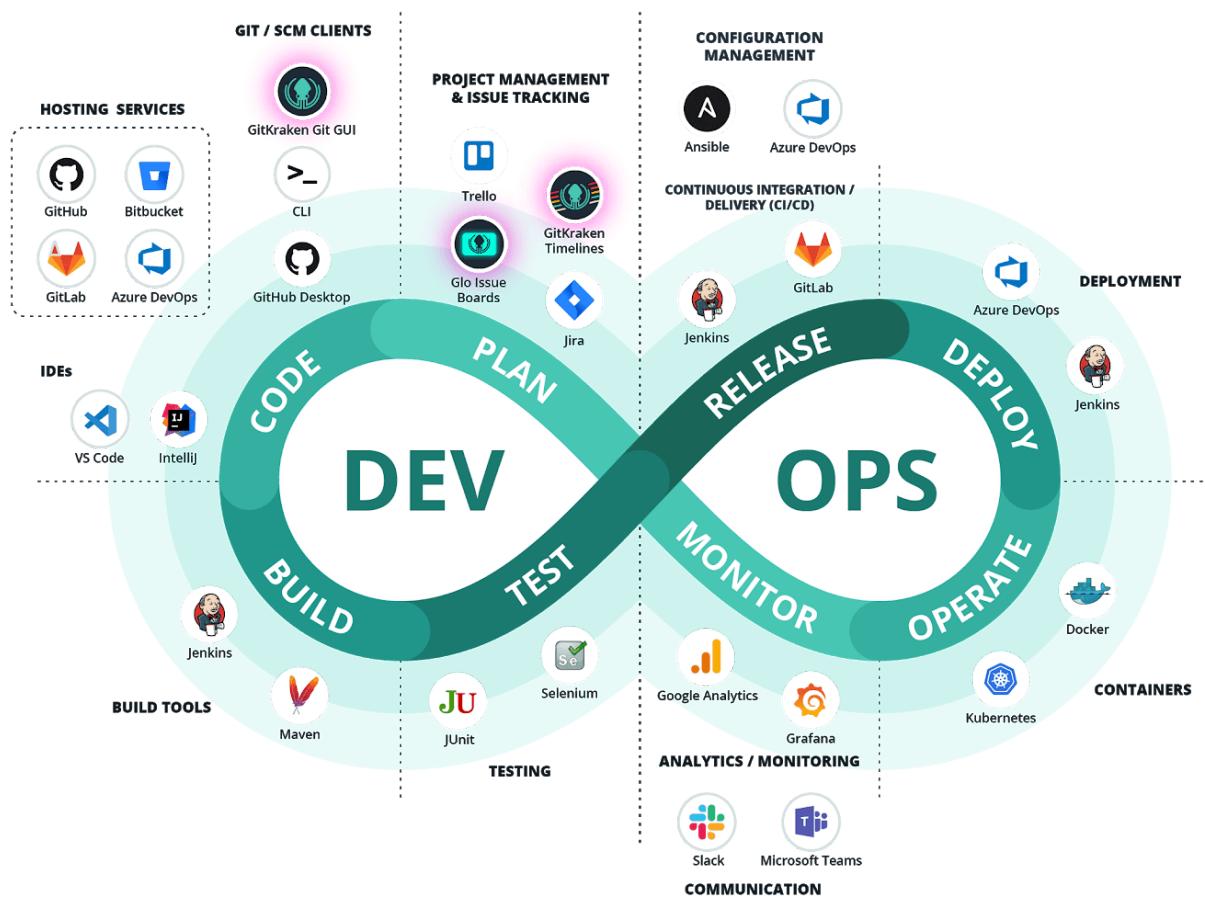


# DevOps Tools Report 2020

By GitKraken



# Table of Contents

<b>Executive Summary</b>	<b>2</b>
<b>Key Findings</b>	<b>2</b>
<b>Plan</b>	<b>3</b>
Project Management & Issue Tracking	3
Key Findings	3
Tools	3
<b>Code</b>	<b>7</b>
Version Control	7
Key Findings	7
Tools	8
Hosting Services	7
Key Findings	7
Tools	8
Git/SCM Clients	11
Key Findings	11
Tools	12
Integrated Development Environments	16
Key Findings	16
Tools	17
<b>Build</b>	<b>20</b>
Build Tools	20
<b>Test</b>	<b>21</b>
Testing	21
Key Findings	21
Tools	22
<b>Release</b>	<b>25</b>
Configuration Management	25
Continuous Integration/Delivery (CI/CD)	28
Key Findings	28
Tools	28
<b>Deploy</b>	<b>30</b>
Deployment	30
Key Findings	30
Tools	30
<b>Operate</b>	<b>31</b>
Containers	31
<b>Monitor</b>	<b>33</b>
Analytics/Monitoring	33
Communication	36
<b>Conclusion</b>	<b>38</b>

# Executive Summary

While many software methodologies have fallen in and out of favor over the decades, it's clear that DevOps is not a trend, and it's well on its way to becoming the standard way of software development and operations. Today, enterprise teams are at various stages of their DevOps transformations, working towards faster, more secure delivery of their technologies to gain a competitive advantage.

When developers work in tandem with IT leaders to remove roadblocks and create a seamless software development lifecycle, it can have an incredible effect on the productivity, performance, visibility, collaboration, and innovation of the organization.

That being said, it's not an easy or straightforward transition to DevOps. It requires significant changes, including: evolving employee mindsets, introducing the appropriate tools, and teaching new skills. No matter the stage of your DevOps transformation, your focus should be continuous improvement. Start with foundations, and then identify your unique constraints; once those constraints no longer hold you back, repeat the process.

Not having the right tools is a fairly easy constraint to eliminate and well-worth the investment. The mission of this report is to provide guidance and insights on the best DevOps tools from trailblazers who have already successfully implemented them. We asked our global community of developers which tools they rely on for DevOps success, and are proud to present this report, which is based on over 2,700 responses.

## Key Findings

When it comes to tools, useful and easy-to-use are the main criteria consumers have come to expect, but technology professionals often assume, because of their expertise, they can make any tools work. In reality, the opposite is true though: due to the degree of difficulty of building complex systems and managing business-critical infrastructure, good tools are even more important.

*"The highest-performing engineers are 1.5 times more likely to have easy-to-use tools."*

- [Accelerate: State of DevOps 2019](#) report by DORA & Google Cloud

Development teams that are empowered to make their own decisions about tools contribute to better software delivery performance. It's clear that high-performing software engineers and IT leaders choose useful and usable tools, which improve productivity and

deliver value during the DevOps transformation. They also automate and integrate tools into their toolchains, freeing up time to spend on new development, and debunking the argument that it's too time-intensive or expensive to implement.



## Plan

### Project Management & Issue Tracking

#### Key Findings

Whether your teams use Scrum, Kanban or a hybrid methodology of agile project management, you've likely been doing issue tracking for a long time. Project management and issue tracking are foundational to the planning process of DevOps. When working on a team, these tools can provide valuable transparency, accountability, and planning accuracy.

Integration and automation are of growing importance in this space, allowing for less context switching and providing the ability to perform and track tasks across platforms. High-performing technology teams often integrate tools like Slack and GitHub into their issue tracking process.

