What Is DevOps?

DevOps is a set of practices, tools, and a cultural philosophy that automate and integrate the processes between software development and IT teams. It emphasizes team empowerment, cross-team communication and collaboration, and technology automation.
The DevOps movement began around 2007 when the software development and IT operations communities raised concerns about the traditional software development model, where developers who wrote code worked apart from operations who deployed and supported the code. The term DevOps, a combination of the words development and operations, reflects the process of integrating these disciplines into one, continuous process.

How does DevOps work?

A DevOps team includes developers and IT operations working collaboratively throughout the product lifecycle, in order to increase the speed and quality of software deployment. It's a new way of working, a cultural shift, that has significant implications for teams and the organizations they work for.

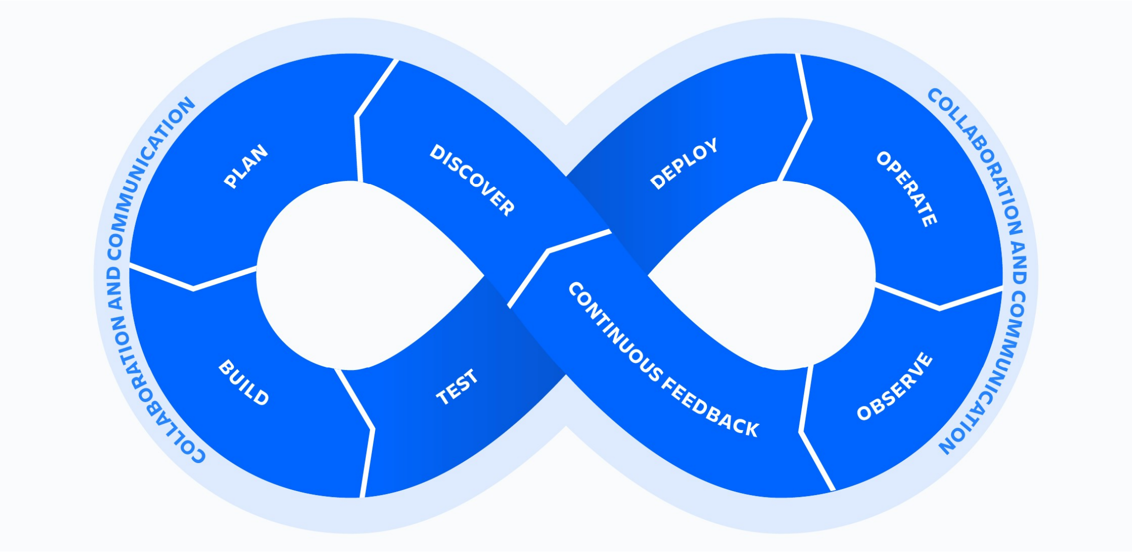
Under a DevOps model, development and operations teams are no longer "siloed." Sometimes, these two teams merge into a single team where the engineers work across the entire application lifecycle — from development and test to deployment and operations — and have a range of multidisciplinary skills.

DevOps teams use tools to automate and accelerate processes, which helps to increase reliability. A DevOps toolchain helps teams tackle important DevOps fundamentals including continuous integration, continuous delivery, automation, and collaboration.

DevOps values are sometimes applied to teams other than development. When security teams adopt a DevOps approach, security is an active and integrated part of the development process. This is called DevSecOps.

The DevOps lifecycle

Because of the continuous nature of DevOps, practitioners use the infinity loop to show how the phases of the DevOps lifecycle relate to each other. Despite appearing to flow sequentially, the loop symbolizes the need for constant collaboration and iterative improvement throughout the entire lifecycle.


COLLABORATION AND COMMUNICATION
PLAN
DISCOVER
BUILD
TEST
DEPLOY
OPERATE
CONTINUOUS FEEDBACK
OBSERVE
COLLABORATION AND COMMUNICATION

What is a Full Stack Engineer?

Full stack technology refers to the entire depth of a computer system application, and full stack developers straddle two separate web development domains: the front end and the back end.

The front end includes everything that a client, or site viewer, can see and interact with. By contrast, the back end refers to all the servers, databases, and other internal architecture that drives the application; usually, the end-user never interacts with this realm directly.

