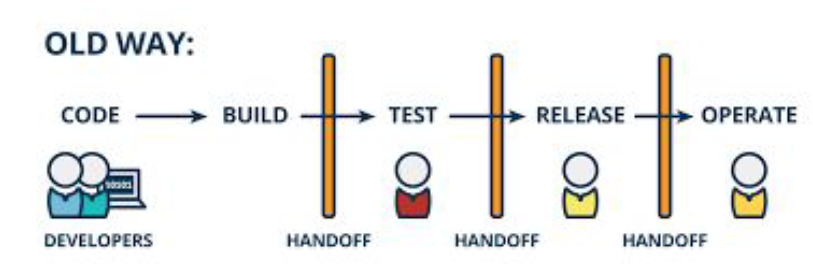
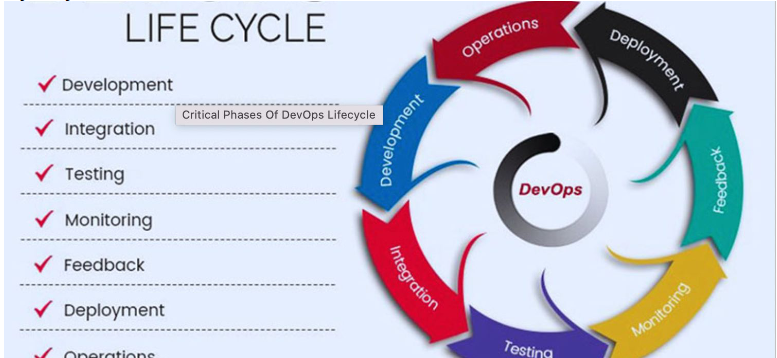
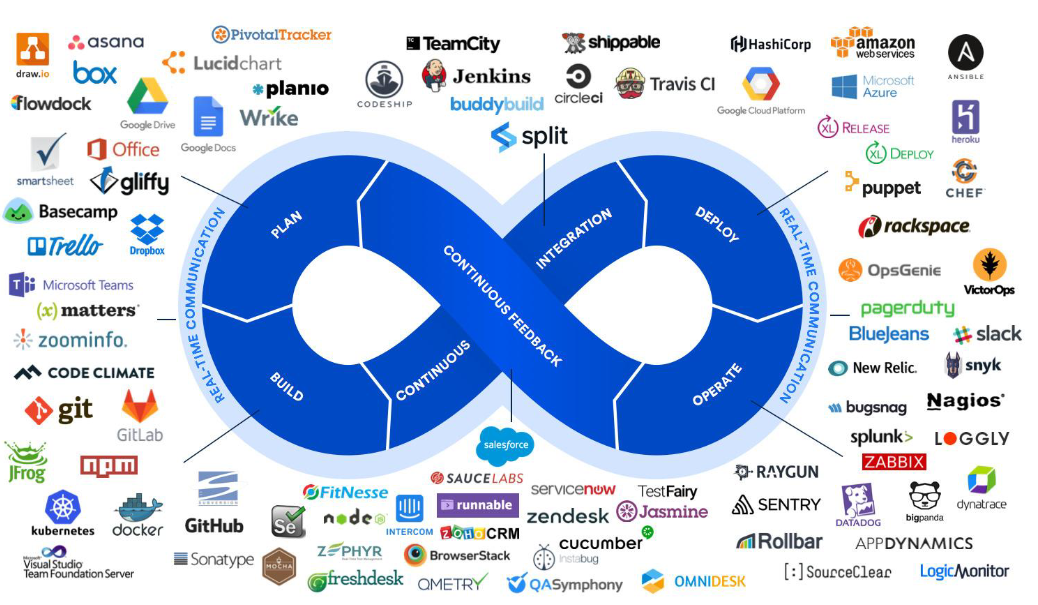


The Core Principles of  
DevOps  
DevOps revolves around a set of core principles that guide its implementation and practices. These principles  
include:  
•Collaboration – DevOps encourages open communication and collaboration between development and operations  
teams, fostering a culture of shared responsibility and accountability.  
•Automation – Automation is a cornerstone of DevOps. It streamlines processes, reduces human error, and allows for  
faster delivery and improved consistency.  
•Continuous Integration and Continuous Delivery (CI/CD) – CI/CD is the practice of automatically building, testing,  
and deploying software changes to production. This approach ensures a continuous flow of high-quality software  
releases.  
•Feedback and Monitoring – DevOps emphasizes the importance of monitoring and gathering feedback from both  
technical and business stakeholders to improve processes and continuously iterate on products.  
•Continuous Learning and Improvement – A commitment to continuous learning and improvement is essential in a  
DevOps culture. Teams should regularly evaluate their processes and tools to identify areas for growth and  
optimization. 