#### Tools

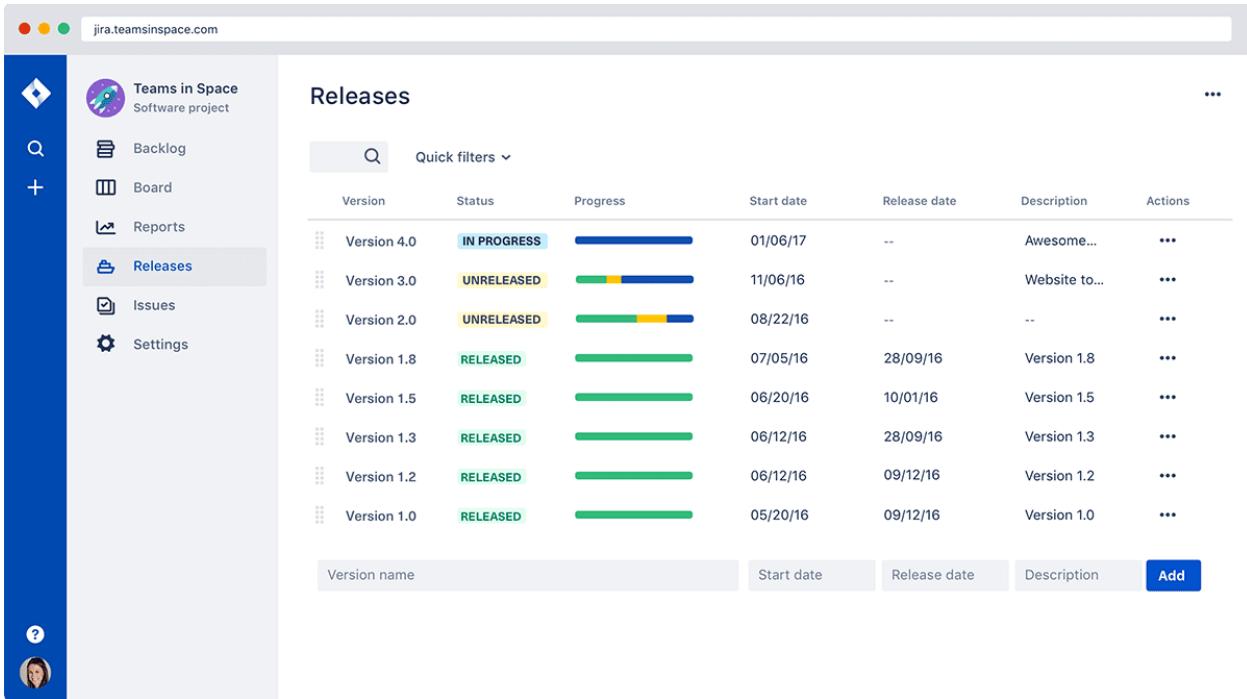


**Jira** - Jira is an Atlassian tool that was originally designed as a bug and issue tracker. Today, Jira has evolved into a powerful work management tool for all kinds of use cases, from requirements and test case management to agile software development.

Jira provides planning and roadmap tools so teams can manage stakeholders, budgets, and feature requirements. Jira integrates with a variety of CI/CD tools to facilitate transparency throughout the software development life cycle. When it's time to deploy, live production code status information is surfaced in a Jira issue. Integrated feature flagging tools allow teams to roll out new features gradually and safely.

This is a popular DevOps planning tool because it includes: release and sprint planning, CI/CD integrations, issue management, project backlogs, feature flagging, Jira Service Desk integration, and other developer tool integrations. It's highly configurable, which can be

great for complex project management but might not be the best option if you're looking for a system that's easy to implement and maintain.



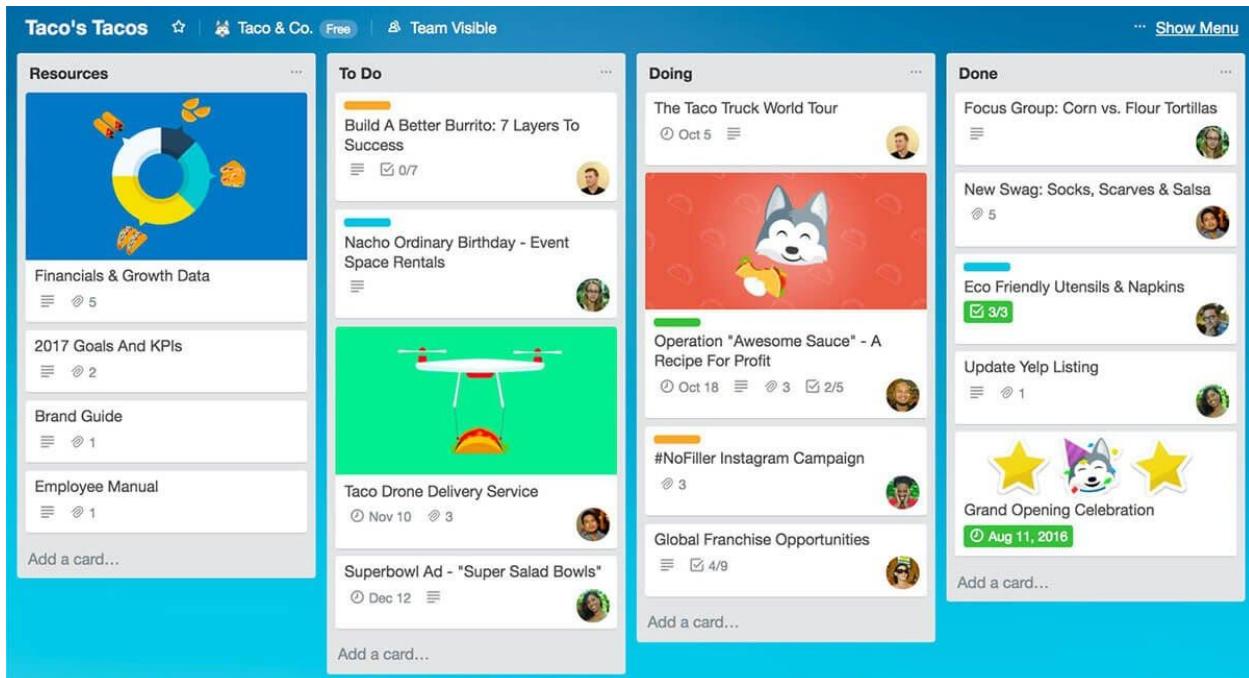
Version	Status	Progress	Start date	Release date	Description	Actions
Version 4.0	IN PROGRESS	<div style="width: 50%;"></div>	01/06/17	--	Awesome...	...
Version 3.0	UNRELEASED	<div style="width: 10%;"></div>	11/06/16	--	Website to...	...
Version 2.0	UNRELEASED	<div style="width: 15%;"></div>	08/22/16	--	--	...
Version 1.8	RELEASED	<div style="width: 100%;"></div>	07/05/16	28/09/16	Version 1.8	...
Version 1.5	RELEASED	<div style="width: 100%;"></div>	06/20/16	10/01/16	Version 1.5	...
Version 1.3	RELEASED	<div style="width: 100%;"></div>	06/12/16	28/09/16	Version 1.3	...
Version 1.2	RELEASED	<div style="width: 100%;"></div>	06/12/16	09/12/16	Version 1.2	...
Version 1.0	RELEASED	<div style="width: 100%;"></div>	05/20/16	09/12/16	Version 1.0	...



**Trello** - Trello is a project management tool that organizes projects into Kanban-style boards. In 2017, Trello was acquired by Atlassian and has since gained traction amongst software development teams as a lightweight alternative to Jira. If your organization isn't already reliant on Jira, you might consider Trello. It's easier to configure and manage, and often-times developers prefer to interface with it over Jira, which can become quite complex and sometimes unwieldy due to all the customization capabilities.

