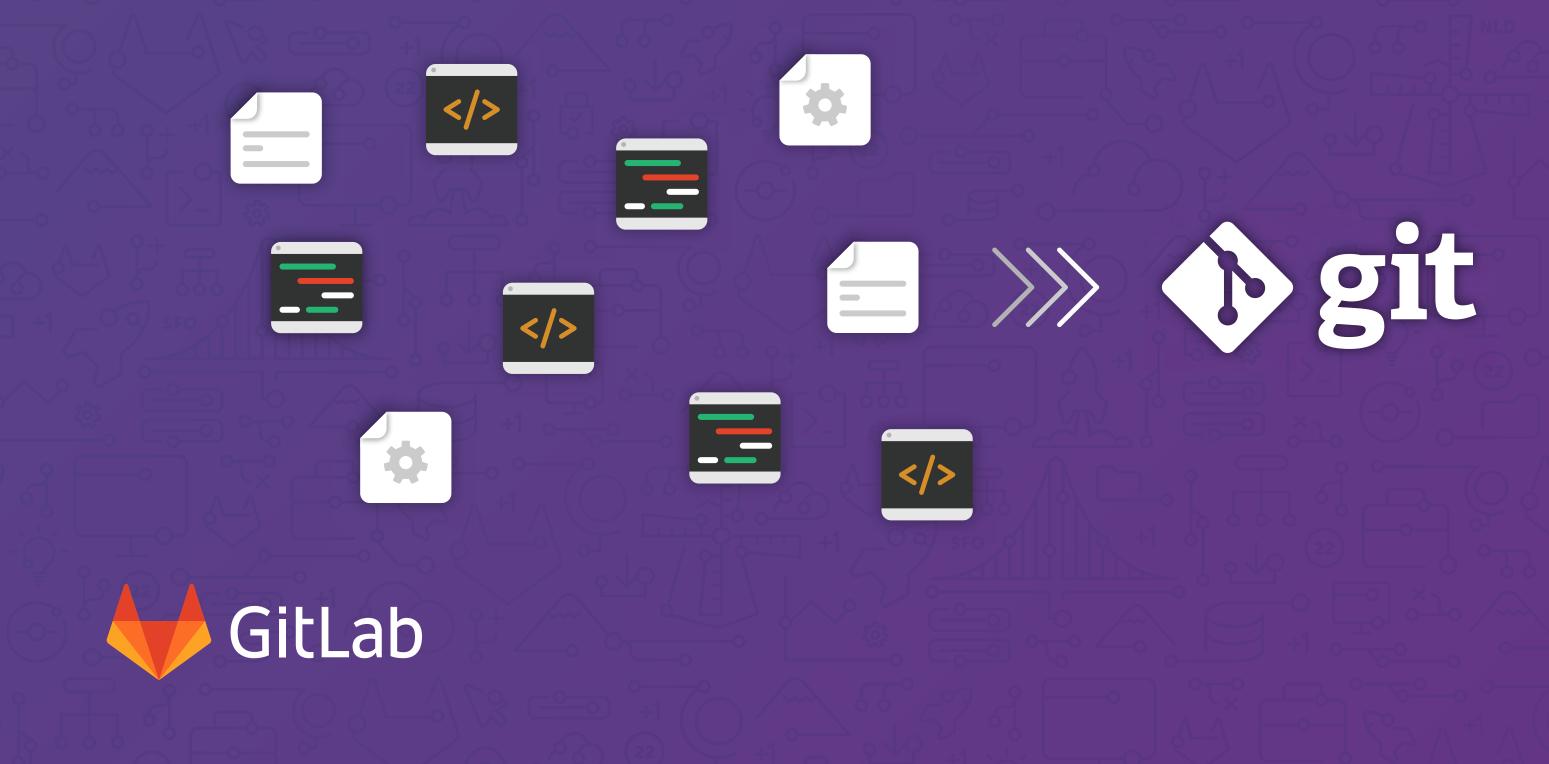
MOVING TO GIT

IMPLEMENT GIT TO IMPROVE COLLABORATION AND SHIP FASTER



WHAT'S INSIDE?

BUSINESS ACCELERATED BY SOFTWARE

» Modern Development, Modern Tools

FROM CENTRALIZED TO DISTRIBUTED VERSION CONTROL

» Looking Back: Centralized Version Control Systems

» Looking Forward: Distributed Version Control Systems

WHY GIT?

