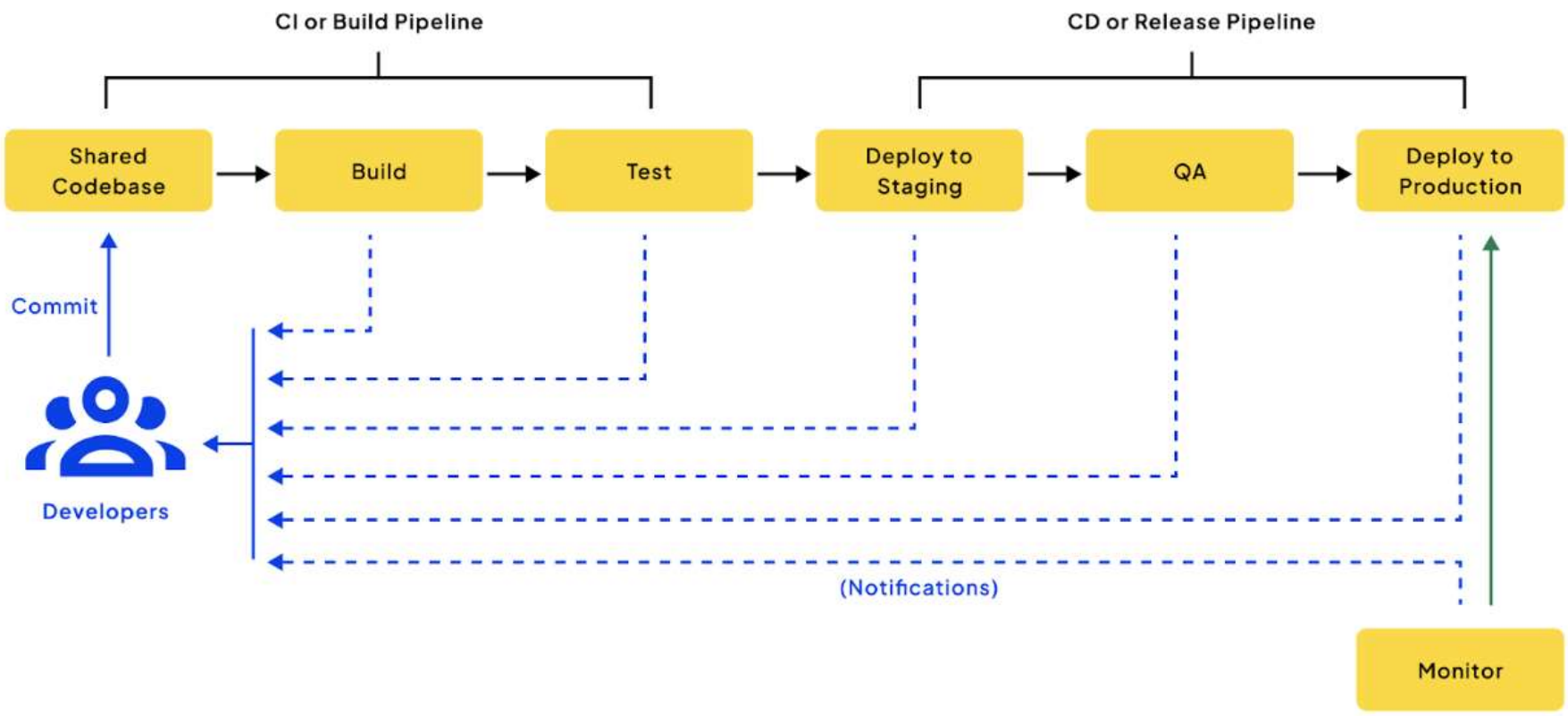
AmazingDevOps

CI/CD Pipeline



Source: https://www.uffizzi.com/blog/continuous-integration-vs-continuous-deployment

1. Planning Stage:

- Define project requirements and plan development tasks.

2. GitHub Repository:

- Store the quality code that has passed the code review process.
- Developers commit their code changes to GitHub.

3. Git:

- Clone the source code from the GitHub repository.

4. Jenkins:

- Automated build and continuous integration server.
- Triggered by code commits or pull requests to the GitHub repository.
- Builds the application using Maven.
- Configured and provisioned using Ansible.

5. Maven:

- Build and dependency management tool.
- Used by Jenkins to build the application.

6. SonarQube:

- Code quality and static code analysis tool.
- Analyzes the code for bugs, vulnerabilities, and code smells.
- Provides feedback on code quality.