Trello offers web-based and mobile versions. Your boards tell you what's being worked on, by who, and where it is in a process. Boards are filled with cards, which are tasks for you and your team. Your team can comment and collaborate on cards, and each individual card can have photos, attachments, due dates, and more.

To get more out of Trello, you can integrate with other development tools like Slack and GitHub, through Power-Ups.



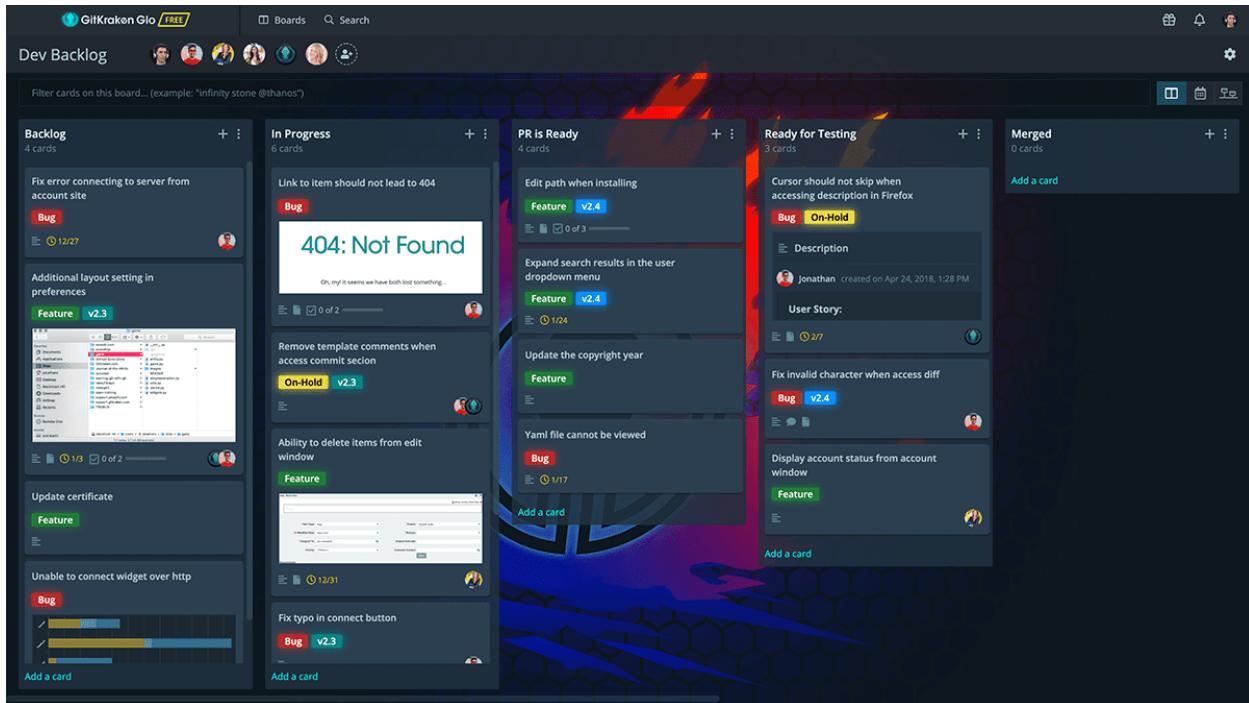
**Glo Issue Boards** - Glo Issue Boards is an Axosoft product in the GitKraken suite of tools. If you're not familiar, it's [similar to Trello but Glo is more developer-centric](#). This task and issue tracking system allows development teams to visualize tasks in a Kanban board, calendar, timeline, or dashboard.

Glo integrates directly with GitHub in order to reduce context switching as your development team completes tasks and makes progress towards milestones. Glo performs two-way sync with GitHub Issues and Milestones in real-time, so developers and managers have visibility into current project progress at all times.

High-performing developers rely on automation to increase productivity, and planning/tracking tools that empower automation can be really powerful for this reason. It's easy to set up Glo card automation using GitHub Actions to reduce the number of steps in a developer's workflow: essentially automating the progression of tasks through a workflow.

Additionally, linking cards to pull requests provides further automation. When a pull request status is updated in GitHub, the Glo card will automatically advance to another column based on your chosen mapping. Glo is fully integrated with the GitKraken Git GUI so that developers can actually link a Glo card while creating a GitHub pull request in

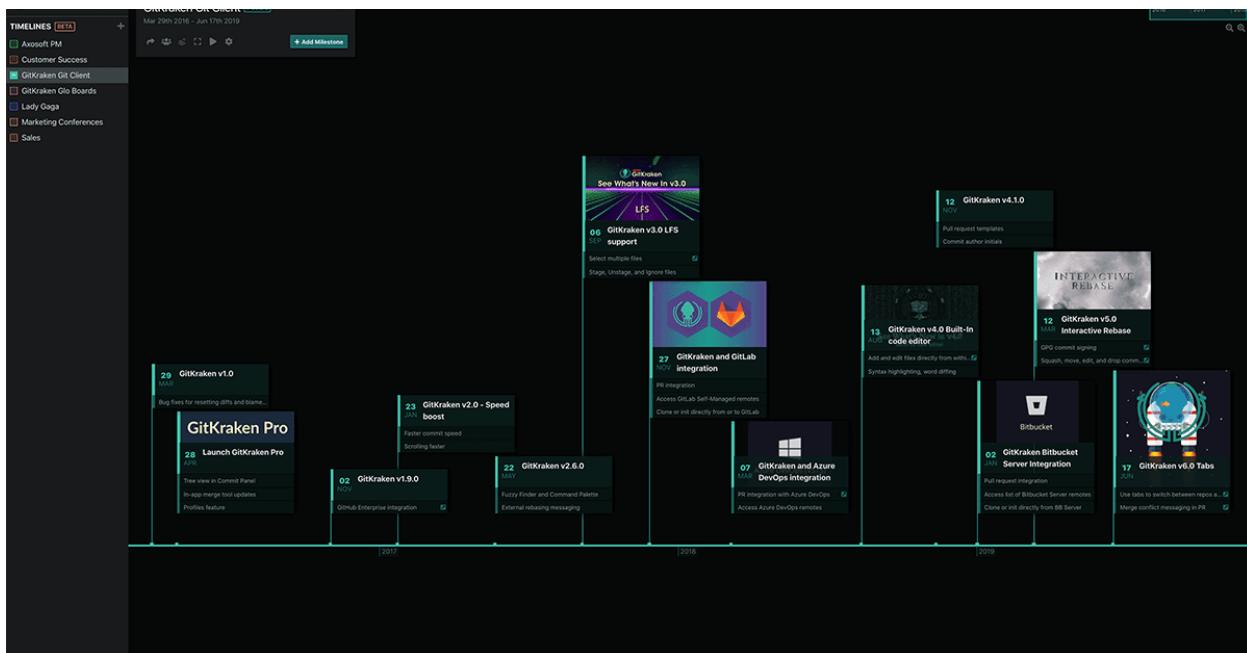
GitKraken. These automation features are very much in line with a DevOps strategy that enables less context switching and increased efficiency.



### GitKraken Timelines

- GitKraken Timelines is an Axosoft product in the GitKraken suite of tools. It's designed to help teams plan and communicate project goals and milestones in a timeline view. For high-level planning, this is a great tool to quickly convey upcoming deadlines or to review progress. Each milestone can have an image or gif and associated sub-items. It's easy to overlay multiple timelines to compare deadlines across projects or teams. Timelines can be private, collaborative or public. And they're easily shareable via links, embedding or in presentation mode.