- » Collaboration & Code Quality
- » Cheaper Branching
- » Easier Merging
- » Reduced Costs

CHOOSING A GIT MANAGEMENT SOLUTION

- » What to consider when choosing a Git management solution
- » Integration with other tools
- » Cost

GETTING STARTED

ABOUT GITLAB



BUSINESS ACCELERATED BY SOFTWARE

Technology has the power to change the world. Whether it's software for a critical medical device or an app that can detect your running speed, technology is constantly changing the way we interact with the world around us. This transformation in the way people live and communicate has had a profound effect on the way people interact with brands, both personally and professionally.

"Software is eating the world¹," and transforming business in its wake. In a world where even books are digital, building great, reliable software is core to every business. Long-established brands are digitizing to mitigate risk of disruption. Innovation and speed are a universal competitive advantage. The ability to release often and quickly respond to customer feedback has a direct impact on a brand's ability to stay relevant. With all this rapid change, business executives must ask themselves, *"How can we phase out legacy software, keep up with the pace and get ahead of the curve?"*

Today's software landscape demands an upgrade.

¹ Why Software Is Eating the World. Andreessen, Marc, a16z.com



MODERN DEVELOPMENT, MODERN TOOLS

Modern software is built differently. Lean, flexible, fast, and ready for the cloud, modern software operations are regularly capitalizing on the newest technologies and finding innovative solutions to delivering customer value fast and efficiently.

To meet the expectations of a digitally transformed economy, more enterprise businesses are evolving to adopt modern tools. The International Data Corporation (IDC) has estimated that by the end of 2017, two-thirds of the CEOs of Global 2000 Enterprises will have digital transformation at the center of their corporate strategy, and that by 2020, more than 30 percent of legacy IT vendors will not exist².

By 2020, more than 30% of legacy IT vendors will not exist

Why the dramatic shift?

As we've learned from some of the earliest disruptive companies (Amazon, Netflix, Facebook), **business is software**. This means that a company's

development and IT functions must adapt and respond to the needs of the business, including revenue targets and customer engagement and satisfaction.

To keep up with this digital transformation, software development teams have to change how they work — from shifting from monolith applications to microservices and releasing frequently to agile development and DevOps. Unfortunately, the legacy systems of the past simply were not built to support the collaborative and distributed workflows the modern developer needs.



Upgrading your source code management solution from a centralized system to a distributed version control system (DVCS) is the first step toward building a flexible working environment that can support modern development teams.

Modern Software Development
Frequent releases
Cross-functional teams
Tool standardization
Infrastructure as code
Automated test & release
Microservices



FROM CENTRALIZED TO DISTRIBUTED VERSION CONTROL

Over the last decade, distributed version control systems, like Git, have gained popularity and are regarded as the most important development tools by developers. Distributed version control systems are different from centralized systems because they don't rely on a central server. Instead, DVC systems are decentralized, meaning a complete copy and full history of a codebase is stored in multiple repositories.

> 92% of developers say a distributed version control system, like Git, is the most important tool for their day-to-day work.

