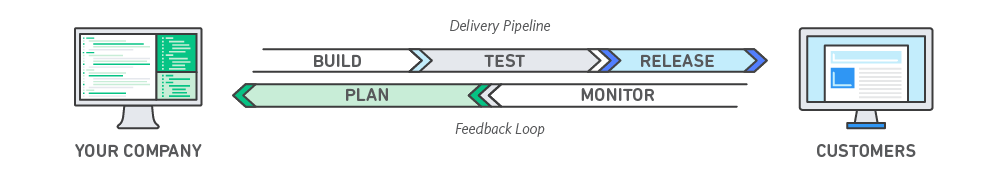
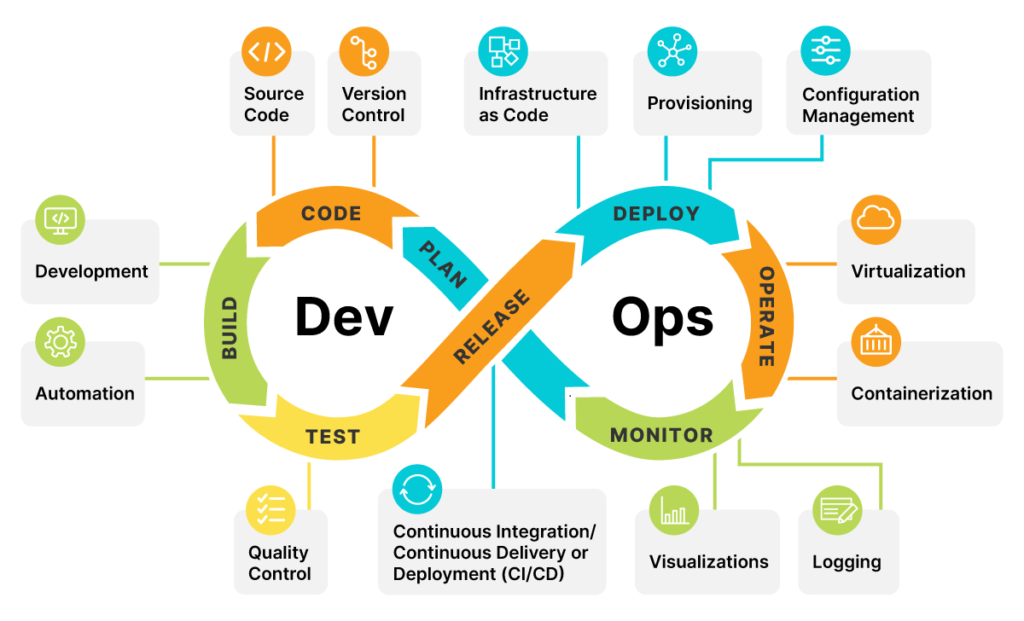
**DEVOPS**

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization’s ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. This speed enables organizations to better serve their customers and compete more effectively in the market.





**DevOps Tool Chain:**

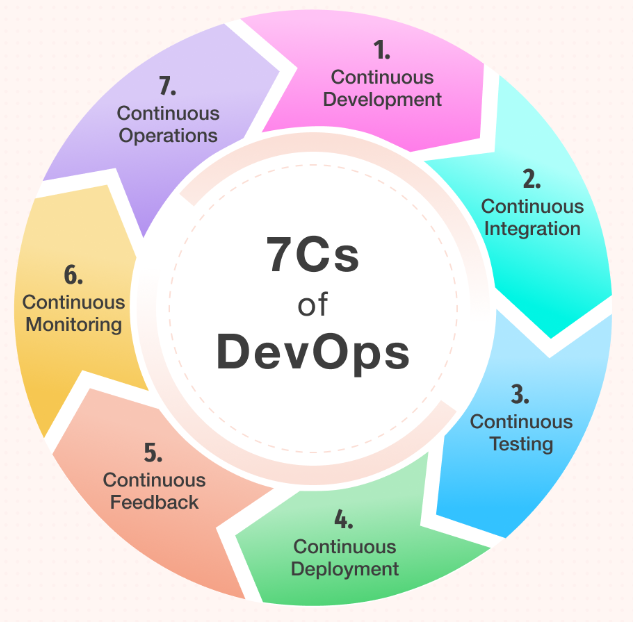
The goal of tools is to further streamline, shorten, and automate the various stages of the software delivery workflow (or “pipeline”). Many such tools also promote core DevOps tenets of automation, collaboration, and integration between development and operations teams. The following shows a sample of tools used at various DevOps lifecycle stages.