As you set out on your DevOps transformation, consider creating timelines to clearly communicate the major milestones, what tasks need to be completed for each, and what the associated deadlines are. This planning tool will keep everyone moving toward the same goals.



## Code

### Version Control

#### Key Findings

Version control is a method of tracking and managing changes to code; allowing developers to see the complete revision history of a project and revert back to a former version or file if needed. Teams worldwide are moving away from centralized version control systems (VCS), like Subversion, and instead, migrating to Git. Because Git is a free, distributed VCS that utilizes branching and merging, over [90% of developers now use Git](#) for version control, according to Stack Overflow's Developer Survey.

### Hosting Services

#### Key Findings

In order to collaborate on projects using Git, you'll need a hosting service for your repositories; alternatively, some enterprises choose to host them on internal servers to better suit their security or deployment requirements. When making the decision on which

hosting service to use, your organization will want to consider price, storage capacity, integrations with your current tools, etc. It's also not uncommon for enterprise teams to host their repos on multiple services.

## Tools



[GitHub](#) - At its core, GitHub is a platform where hundreds of millions of private, public, and open source repositories are hosted and reviewed. GitHub is also the name of the company that builds the product, which was acquired by Microsoft in 2018. Not only is GitHub the #1 hosting service in our DevOps report, it came in at #7 in our report of the [Top 20 Developer Tools for 2020](#).

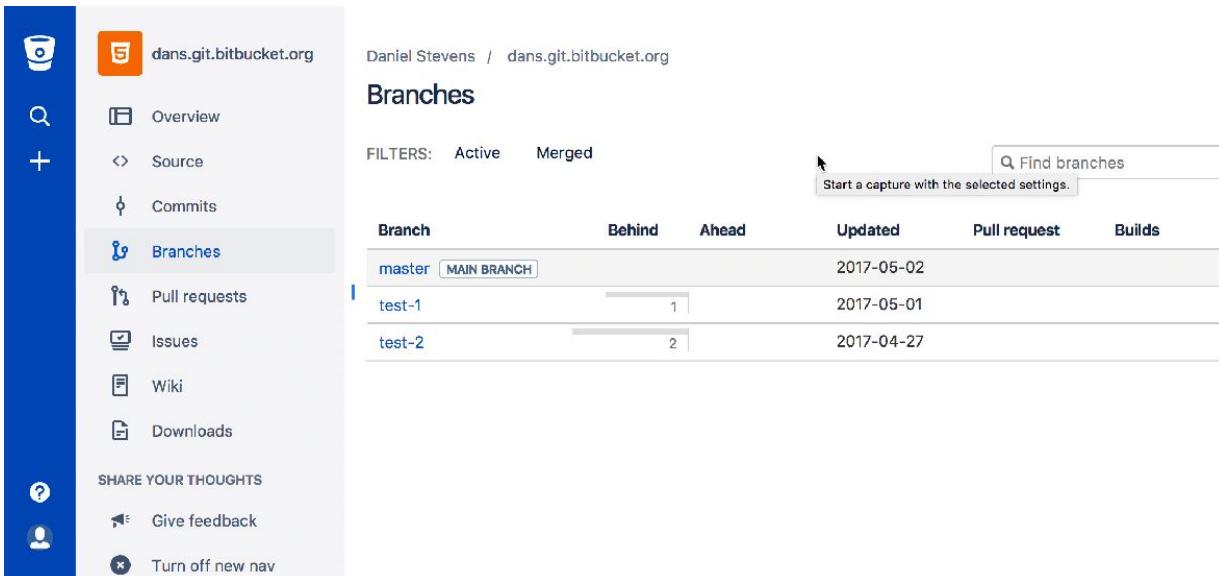
Owner	Repository name
<input checked="" type="checkbox" value="public"/>  hubot	<input style="border: 1px solid #ccc; padding: 2px; width: 150px;" type="text" value="hello-world"/> 
Great repository names are short and memorable. Need inspiration? How about <a href="#">petulant-shame</a> .	
<b>Description (optional)</b>	
Just another repository	
<b><input checked="" type="radio"/>  Public</b> Anyone can see this repository. You choose who can commit.	
<b><input type="radio"/>  Private</b> You choose who can see and commit to this repository.	
<b><input checked="" type="checkbox"/> Initialize this repository with a README</b> <small>This will allow you to git clone the repository immediately. Skip this step if you have already run git init locally.</small>	
<input style="border: 1px solid #ccc; padding: 2px; width: 200px;" type="button" value="Add .gitignore: None"/>	<input style="border: 1px solid #ccc; padding: 2px; width: 200px;" type="button" value="Add a license: None"/>
<input style="background-color: #28a745; color: white; border: 1px solid #28a745; padding: 5px; font-weight: bold; border-radius: 5px; width: 150px;" type="button" value="Create repository"/>	

GitHub is increasingly expanding its offerings to align with more and more processes in the DevOps workflow. To interface with GitHub repositories, many developers use the GitKraken Git GUI, which seamlessly integrates with GitHub.com and GitHub Enterprise. GitHub offers basic project management with Projects and Issues. Tools like Glo Issue Boards offer integrations that sync with GitHub to provide more robust planning and tracking capabilities. The other core components of GitHub are code review, secure development and, most recently, CI/CD.



**Bitbucket** - Bitbucket is an Atlassian product, and is a code management tool first and foremost. To interface with Git repos hosted on Bitbucket, the GitKraken Git GUI and several other Git clients integrate with Bitbucket.org or Bitbucket Server to provide a streamlined workflow.

Following the DevOps methodology, Bitbucket has extended its offerings to not just hosting, but project planning, collaboration, testing, and deployment services. As you'd expect, Bitbucket offers tight integration with Jira, and Trello has a powerup to integrate with Bitbucket Cloud.