7. JFrog Artifactory:

- Artifact management system.
- Store .jar artifacts produced by the build process.
- Acts as a central repository for artifacts.

8. Ansible:

- Configuration management tool.
- Manages the setup and configuration of Jenkins, Maven, SonarQube, JFrog Artifactory, and other CI/CD servers.

- Ensures consistency and reproducibility of the server environment.

9. Docker:

- Containerization platform.

- Builds, ships, and runs applications inside containers for consistency.

10. Docker Hub Repository (Staging Area):

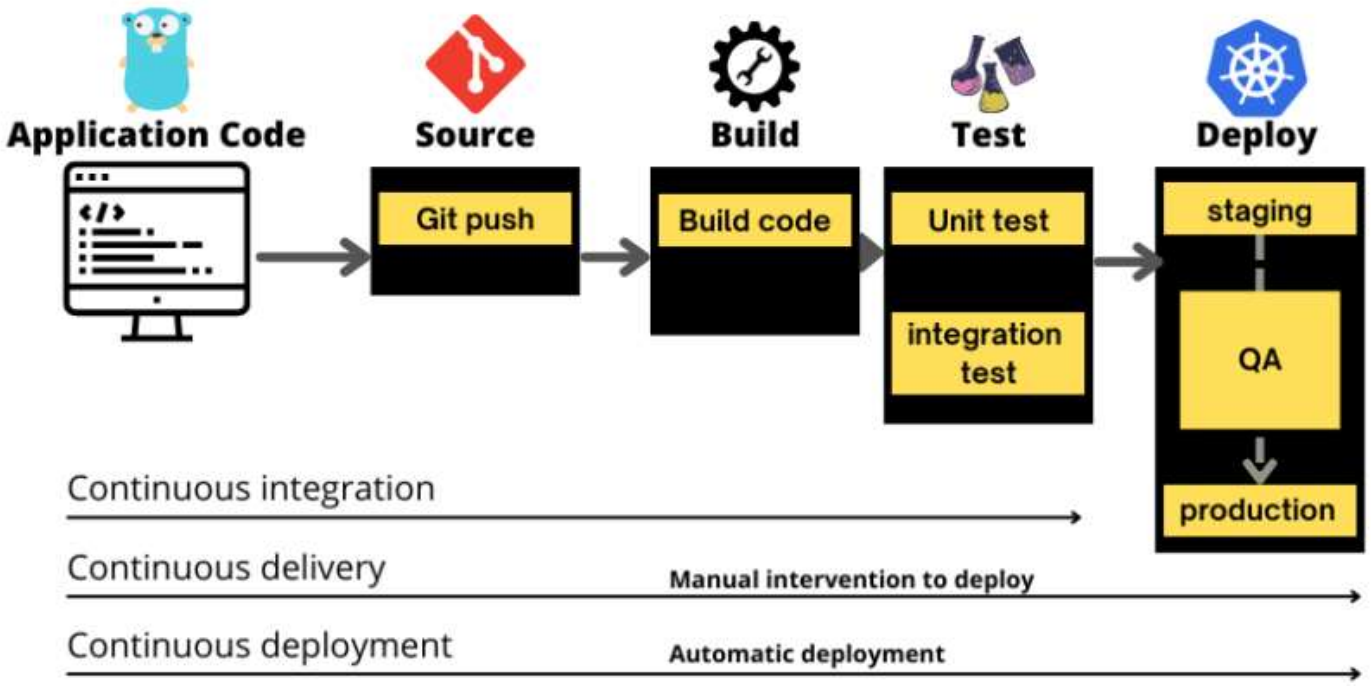
- Store Docker images that are considered the staging area for production.

- After successful build and tests, Docker images are pushed to Docker Hub.

11. Production Environment:

- The Docker images from Docker Hub are deployed to the production environment.

With the addition of JFrog Artifactory, your CI/CD pipeline will have a reliable system for storing and managing the .jar artifacts produced during the build process. Artifactory ensures that you can easily retrieve and share these artifacts across different environments and projects, improving the efficiency and reliability of your software delivery process.



Source: https://dev.to/pavanbelagatti/learn-how-to-setup-a-cicd-pipeline-from-scratch-for-a-go-application-4m69

Stage View

	git cloning	Maven Clean Build	Build Docker Image	Docker Image list	Docker Image Tag	Docker Login to Hub Docker	Docker Image Push to the staging of production
Average stage times: (Average full run time: ~14s)	391ms	9s	3s	287ms	509ms	362ms	7s
#11 Jul 21 14:28 No Changes	284ms	Success Logs	2s	283ms	553ms	320ms	5s