> > - 2016 GLOBAL DEVELOPER SURVEY

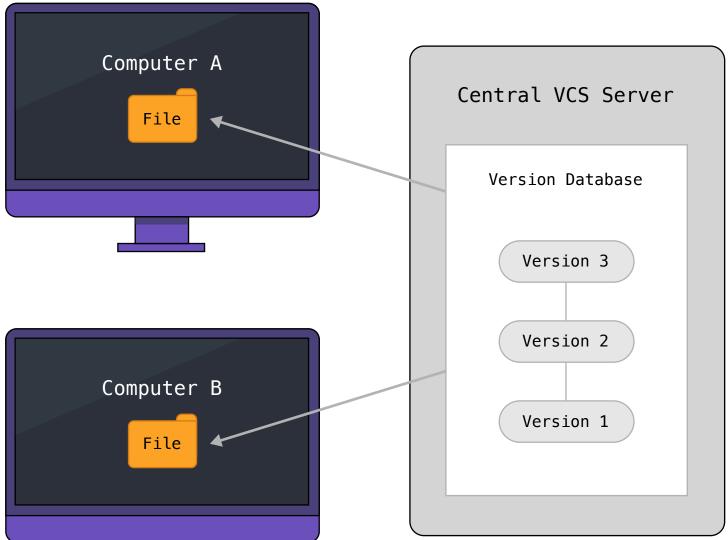


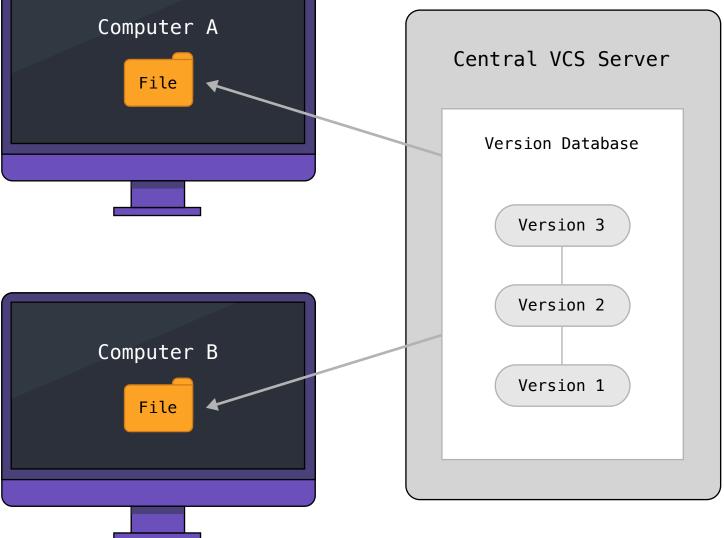
LOOKING BACK: CENTRALIZED VERSION CONTROL SYSTEMS

Using a traditional, centralized model, development workflows must work with a centralized version control or "legacy system" which prevents teams from meeting the demands of releasing often and responding quickly to customer feedback.

With a centralized version control system, almost all actions require access to a central repository, usually located on a remote server. In order to see a history of changes, create a tag, make a new branch and so on, developers require a connection to that server. The drawback of this for enterprises is obvious: that single, central server becomes a bottleneck as teams and projects grow, blocking progress.

Although it can be challenging to migrate from a legacy system, the benefits outweigh the costs, and you only have to do it once to reap the rewards. It's a decision that can drastically improve your team's culture, productivity, and modes of collaboration.



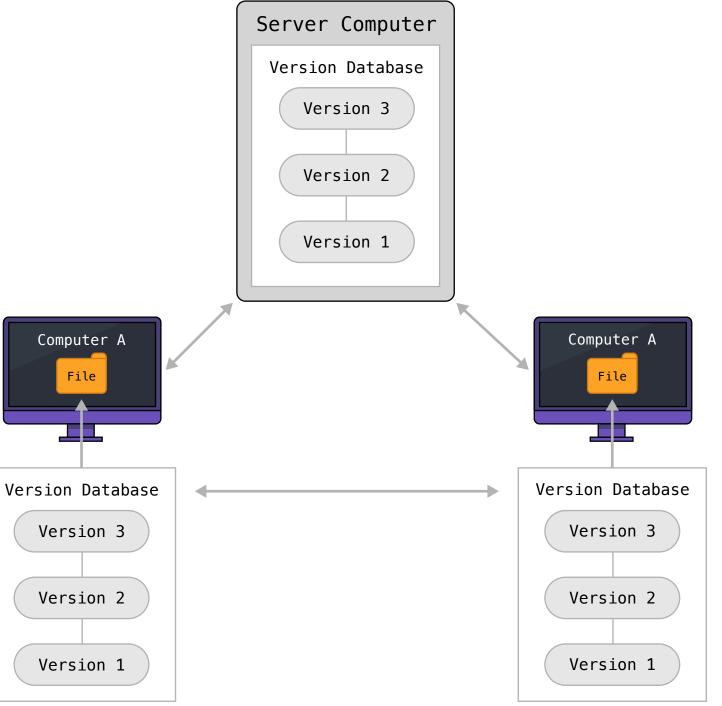




LOOKING FORWARD: DISTRIBUTED VERSION CONTROL SYSTEMS

Distributed version control makes it easier to share, test, and merge changes while maintaining a clean master version of your code. Using a DVCS, every developer has the full history of the repository on their own machine.

With a clone of the repository stored locally, commits, merges and other commands are much faster, with developers even able to work offline. But speed and convenience aren't the only advantages of using a DVCS: once it's established you'll enjoy the benefits of improved collaboration, cheaper branching, easier merging, a better workflow, and reduced costs, too.





