and test results to ensure everything integrates smoothly.

With CD I'd configure tools to automatically deploy my app to a production server whenever the CI tests pass. Consistently testing the environments to ensure it is similar to the production environment to catch potential issues early. Use incremental updates, ensuring they are well-tested and ready to be deployed. Automate monitoring through Atatus, to keep an eye on my app in the production environment and automatically notify you if something goes wrong. Gather feedback from users after each deployment to make improvements and updates based on their experience.