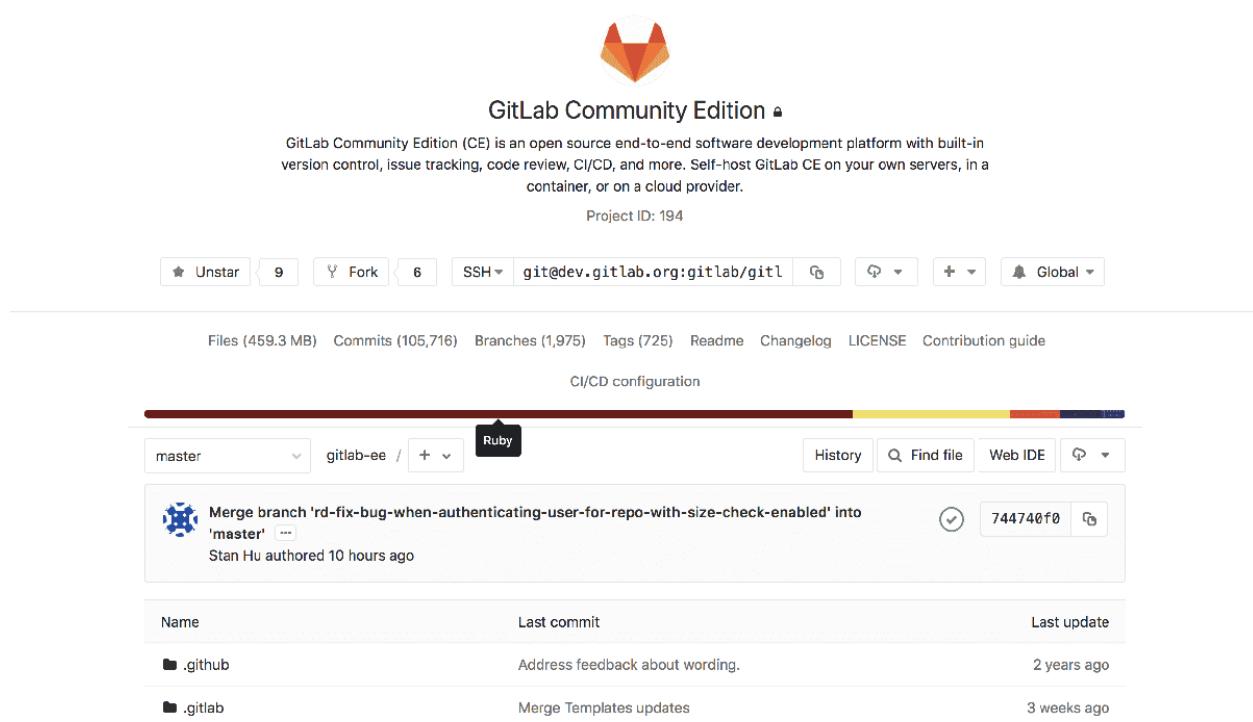
The screenshot shows the Bitbucket interface for the repository `dans.git.bitbucket.org`. The left sidebar includes links for Overview, Source, Commits, Branches (which is selected), Pull requests, Issues, Wiki, and Downloads. The main content area displays the 'Branches' section with a table. The table has columns: Branch, Behind, Ahead, Updated, Pull request, and Builds. It lists three branches: 'master' (MAIN BRANCH), 'test-1', and 'test-2'. The 'master' branch was updated on 2017-05-02. The 'test-1' branch is 1 commit behind and was updated on 2017-05-01. The 'test-2' branch is 2 commits behind and was updated on 2017-04-27.

Branch	Behind	Ahead	Updated	Pull request	Builds
master <small>MAIN BRANCH</small>			2017-05-02		
test-1	1		2017-05-01		
test-2	2		2017-04-27		



**GitLab** - GitLab is the #3 most-used hosting service in our DevOps report and #14 in the [Top 20 Developer Tools for 2020](#). It was one of the first hosting services to fully embrace DevOps and has since been on a mission to create a complete DevOps platform. GitLab provides everything to manage, plan, create, verify, package, release, configure, monitor, and secure your applications. To streamline the development workflow one step further, utilize the GitKraken Git GUI to integrate with Git repos hosted on GitLab.com and GitLab Self-Managed.

 GitLab >  GitLab Community Edition > Details

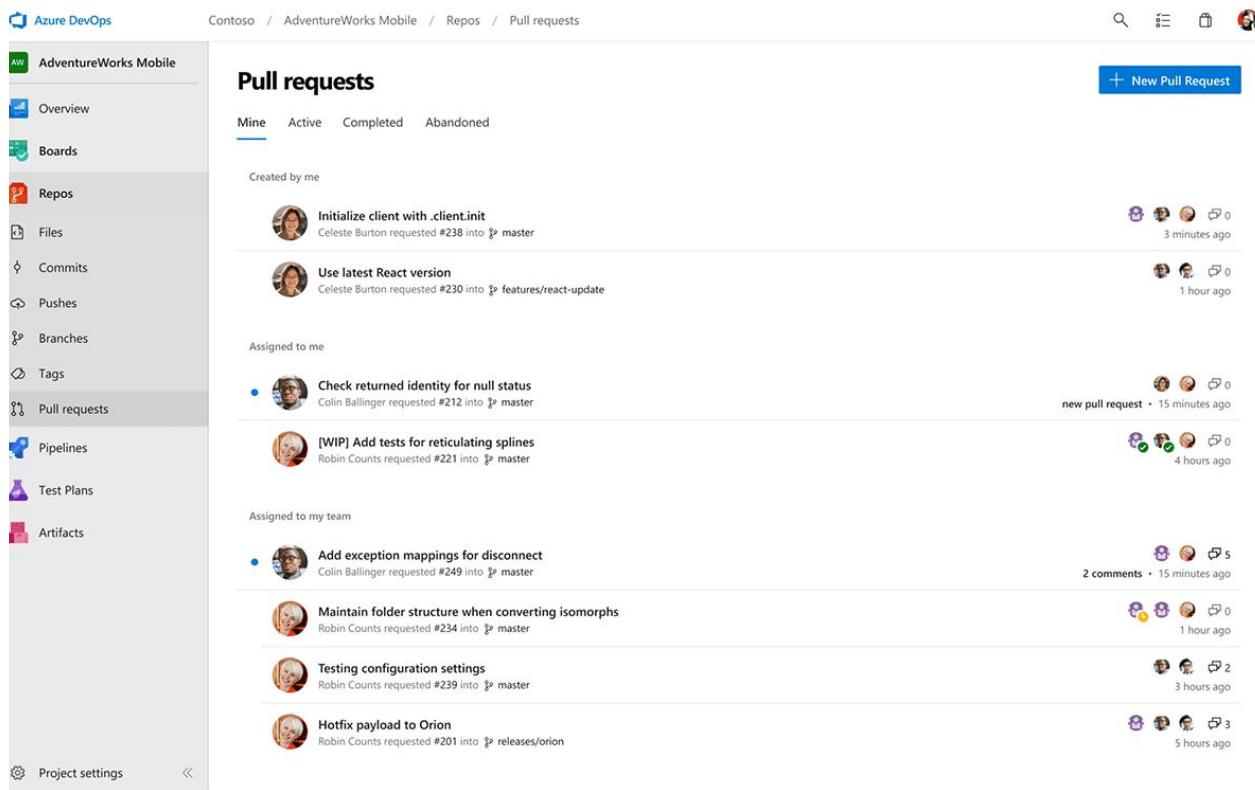


The screenshot shows the GitLab Community Edition repository details. It includes the repository icon, name, description, project ID, and various navigation links like Unstar, Fork, SSH, and Global notifications. Below this is the commit history, showing a merge commit from 'rd-fix-bug-when-authenticating-user-for-repo-with-size-check-enabled' into 'master' by Stan Hu. The commit message is 'Address feedback about wording.' and it was authored 10 hours ago. At the bottom, there's a table of files and their last commits.

Name	Last commit	Last update
.github	Address feedback about wording.	2 years ago
.gitlab	Merge Templates updates	3 weeks ago



**Azure DevOps** - Azure DevOps is a Microsoft product that provides hosting for Git repositories, reporting, requirements management, project management, automated builds, lab management, testing and release management capabilities. This offering was released in 2018, replacing tools like Visual Studio Team Services (VSTS) and combining many others to cover the entire application lifecycle and enable DevOps capabilities. To extend your DevOps toolchain and streamline development one step further, integrate the GitKraken Git GUI with your Azure DevOps hosted Git repos.