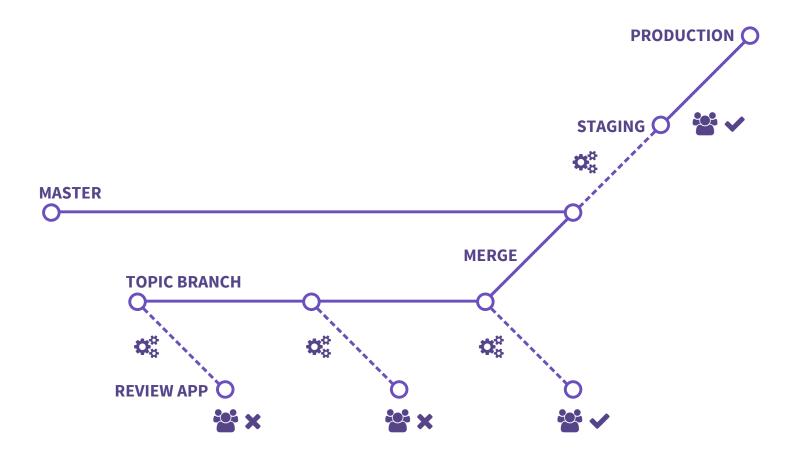
WHY GIT?

Git is an open source, distributed version control system. It is purpose-built for distributed development models, offering strong support for distributed and remote teams, and non-linear development practices.

Since its inception in 2005, Git has grown to be the most popular version control system for developers — 70 percent of the 64,000 developers who responded to <u>Stack Overflow's 2017 Developer Survey</u> are using Git, compared to only 9 percent who say they still use Subversion. <u>Another study done by RhodeCode</u> validates the claim that Git is the version control system of choice, claiming:

- » Better collaboration & code quality
- » Cheaper branching
- » Easier merging
- » Reduced costs

Git is open source: it's free, secure, flexible and it has a large community maintaining a high standard of quality. A study done by Black Duck Software revealed that 78 percent of companies run at least part of their operations on open source software and **55 percent say open source delivers superior security**³. Enabling safe and reliable distributed development workflows at a low cost, Git greatly improves team collaboration and can accelerate deployments and release cycles. Adopting Git gives software development teams the competitive edge they need in today's hyper-digitalized and ever-transforming world.





Collaboration and Code Quality

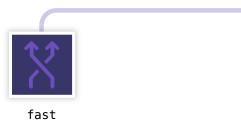
Git is the first VCS to enable a truly modern workflow, making it easy and fast to create, merge, and manage many branches. <u>Feature branches</u>, or branches containing one feature or bug fix for their project, allow teams to iterate, discuss, and perfect their code at the merge request stage, before their changes are accepted into master. Git management solutions, like GitLab, encourage this modern workflow.

By updating each other early and often, this workflow enables teams to collect feedback regularly and integrate suggestions throughout the process.

Cheaper Branching

Git's branching capabilities are one of the main reasons why Git has become so popular. Developers can create a new branch and isolated environment to work on new features, bug fixes, customer requests, or even just experiment on something new at little to no cost.

Branching with Git is "cheap" because it is quick, easy, and doesn't take up a lot of space. Creating a new branch only requires a single command, and because it's distributed, no network connection needed—unlike Subversion and other centralized version control systems. In older version control systems, branching requires making a complete copy of all the code, making branching slow and repetitive — risking multiple versions of the same software.



Easier Merging

Because of its cheap branching abilities, Git encourages developers to make smaller commits, more often, making merging a lot easier and less scary. Before merging to master, a developer can commit their code locally to ensure it works, preserving the quality of the master code base.

When coupled with a source code management tool like GitLab, you can enhance merging capabilities with a user interface that allows you to review and comment on changes on branches before merging. At GitLab, we call this a <u>merge request</u> and run continuous integration on every branch as an additional quality gate to ensure everything that goes into the master code base works. With GitLab branching and merging, you can even protect branches to prevent merges that aren't ready, and apply merge request approvals for an added layer of security.

Reduced costs

Git on its own is open source and completely free. This also means you benefit from the contributions of the open source community working on bug fixes and feature improvements. Even if you choose a Git management solution (see below), these are often a fraction of the price of licenses per user and the cost of maintenance for legacy systems.