* **Plan.** This phase helps define business value and requirements. Sample tools include to help track known issues and perform project management.
* **Code.** This phase involves software design and the creation of software code. Sample tools include GitHub, GitLab, Bitbucket, or Stash.
* **Build.** In this phase, you manage software builds and versions, and use automated tools to help compile and package code for future release to production. You use source code repositories or package repositories that also “package” infrastructure needed for product release. Sample tools include Docker, Gradle, Maven, or NPM.
* **Test.** This phase involves continuous testing (manual or automated) to ensure optimal code quality. Sample tools include JUnit, Codeception, Selenium, TestNG, or BlazeMeter.
* **Release.** This phase involves preparing the application ready for delivery. This includes tools such as Jenkins, Bamboo, AWS Code pipeline.
* **Deploy.** This phase can include tools that help manage, coordinate, schedule, and automate product releases into production. Sample tools include Docker, K8s, AWS Cloud.
* **Operate.** This phase manages software during production. Sample tools include Ansible, Puppet, PowerShell, Chef, Salt, or Otter.
* **Monitor.** This phase involves identifying and collecting information about issues from a specific software release in production. Sample tools include New Relic, Datadog, Grafana, Wireshark, Splunk, Nagios, or Slack.

|  |  |
| --- | --- |
| **DevOps tools by Stage** | |
| Plan | Jira |
| Code | GitHub, GitLab, Bitbucket, |
| Build | Docker, Gradle, Maven, or NPM |
| Test | JUnit, Selenium, TestNG, or BlazeMeter |
| Release | Jenkins, Bamboo, AWS Code pipeline |
| Deploy | Docker, AWS Cloud |
| Operate | Ansible, Chef, K8S |
| Monitor | Datadog, Grafana, Splunk, Nagios |

**DevOps Practices:**

DevOps practices reflect the idea of continuous improvement and automation. Many practices focus on one or more development cycle phases. These practices include:



* **Continuous Development.** This practice spans the planning and coding phases of the DevOps lifecycle. Version-control mechanisms might be involved.
* **Continuous Integration (CI).** This practice brings configuration management (CM) tools together with other test and development tools to track how much of the code being developed is ready for production. It involves rapid feedback between testing and development to quickly identify and resolve code issues.
* **Continuous Testing.** This practice incorporates automated, prescheduled, continued code tests as application code is being written or updated. Such tests can speed the delivery of code to production.
* **Continuous Delivery:**This practice automates the delivery of code changes, after testing, to a preproduction or staging environment. A staff member might then decide to promote such code changes into production.
* **Continuous Deployment (CD):** Similar to continuous delivery, this practice automates the release of new or changed code into production. A company doing continuous deployment might release code or feature changes several times per day. The use of container technologies, such as Docker and Kubernetes, can enable continuous deployment by helping to maintain consistency of the code across different deployment platforms and environments.
* **Continuous Feedback:** In the continuous feedback phase of the DevOps lifecycle, the evaluation of the effect of each release on the user experience takes place, and this evaluation is reported back to the team to improve the future releases.
* **Continuous Monitoring:** This practice involves ongoing monitoring of both the code in operation and the underlying infrastructure that supports it. A feedback loop that reports on bugs or issues then makes its way back to development.
* **Continuous Operations:** The continuous operations phase involves the reduction or elimination of planned downtime like scheduled maintenance. The goal of this phase is to increase the uptime or the time the users can use the application. Companies use container management systems like Kubernetes or Swarm in this phase.

**FAQs:**

1. What is DevOps?
2. What are all stages in DevOps Explain in Order?
3. What are tools used in different Phases in DevOps?
4. Explain 7C of DevOps?
5. What is Version Controlling?
6. What is CI / CD concept?
7. What is the difference between Continuous Deliver and Continuous Deployment?
8. Where to Store Artifacts or build files in real-time?
9. Which tools are used to analyse code quality?
10. How can we make our application independent of OS/Runtime?
11. How to handle Multiple / so many container in Real-time?
12. What are monitoring tools used?
13. What is IAC and Explain Ansible usages?
14. What are testing tools/ Frameworks used for testing applications?