The screenshot shows the Azure DevOps interface for the AdventureWorks Mobile project. The left sidebar includes options like Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pull requests (which is selected), Pipelines, Test Plans, and Artifacts. The main content area is titled "Pull requests" and lists several items:

- Mine**: Shows a pull request from Celeste Burton to initialize a client with ".client.init". It was created 3 minutes ago.
- Active**: Shows a pull request from Celeste Burton to use the latest React version. It was created 1 hour ago.
- Completed**: Shows a pull request from Colin Ballinger to check returned identity for null status. It was created 15 minutes ago.
- Abandoned**: Shows a pull request from Robin Counts to add tests for reticulating splines. It was created 4 hours ago.
- Created by me**: Shows a pull request from Colin Ballinger to add exception mappings for disconnect. It has 2 comments and was created 15 minutes ago.
- Assigned to me**: Shows a pull request from Robin Counts to maintain folder structure when converting isomorphs. It was created 1 hour ago.
- Assigned to my team**: Shows two pull requests from Robin Counts: one to testing configuration settings (created 3 hours ago) and another to hotfix payload to Orion (created 5 hours ago).

## Git/SCM Clients

### Key Findings

Source control management (SCM) systems are tools for helping teams and developers track their project history. A Git GUI (graphical user interface) translates what's going on under the hood of Git into an interface that your eyes and brain can easily understand.

GUIs provide a vital layer of visual clarity, demystifying the black box experience of the CLI. GUIs also reduce the steep learning curve of Git by replacing the need to memorize a list of commands with simple drag-and-drop actions. It's quicker and more efficient to perform all of these tasks in the GitKraken Git GUI rather than the CLI: authentication, cloning a repo, viewing your commit graph, viewing remote URLs, viewing a file diff, pushing changes from a local repo to a remote, accessing file history and blame, performing pull requests and merge resolutions, and performing an interactive rebase.

Without a Git GUI, enterprises struggle to standardize and scale Git across all their development teams. The continued growth of the GitKraken Git GUI speaks to the significance of why the GUI approach is integral to achieving a successful DevOps workflow with Git.

## Tools



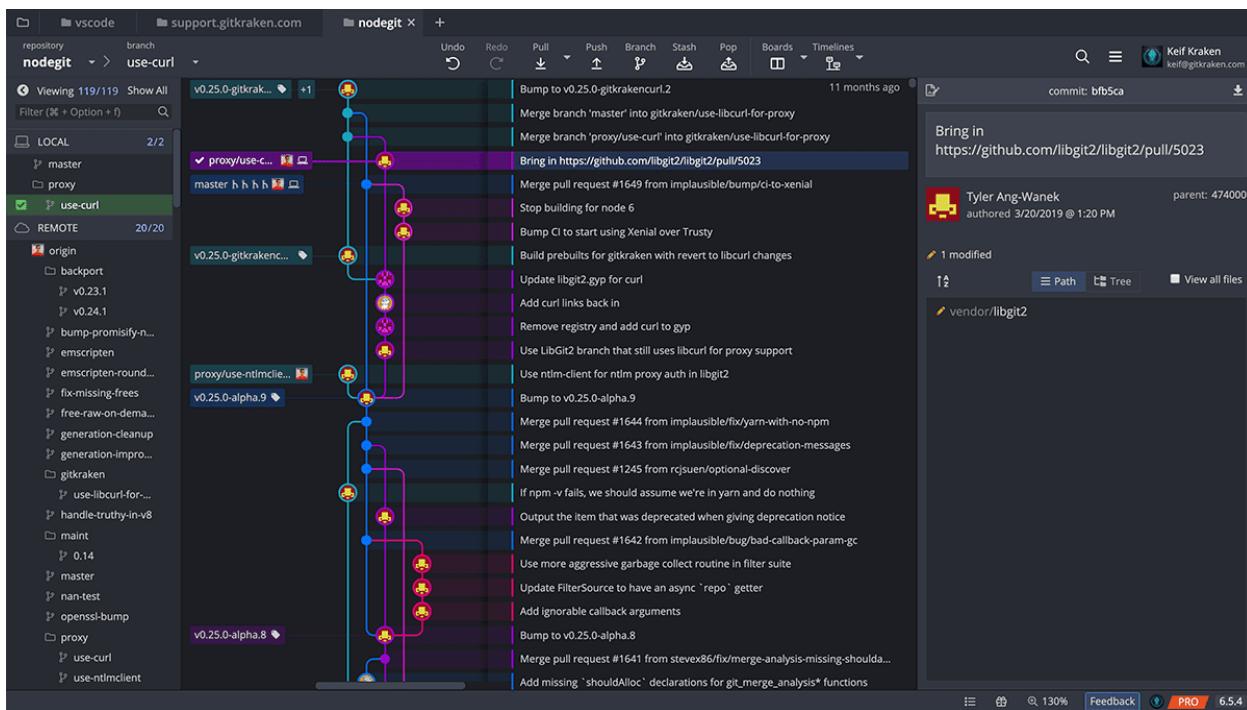
### GitKraken Git GUI

GitKraken Git GUI - Voted [#1 developer tool](#) four years in a row, the GitKraken Git GUI is the flagship product in the GitKraken suite of tools built by Axosoft. It allows developers to visualize the history of their Git repositories in a colorful graph, and it simplifies complicated Git commands into drag-and-drop actions. With a built-in merge conflict editor, interactive rebase mode, built-in code editor, integrations, and more, the GitKraken Git GUI streamlines the Git workflow for experienced developers and reduces the steep learning curve for those who are new to Git.

GitKraken stands out from other clients because it's one of the few GUIs available on **Linux**, Mac, and Windows—unlike GitHub Desktop and Sourcetree, which do not support Linux. It plays a key role in the DevOps workflow by tightly connecting the various Code tools: Git client, hosting service and IDE (it has the Monaco code editor from VS Code built-in).

The GitKraken Git GUI integrates with all the top Git hosting services listed in the previous section: GitHub, GitLab, Bitbucket, and Azure DevOps, and their self-hosted offerings—excluding TFS—to do enable all of the following from inside GitKraken: create repositories on your hosting account including .gitignore and license; automatically generate an SSH key pair and add it; fork repositories; save authentication into profiles; clone from your repo list; add remotes for repos; create pull requests with added assignees, reviewers, and labels; view build statuses of pull requests.

The GitKraken Git GUI also supports a seamless DevOps workflow by connecting the Plan and Code steps. The GitKraken Git GUI has planning tools like Glo Issue Boards and GitKraken Timelines built-in. For easy tracking, repositories in the Git GUI can be associated with Glo Boards. When creating a new GitHub pull request, simply link a Glo card; this will automatically update the pull request description in GitHub to include a link to the card. Plus, tighter, more robust integrations for the GitKraken Git GUI with Glo Issue Boards, Jira, GitHub Issues and GitLab Issues are coming soon!