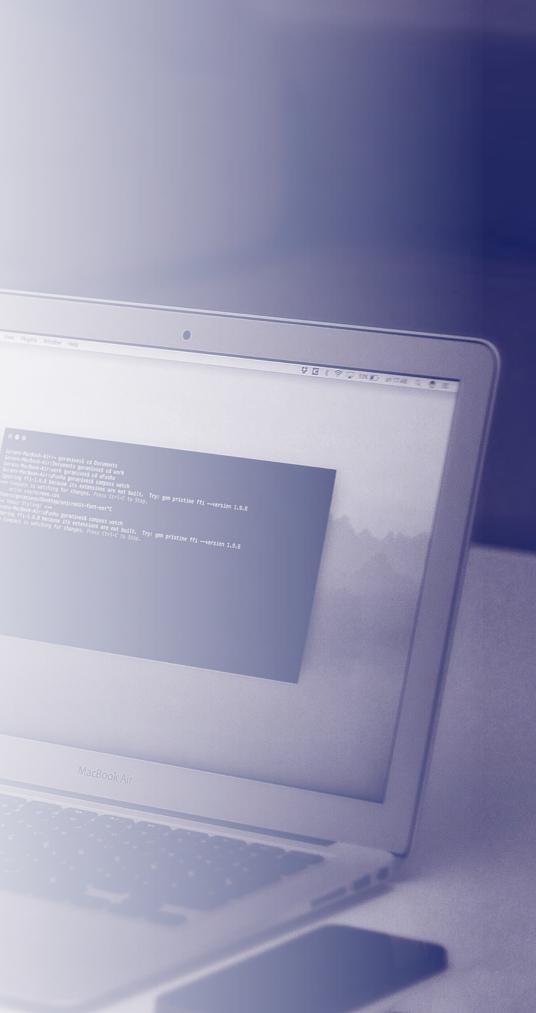




CHOOSING A GIT MANAGEMENT SOLUTION

One of the cited drawbacks of Git is the lack of graphical user interface. However, thanks to the open source community, there are a variety of management solutions that are built on top of Git, like <u>GitLab</u>. With a Git management solution, you can dramatically improve the user experience of Git. Git management solutions offer a user interface for visualizing, managing, and discovering your projects and repositories, access and security management, and a UI for reviewing code changes. Working within the UI allows non-technical team members to join the conversation also, so they can get to work on promoting your next release or gathering feedback from customers.

A Git management solution can also offer enterprise-ready features that are designed to simplify and secure the software development lifecycle for larger organizations, such as time tracking, file locking, CI/CD and merge or pull request approvals. Another advantage is the ability to integrate with modern technology and processes to make the most of other tools and container schedulers such as Kubernetes.



WHAT TO CONSIDER WHEN CHOOSING A GIT MANAGEMENT SOLUTION

Hosting

Installing and customizing software on your own machines in your own data center (on-premise) has several advantages. Firstly, you have full control over company systems and the security of knowing everything is housed within the walls of your own business. By not relying on a hosted service, you aren't affected by their downtime which means no interruption to your own service.

The disadvantage of on-premise is that this places the burden of managing and maintaining in-house servers on your organization, and you'll need a dedicated IT team for that, otherwise you may find your customers growing frustrated with your own downtime anyway.

Opting for a hosted service comes with benefits too: released from the responsibility of managing and maintaining servers, you free up resources to focus on delivering great service and a better product to your customers. The risk here is that any downtime or data loss experienced by your cloud service provider will impact on your own service. It's up to you to weigh up which option makes the most of your organization's skills and resources.

Integration with Other Tools

Source code management is just one element of the software development lifecycle. Your teams will most likely want other features such as issue tracking, <u>CI/CD</u>, code review and collaboration tools. When choosing a Git management solution it's important to find out first what other tools or features your teams require, and whether the solution you're considering includes or supports those – some solutions have built-in features while others offer integration with popular tools. Whatever your preference, establish your team's needs up front to avoid being caught off guard when the solution you've chosen doesn't support an integral component of their workflow.

Cost

What is your budget and how do you want to spend it? Opting for an open source solution rather than a commercial one can seem like a shortcut to savings, and many organizations choose open source for this very reason. It's also critical though to consider whether the features that are included and support offered are sufficient for your needs, and will scale as your organization grows. Some enterprise owners find it reassuring to have access to priority support should they need it, or special features required for larger organizations (such as approvals, file locking and repository mirroring).



GETTING STARTED

The most important thing to remember when migrating from a centralized system to Git is that it requires a new way of thinking about version control, and the old way of doing things simply will not fit into your new system. Luckily, the versatility of Git allows for multiple types of workflows, so you and your team can decide what's best for you.

