Jenkins

- Jenkins is a Software tool that automates the CI/CD Pipeline.

- It's like an automated tool that decreases the efforts of the testing team and fastens the Continuous Delivery Process.

i.e., the whole process b/w the `committing changes’ and `deploying the changes’ is automated with minor validations and inputs from the testing team.

- Jenkins offers a server to perform automated tasks, known as CI Server.

=>**Before Jenkins**:

- The Developers make changes to the code and commit them.

- Before pushing it to the common repository, the Testing team creates a build to run the application locally with latest code on it, and creates various test cases to test the code, including unit testing, integration testing, regression testing, and QA testing.

- After the testing phase, the committed code will be pushed and integrated in the common repository of the application.

- Problems Caused:

+ Since the testing phase is manual, the testing team is not always available to perform tests and developers can create pull requests at any time. Due to which the developers must wait until the testers are available and until the testing phase is completed, to get the review updates, which delays the project.
+ In the Testing phase, the testers must create a build manually and create and perform test cases manually. This only works for short projects. But in long projects which have large development team, many developers commit the changes at any time and the testing team cannot be as large as the development team. Hence, the testers may take ample amounts of time to go through all the commits and test them individually and manually, which further delays the project.

=>**After Jenkins**:

- With Jenkins, the whole process of creating a build and performing test cases on the build with newly committed code can be automated without the testers input.

- The inputs of testing team are:

1. Creating test cases to perform whenever the developer commits the new changes. (only once)
2. Ensuring the automated testing process.
3. Validating and pushing the committed code into a common repository.

(validating => check if all the test cases are passed and if the build is performing as intended.)

- Overall Efficiency:

+ Now, the Developers can commit the changes at any time and expect the reviews instantly.

+ Testing team doesn’t have to create builds and perform tests manually (but must monitor them). Hence the testing team doesn’t have to be large.
+ The latest builds are delivered on time without any delays.