[Git CLI](#) - Before Git GUIs, programmers were forced to use the command line interface (CLI). The CLI continues to be widely used because it's free and many people still learn Git by memorizing commands. Additionally, it offers the added bonus of being able to automate certain tasks with scripts. Some developers also just prefer to not rely on a GUI application and would rather see the output of the commands they're typing.

```
● ● ● 2. git diff ece08a5821e426eabbb76b83066fd134f955cb07 (less)
diff --git a/test/sequential/test-inspector-contexts.js b/test/sequential/test-inspector-contexts.js
index 2307caa572..f27f95da99 100644
--- a/test/sequential/test-inspector-contexts.js
+++ b/test/sequential/test-inspector-contexts.js
@@ -5,8 +5,8 @@
 const common = require('../common');
 common.skipIfInspectorDisabled();

-const { ifError, strictEqual } = require('assert');
-const { createContext, runInNewContext } = require('vm');
+const assert = require('assert');
+const vm = require('vm');
 const { Session } = require('inspector');

 const session = new Session();
@@ -31,18 +31,18 @@
     async function testContextCreatedAndDestroyed() {
         // "Administrator: Windows PowerShell[42]" because of a GetConsoleTitle()
         // quirk. Not much we can do about either, just verify that it contains
         // the PID.
-        strictEqual(name.includes(`[${process.pid}]`), true);
+        assert.strictEqual(name.includes(`[${process.pid}]`), true);
     } else {
         let expects = `${process.argv0}[${process.pid}]`;
         if (!common.isMainThread) {
             expects = `Worker[${require('worker_threads').threadId}]`;
         }
-        strictEqual(expects, name);
+        assert.strictEqual(expects, name);
     }
-        strictEqual(origin, '',
-                    JSON.stringify(contextCreated));
-        strictEqual(auxData.isDefault, true,
-                    JSON.stringify(contextCreated));
+        assert.strictEqual(origin, '',
+                           JSON.stringify(contextCreated));
+        assert.strictEqual(auxData.isDefault, true,
+                           JSON.stringify(contextCreated));
     }

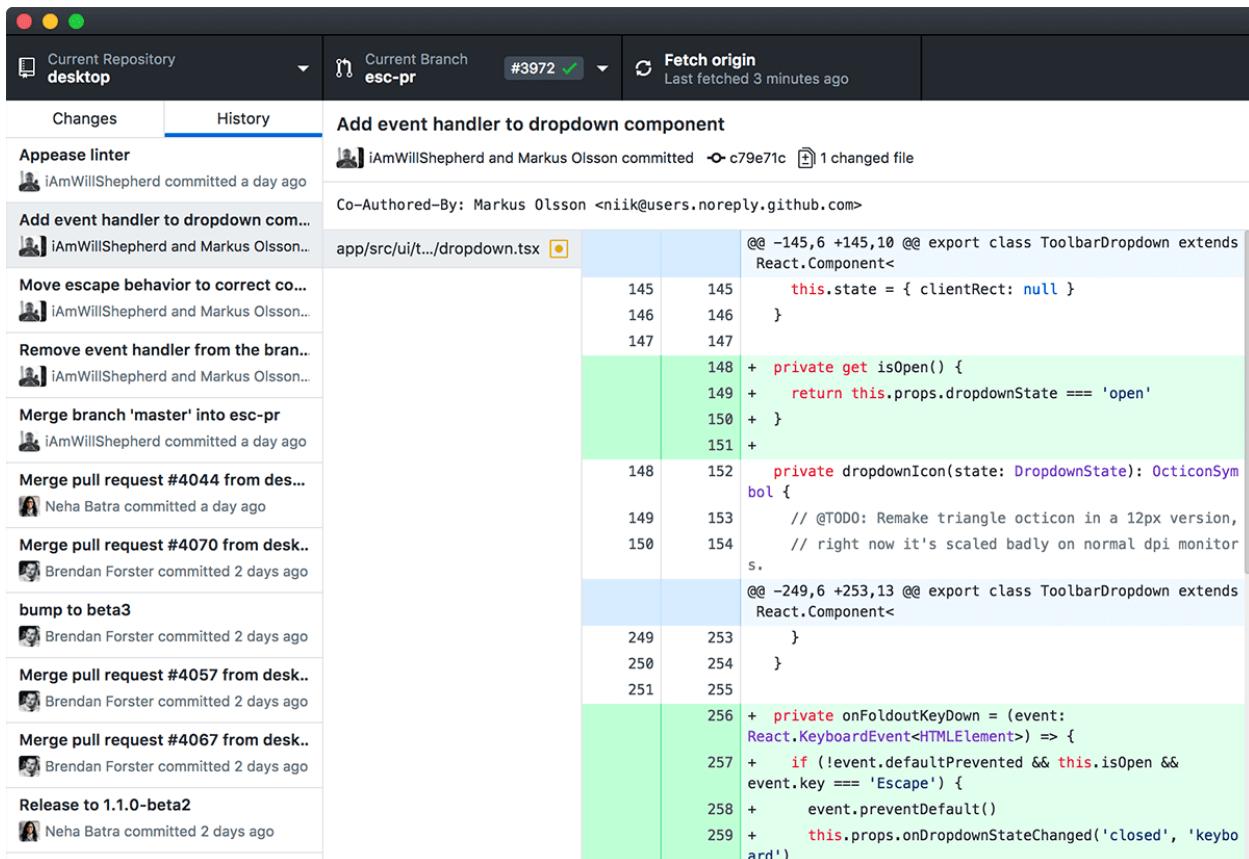
     {
@@ -53,23 +53,23 @@
         async function testContextCreatedAndDestroyed() {
             session.once('Runtime.executionContextDestroyed',
                         (notification) => contextDestroyed = notification);

-            runInNewContext('1 + 1');
+:
```



[GitHub Desktop](#) - As the name suggests, GitHub Desktop is a GitHub tool, under the Microsoft umbrella. This tool has been widely adopted because of GitHub.com's extensive user base. It's a free, easy-to-use tool for GitHub users, but it's not nearly as feature robust as the GitKraken Git GUI or Sourcetree. And for those not using GitHub as their hosting service—or using multiple services—there are no integrations for GitLab, Bitbucket or Azure DevOps, which is a pretty significant workflow roadblock.

GitHub Desktop is ideal for Windows and Mac developers who already use GitHub.com and need basic Git client functionality, such as attributing commits with collaborators; checking out branches with pull requests and viewing CI statuses; syntax highlighted diffs; expanded image diff support. This Git client is not for Linux users or teams who need a client for interactive rebase, commit signing, GitFlow, submodules, blame, opening multiple repos, auto stash, commit templates, file/diff views, file editing, etc. GitHub Desktop also doesn't feature a commit graph, so the ability to visualize project history is limited.