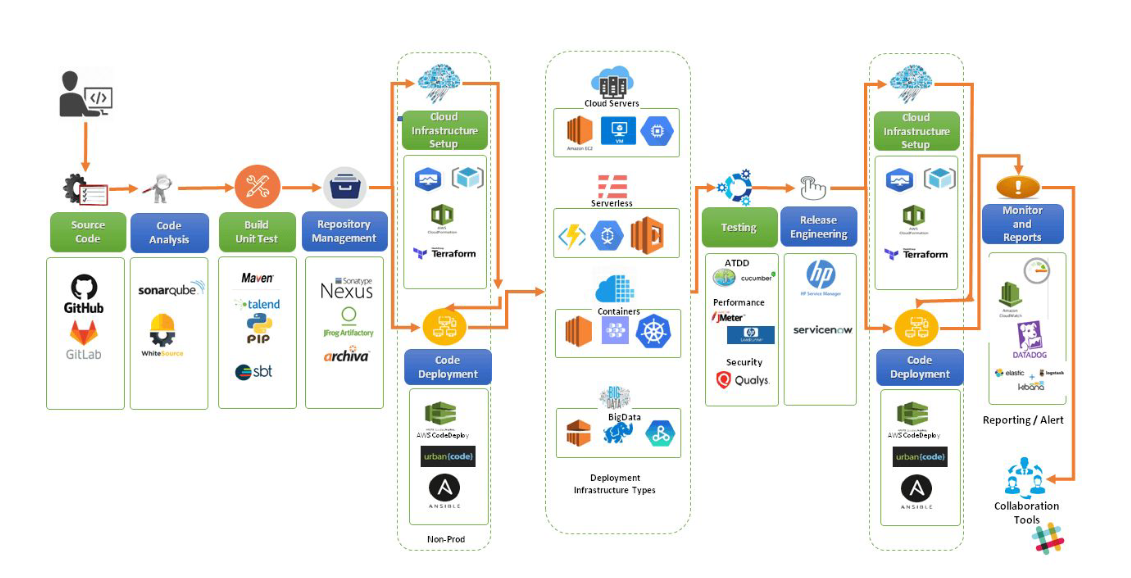
Benefits of Implementing DevOps in Your  
OrganizationAdopting a DevOps mindset can bring about significant benefits for your organization. These include:  
•Faster Delivery and Time-to-Market – DevOps practices enable quicker software releases, allowing your  
organization to respond more effectively to market changes and customer demands.  
•Improved Quality and Reliability – Through automation, continuous integration, and testing, DevOps helps  
ensure that the software delivered is of high quality, reliable, and secure.  
•Enhanced Collaboration and Communication – DevOps fosters a culture of open communication and  
collaboration, breaking down silos between teams and leading to better alignment, shared goals, and more  
efficient processes.  
•Cost Savings – By reducing the time spent on manual tasks and streamlining workflows, DevOps can lead to  
significant cost savings for your organization.  
•Increased Customer Satisfaction – With faster delivery and higher-quality products, your organization can better  
meet customer expectations, resulting in increased satisfaction and loyalty.



The Big Picture



The DevOps Flow



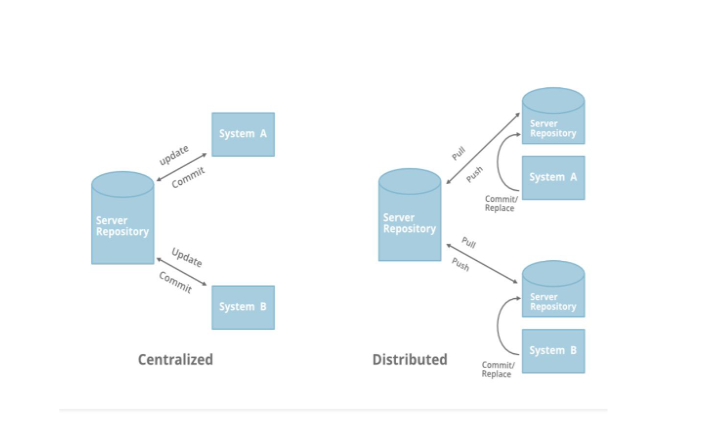
Git is not Github. Git is the version control software, and  
Github is a git repository hosting service which offers all  
the source code management provided in git. Github is  
where you upload your git repository.

Centralized Version Controlling

The systems such as CVS, Subversion, and Perforce  
have a single server that contains all the versioned  
files, and a number of clients that check out files  
from that central place

Distributed Version Controlling.

In a DVCS (such as Git), clients don’t just check out the latest snapshot of  
the files; rather, they fully mirror the repository, including its full history.  
Thus, if any server dies, and these systems were collaborating via that  
server, any of the client repositories can be copied back up to the server to  
restore it. Every clone is really a full backup of all the data



Three Stages in GIT

The Working Tree/untracked (current state of the project ) -- git status  
The Working Tree is the area where you are currently working.  
It is where your files live. This area is also known as the  
“untracked” area of git. Any changes to files will be marked and  
seen in the Working Tree

The Staging Area (Index) -- git add  
The Staging Area is when git starts tracking and saving changes  
reflect in that occur in files. These saved changes the .git  
directory. That is about it when it comes to the Staging Area.

Local Repository/Commit Region -- git commit  
The Local Repository is everything in your .git directory,  
add items from your Staging Area to your Local Repository

•git init → Create a new git repository  
•git add “newfile” → Add a new file to your staging  
area  
•git commit → Adds staged changes to your local  
repository  
•git push “remote” “ branch” → Push local repository  
changes to your hosting service  
•git pull “remote” “ branch” → pull code from your  
hosting service to your local directory  
•git branch → See local branches  
•git branch “newName” → Create new local branch  
•git checkout “branchName” → Switch branches  
•git diff → See the actual difference in code between  
your working tree and your staging area  
•git status → Show which files are being tracked v.  
untracked  
•git log → Show recent commit history  
•git show “commit\_id” → show details of specific  
commit  
•git stash → stash working directory  
•git help → manpages for git  
•git help “gitCommand” → man pages for specific git  
command