ElectricFlow vs GitLab

GitLab compared to other DevOps tools

Electric Cloud ElectricFlow is a platform which provides deployment automation, release orchestration, and DevOps insights to help organizations deliver better software faster. The base platform (formerly known as Electric Commander) is used by many organizations to automate their CI/CD pipelines.

Although Electric Cloud claims complete end to end DevOps, the platform requires a lot of integration to other tools in the tool chain in order to supplement functionality, as do just about all CI/CD point tools. In contrast, GitLab come pre-integrated with fundamental and extended functionality built-in across the DevOps lifecycle. An example is with security tools, where other CI/CD vendors such as Electric Cloud claim DevSecOps, they merely integrate to 3rd party security tools and maybe provide a dashboard. GitLab comes with many security scanning capabilities built-in

FEATURES		
Environments and deployments GitLab CI is capable of not only testing or building your projects, but also deploying them in your infrastructure, with the added benefit of giving you a way to track your deployments. Environments are like tags for your CI jobs, describing where code gets deployed. Learn more about environments	✓	✓
Per-environment permissions Developers and QA can deploy to their own environments on demand while production stays locked down. Build engineers and ops teams spend less time servicing deploy requests, and can gate what goes into production. Learn about protected branches in GitLab	✓	✗
Environments history Environments history allows you to see what is currently being deployed on your servers, and to access a detailed view for all the past deployments. From this list you can also re-deploy the current version, or even rollback an old stable one in case something went wrong. Learn more about history of an environment	✓	✓
Environment-specific variables Limit the environment scope of a variable by defining which environments it can be available for. Learn how to configure environment-specific variables	✓	✓

Group-level variables Define variables at the group level and use them in any project in the group. Learn how to configure variables	✓	✓
Object storage for artifacts Artifacts can be stored on Object Storage (Amazon S3) Learn how to store artifacts on object storage	✓	✓
Run CI/CD jobs on Windows GitLab Runner supports Windows and can run jobs natively on this platform. You can automatically build, test, and deploy Windows-based projects by leveraging PowerShell or batch files. Install GitLab Runner on Windows	⊖	✓
Run CI/CD jobs on macOS GitLab Runner supports macOS and can run jobs natively on this platform. You can automatically build, test, and deploy for macOS based projects by leveraging shell scripts and command line tools. Install GitLab Runner on macOS	✓	✓
Run CI/CD jobs on Linux ARM GitLab Runner supports Linux operating systems on ARM architectures and can run jobs natively on this platform. You can automatically build, test, and deploy for Linux ARM based projects by leveraging shell scripts and command line tools. Install GitLab Runner on Linux	⊖	✓
Run CI/CD jobs on FreeBSD GitLab Runner supports FreeBSD and can run jobs natively on this platform. You can automatically build, test, and deploy for FreeBSD-based projects by leveraging shell scripts and command line tools. Install GitLab Runner on FreeBSD	⊖	✓

Manage JUnit reports created by CI jobs

Many languages use frameworks that automatically run tests on your code and create a report: one example is the JUnit format that is common to different tools. GitLab supports browsing artifacts and you can download reports, but we're still working on a proper way to integrate them directly into the product.

Read more on the issue

✓ ✗

Details on duration for each command execution in GitLab CI/CD

Other CI systems show execution time for each single command run in CI jobs, not just the overall time. We're reconsidering how job output logs are managed in order to add this feature as well.

Read more on the issue

✓ ✗

Auto DevOps

Auto DevOps brings DevOps best practices to your project by automatically configuring software development lifecycles by default. It automatically detects, builds, tests, deploys, and monitors applications.

Read more about Auto DevOps in the documentation

✗ ✓

Protected Runners

Protected Runners allow you to protect your sensitive information, for example deployment credentials, by allowing only jobs running on protected branches to access them.

Read more on the issue

✓ ✓

Globally distributed cloning with GitLab Geo

When development teams are spread across two or more geographical locations, but their GitLab instance is in a single location, fetching and cloning large repositories can take a long time. Built for distributed teams, GitLab Geo allows for read-only mirrors of your GitLab instance, reducing the time it takes to clone and fetch large repos and improving your collaboration process.

Learn more about GitLab Geo

✓ ✓

Support for High Availability

To avoid downtime, GitLab Enterprise Edition Premium offers support for High Availability (HA). A Service Engineer will help you identify your specific HA needs and map out an architecture.

Learn more about GitLab's High availability

✓ ✓

Deploy Boards GitLab Premium ships with Deploy Boards offering a consolidated view of the current health and status of each CI/CD environment running on Kubernetes. The status of each pod of your latest deployment is displayed seamlessly within GitLab without the need to access Kubernetes. Learn more about Deploy Boards	⊖	✓
Incremental rollout deployments GitLab can allow you to deploy a new version of your app on Kubernetes starting with just a few pods, and then increase the percentage if everything is working fine. Learn more about configuring incremental rollout deployments	✓	✓
Canary Deployments GitLab Enterprise Edition Premium can monitor your Canary Deployments when deploying your applications with Kubernetes. Learn more about configuring Canary Deployments	✓	✓
Minimal CI/CD configuration GitLab CI/CD requires less configuration for your pipelines than other similar setups like Jenkins. Learn more about GitLab CI/CD	✗	✓
Multiple integrations GitLab can integrate with Authentication and Authorization (LDAP / AD) mechanisms, multiple 3rd party services, CI/CD, and other tools such as ALM, PLM, Agile and Automation tools. Learn more about GitLab's integrations	✓	✓
Easy upgrade process Using our official Linux repositories or the official Docker image, upgrading GitLab is a breeze. Learn how to upgrade your GitLab instance	✗	✓
Community based, users can help shape the product GitLab has open issue trackers for almost all of its operations. From GitLab itself to infrastructure and marketing, you can help shape the product. View all GitLab contributors	✗	✓

Kubernetes Cluster Monitoring

Monitor key metrics of your connected Kubernetes cluster.

✗ ✓

Learn more about Cluster Monitoring

ChatOps

Execute common actions directly from chat, with the output sent back to the channel.

✗ ✓

Learn more about ChatOps

Enforced Two-factor Authentication (2FA)

Two-factor authentication secures your account by requiring a second confirmation, in addition to your password. That second step means your account stays secure even if your password is compromised. The ability to enforce 2FA provides further security by making sure all users are using it.

✗ ✓

Learn more about Enforced GitLab 2FA

IP Whitelisting

IP Whitelisting defines safe IP network addresses from which clients can access and interact with the repository server. This helps prevent unwanted third parties from accessing your account even if they have acquired a team memberâ€™s email address and password.

✗ ✓

Learn more about GitLab IP Whitelisting

Domain Specific Lanuage

A Domain Specific Lanuage (DSL) for defining infrstructure configuration allows thinking in resources, not files or commands to write declarative rather then procedural code.

✓ ✗