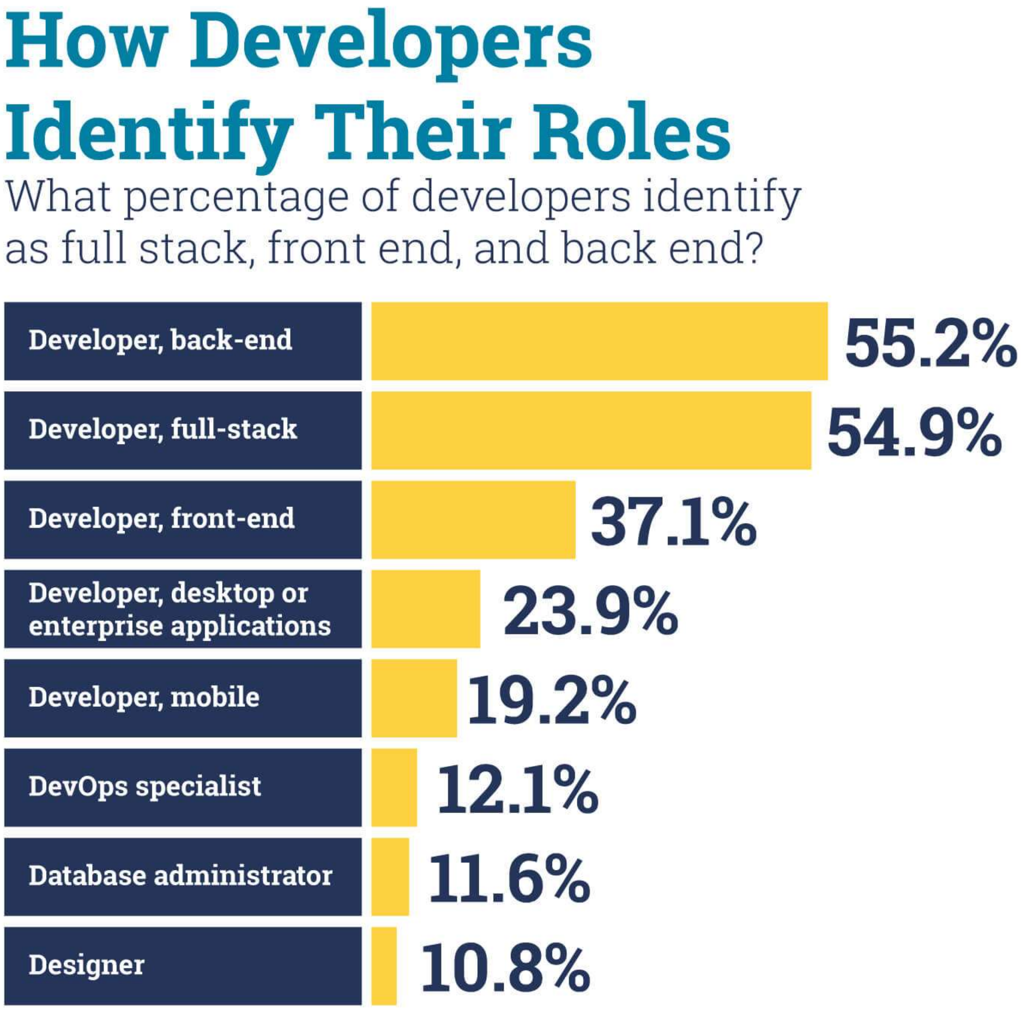
The easiest way to put the full stack into perspective is to imagine a restaurant. The front end encompasses the well-decorated, comfortable seating areas where visitors enjoy their food. The kitchen and pantry make up the “back end” and are typically hidden away from the customer’s view. Chefs (developers) gather permanently stored materials from the pantry (the database) and perform operations on it in the kitchen (the server), and then serve up fully-prepared meals (information) to the user.

Front end developers work to optimize the visible parts of an application for web browsers and mobile devices. Front end platforms are usually built with HTML, CSS, and JavaScript; however, they can also be made via pre-packaged code libraries or content management systems like WordPress. Back end developers, in contrast, refine the software code that communicates with servers, databases, or other proprietary software that conveys information to front end interfaces.

Those knowledgeable in both front end and back end are called full stack developers, meaning they are well versed in both disciplines.

The term “full stack developer” originated during the early days of the web, when websites were small and uncomplicated enough to allow a single person to tackle every aspect of site-building. But in the decades since those initial days, the web has grown ever more complex. The rise of machine learning, predictive computing, and responsive design has made it challenging — but not impossible! — for a single developer to handle every aspect of building and designing a site or application.

Today, modern businesses often rely on entire *teams* of developers to operate network equipment, work with virtual machines, and manage enormous databases. It takes time to develop a comprehensive, nuts-and-bolts understanding of *all* these emerging technologies. The developers who do so are, for that reason, versatile enough to shift fluidly between front and back end development and take on any task that their team might need them to tackle.



StackOverflow, "2020 Developer Survey." (2020)
https://insights.stackoverflow.com/survey/2020#developer-roles

The modern full stack developer is an experienced generalist who can build a minimal viable product — i.e., an application with enough functionality to please early customers and spark feedback for continued development — on their own. Companies rely on full stack professionals to spot errors between the front and back end and tackle tasks that straddle both disciplines. This versatility has become increasingly vital as newer apps begin to incorporate AI and other sophisticated technologies into their programming.

These days, it's mission-critical to have at least one person on a given development team who has a passing understanding of all the components that run an enterprise-level application. And companies understand this need. According to statistics provided by the Government of Canada's Job Bank, new job openings for software developers are expected to top 27,500 between 2019