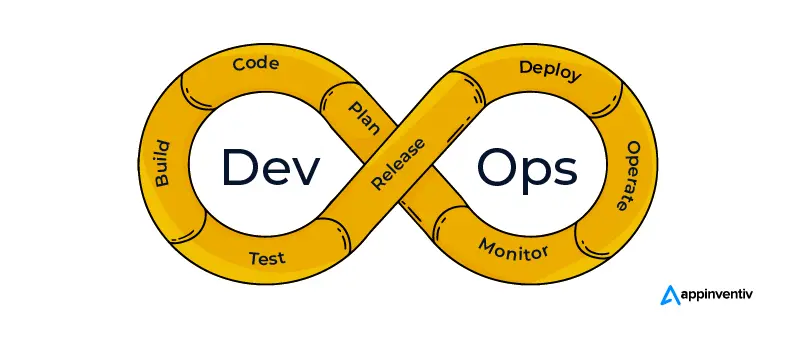
**DevOps adoption in projects**

**What is DevOps?**

DevOps is not a tool, technology or framework. Instead, it is a set of practices that help bridge the gap between development and operations teams in an enterprise. By bridging the gap, DevOps eliminates communication barriers and makes collaboration easier.

****

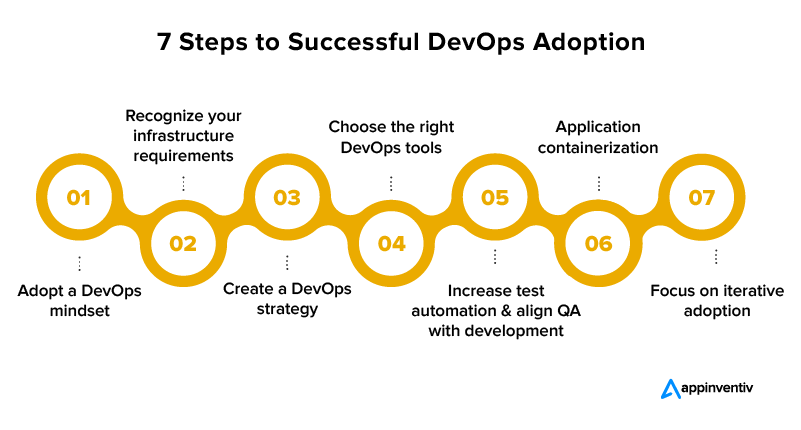
DevOps cannot be efficient without an agile setup. There is no point optimizing development and accelerating build processes if the new code will not go to the users until the next ‘big’ release.

**What is Agile DevOps?**

To bring two technologies together, we must understand both of them individually, which in turn, will help us understand how well they can gel with each other.

Both DevOps and Agile are the modern software development practices that are designed to produce a part of a product, launch, or release, the approaches they follow are different. Let us try and compare the approaches that the two practices follow:

[How Can DevOps and Agile Work Together to Help Your Business Grow? - Appinventiv](https://appinventiv.com/blog/agile-devops/)

**7 Steps to Successful DevOps Adoption**

Although the idea of DevOps is not new—it has been around for more than ten years—many firms have not yet put it into practice. And some organizations still have trouble using DevOps to get the results they want. Here are the steps that will help in the successful adoption of DevOps.

**Adopt a DevOps mindset**

Let’s implement DevOps. The process doesn’t just start by saying that. Everyone in your organization must be willing to change the way things are currently done and have a complete sense of what DevOps is and the specific business demands it may address.

Organizations frequently mix up automation and DevOps. Even while automation helps speed up manual operations, cooperation and communication are the key objectives of DevOps. Automating your operations won’t bring about the desired business benefits unless everyone involved in the software development, delivery, testing, and operating processes adopts excellent communication and collaborative practices.

The best way to implement DevOps effectively is to make sure that everyone involved in the delivery cycle is more flexible and has an innovative mentality.

Everyone participating in the process should be aware of their duties and responsibilities and trained to cooperate for DevOps to become the organization’s culture. For DevOps to succeed, the organization’s leadership must have confidence in it and must assist in fostering a DevOps culture.

**Recognize your infrastructure requirements**

There is no “one size fits all” DevOps solution, despite what those who offer DevOps solutions will tell you. You can’t merely hire a self-described “DevOps engineer” or toss in an online tool and expect success.