The initial migration to Git can a couple of days or a couple of months depending on your migration strategy. To reduce complexity, a complete transition is recommended. However, for organizations with many teams and projects, you can also start by migrating one team or project at a time.

Getting started with Git doesn't have to be intimidating. There are lots of tools out there that can help with the migration and it's relatively simple to <u>get your</u> <u>team up and running on GitLab</u>.

Have questions or need more information?

We've got you covered! <u>Get in touch</u>.



ABOUT GITLAB

Since its founding in 2014, GitLab has quickly become the leading self-hosted Git repository management tool used by software development teams ranging from startups to global enterprise organizations. GitLab has since expanded its product offering to deliver an integrated enterprise-grade source code management system with built-in continuous integration and continuous delivery tooling.

GitLab helps teams eliminate unnecessary steps from their workflow and focus exclusively on building great software. Today, more than 100,000 organizations, including NASA, CERN, Alibaba, SpaceX, O'Reilly, IBM and ING, trust GitLab to bring their modern applications from idea to production, reliably and repeatedly.

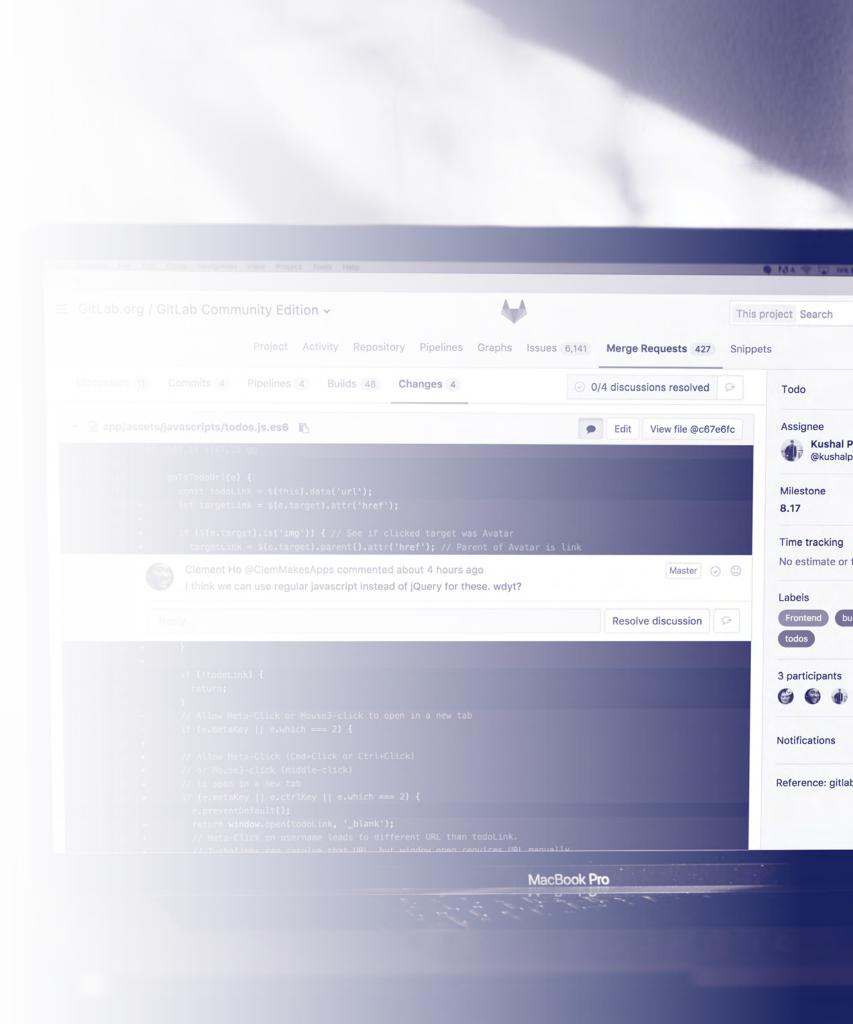
Ready to get started?

Start a free GitLab Enterprise Edition trial today!

Start Your Free Trial

"It made a big difference for us to have a central point for software projects that we could manage by ourselves and have all the tooling in one place. I think using GitLab has also led to improvements in our workflow and collaboration."

> – MIKAEL VAPPULA, IT MANAGER AT VAADIN <u>WHY VAADIN CHOSE GITLAB</u>



GitLab