Each organization’s DevOps journey will be distinct and based on its own business, culture, and infrastructure. The crucial next step is to have a deeper grasp of your application’s requirements. It enables you to make DevOps adoption business-driven and match infrastructure architecture with your organizational goals.

Evaluate your project delivery cycle and testing environments to find areas for improvement and possible bottlenecks.

Your DevOps adoption won’t be successful without integrating Continuous Integration and Continuous Delivery (CI/CD) pipelines into your workflow. Why? Because Continuous Delivery enables your development teams to deploy changes in production, and Continuous Integration helps them develop a product in small phases and identify and rectify faults instantly.

**Create a DevOps strategy**

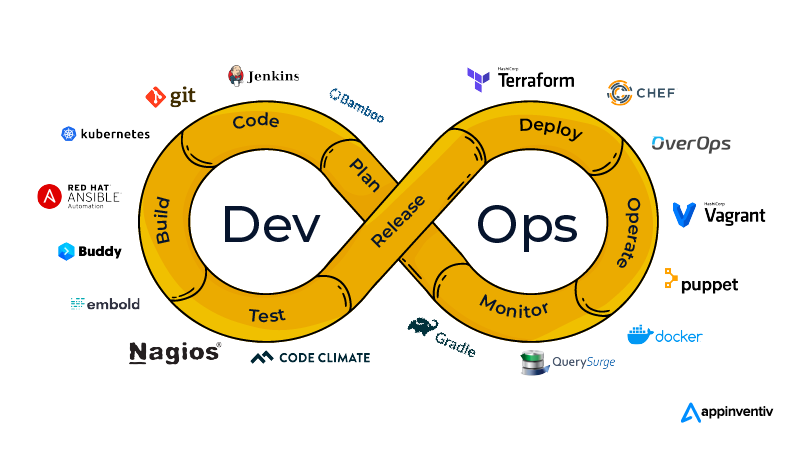
Program managers must establish a shared objective to bring teams together in a collaborative setting. It instills a sense of responsibility and obligation in each team member. DevOps relies heavily on best practices that promote innovative approaches to software development, architecture, and testing while enhancing teamwork.

Your strategy should be focused on two objectives: helping the team as a whole do its work to the best of its ability and facilitating the continuous deployment of processes that are ready for production.

**Choose the right DevOps tools**

There isn’t a single tool that can handle all of the demands and key purposes of DevOps. The best course of action is to select a collection of tools that are ideal for the organization’s software delivery environment, applications, and teams.

The appropriate tools help organizations establish a solid DevOps framework, accomplish a continuous process from development to delivery, aid in resource and cost optimization, support seamless process execution, and ultimately fulfill organizational goals.

****

Organizations must take the following considerations into account when selecting the appropriate DevOps tools:

The tools ought to be capable of enterprise-level automation. Without adding more effort, it will assist in scaling business workflows and continuously improving the operations.

Integrating the entire delivery ecosystem is required in DevOps. Consequently, the tools you select should have integration capabilities.

**Increase test automation and align QA with development**

DevOps requires appropriate automated testing in order to achieve faster delivery. Not all testing types need to be automated. For instance, manual testing should still be done for investigative, security, and usability testing. Functional testing may only be partially automated, depending on the amount of writing effort required.

Development and testing are done simultaneously to prevent bugs after a release. The recommended approach is to run automated tests 1-2 times per day while the program is still being developed. If any issues are discovered, developers can concentrate on stabilizing the software before deploying the latest build.

**Application containerization**

Application containerization is a rapidly developing technology that is altering how cloud-based application instances are tested and run by developers. Your programs become lightweight and simple to execute when you containerize them.

As software is used, its reliability is increased by container packaging. Additionally, the software is independent of the broader infrastructure, thanks to its container components. This improves its ability to operate independently in any context. Furthermore, containerizing enables DevOps teams to quickly manage the application and make any adjustments required for a specific microservice.

**Focus on iterative adoption**

Avoid attempting to launch a comprehensive DevOps in the enterprise while just getting started. Choose a pilot application, put together a cross-functional DevOps team made up of developers, testers, and operations personnel, assess your value stream to discover bottlenecks and restrictions, and develop a preliminary deployment pipeline that takes a few of your processes constraints into account.

Measure your success and growth, then repeat the process. Before starting to expand to additional projects, you must go through a few iterations to gain trust in the framework and the pilot.

Generally, since doing so would have the greatest commercial impact, you should start by addressing your largest value-stream restrictions. Some of these restrictions will be simple to overcome, while others will require a lot of time.