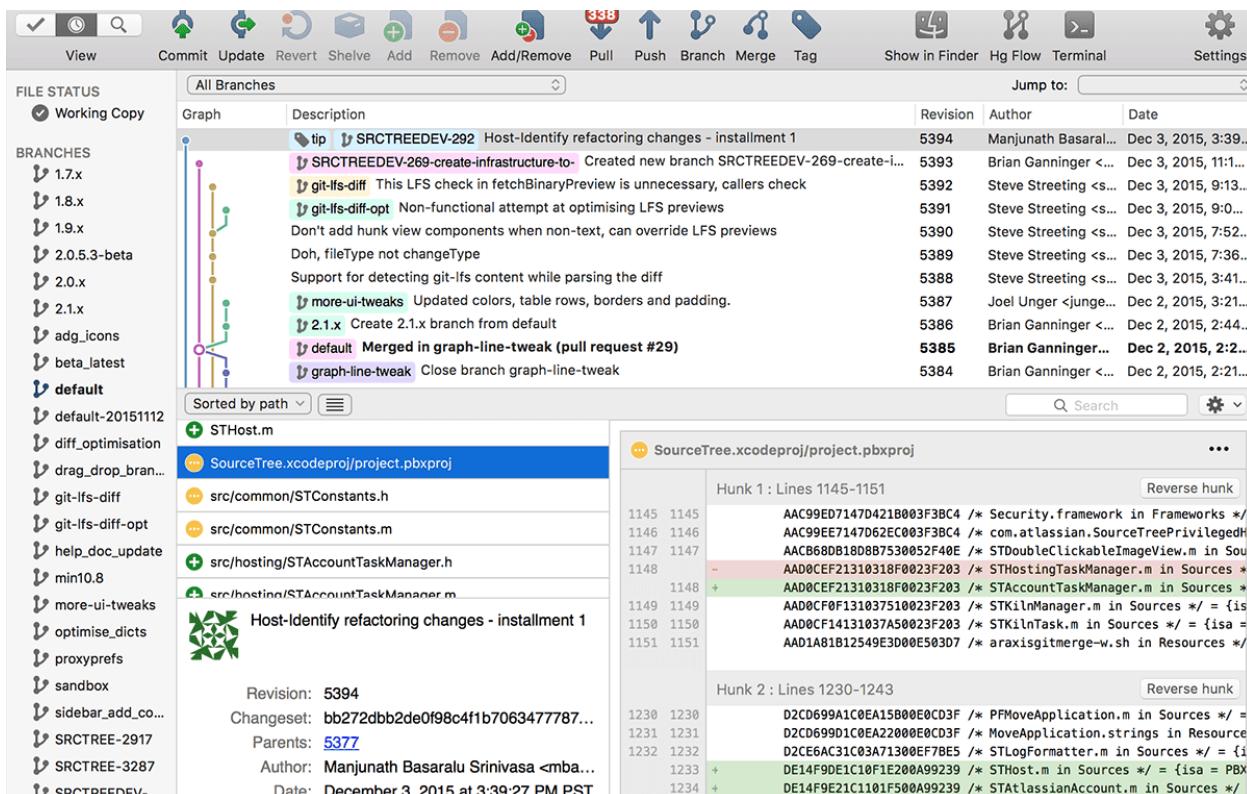
The screenshot shows the GitHub Desktop application interface. At the top, there are three tabs: 'Current Repository' (set to 'desktop'), 'Current Branch' (set to 'esc-pr #3972'), and 'Fetch origin' (last fetched 3 minutes ago). Below the tabs, there are two main sections: 'Changes' and 'History'. The 'Changes' section is active and displays a list of commits from various authors (iAmWillShepherd, Markus Olsson, Neha Batra, Brendan Forster) across different pull requests. The 'History' section is shown below the changes. On the right side of the screen, a detailed view of a specific commit's diff is displayed. The commit subject is 'Add event handler to dropdown component'. It shows code changes in a file named 'app/src/ui/t.../dropdown.tsx'. The diff highlights added lines in green and removed lines in red. The code snippet includes imports, class definitions, and methods like 'isOpen()' and 'onFoldoutKeyDown()'. Line numbers 145 through 259 are visible on both the left and right sides of the diff view.



[\*\*Sourcetree\*\*](#) - Sourcetree is an Atlassian product, first released to the public in 2010. This Git client is free for Windows and Mac, but does not support Linux. As one of the first Git GUIs on the market, Sourcetree was able to gain significant traction amongst developers, especially those already using Atlassian products like Jira and Bitbucket.

Sourcetree is much more feature-rich than GitHub Desktop, with closer feature parity to the GitKraken Git GUI. However, Sourcetree is lacking a Fuzzy Finder, syntax highlighting, auto stash, file/diff views, file editing and pull request templates.

When it comes to DevOps, Sourcetree offers integrations with GitHub, GitLab and Azure DevOps—with a primary focus on Bitbucket.



## Integrated Development Environments

### Key Findings

IDEs have become increasingly popular with software developers because of the convenience they provide. An IDE is a software suite that consolidates many of the tools developers use to write and test software, into one user interface. IDEs provide the advantage of less context switching, which is why many tools are moving in this direction—like the GitKraken Git GUI with its built-in code editor, merge conflict tool and integrated issue tracking.

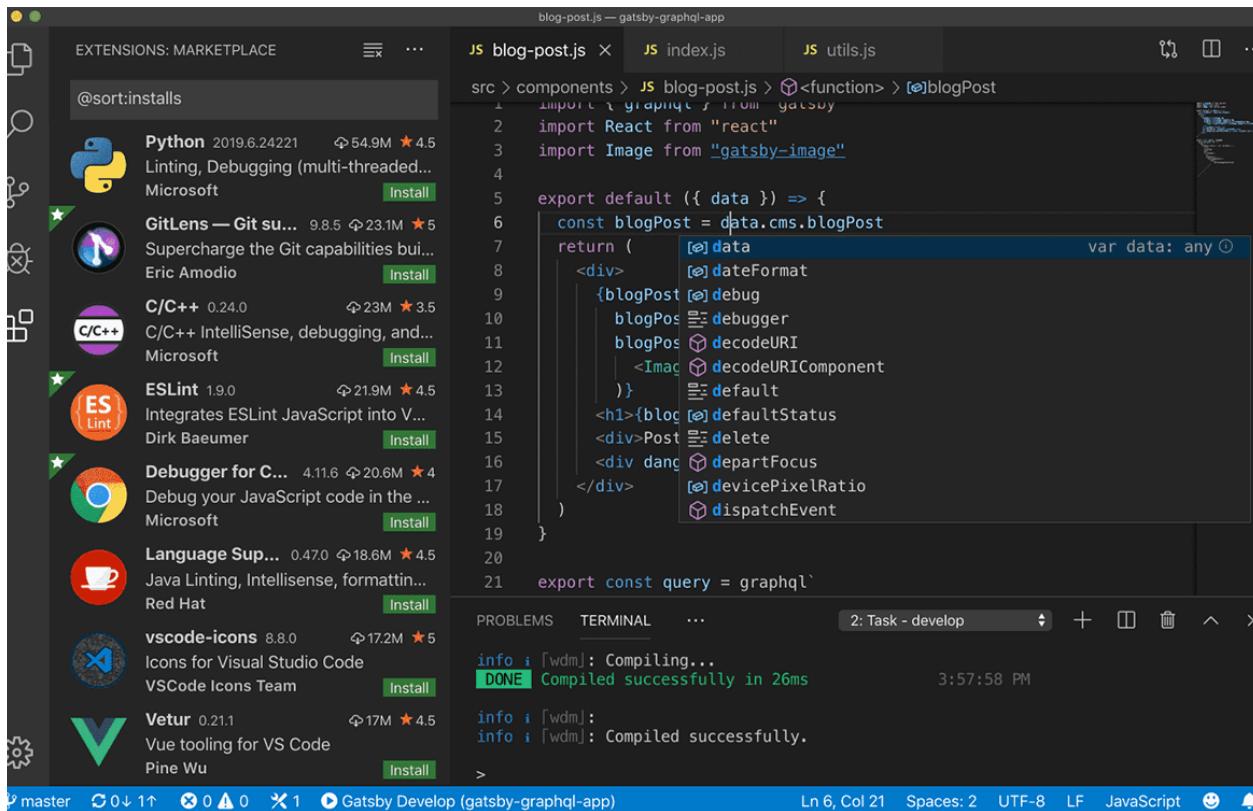
### Tools



**VS Code** - Not only is VS Code the #1 IDE in our DevOps report, but it was also named #2 developer tool in the [Top 20 Developer Tools for 2020](#), 2019 and 2018. VS Code

is a very popular code editor for writing, building and debugging web and cloud applications on Windows, Mac and Linux.

As a Microsoft tool, it has the added advantage of tight integration with Azure, AWS, .NET and a vast ecosystem of extensions which allow you to connect, build and debug many tools and technologies. Streamline your DevOps workflow by using VS Code with Azure to easily deploy and host sites built on React, Angular, Vue, Node, Python, etc. Additionally, expedite task and issue tracking with the Glo Issue Boards plugin for VS Code.



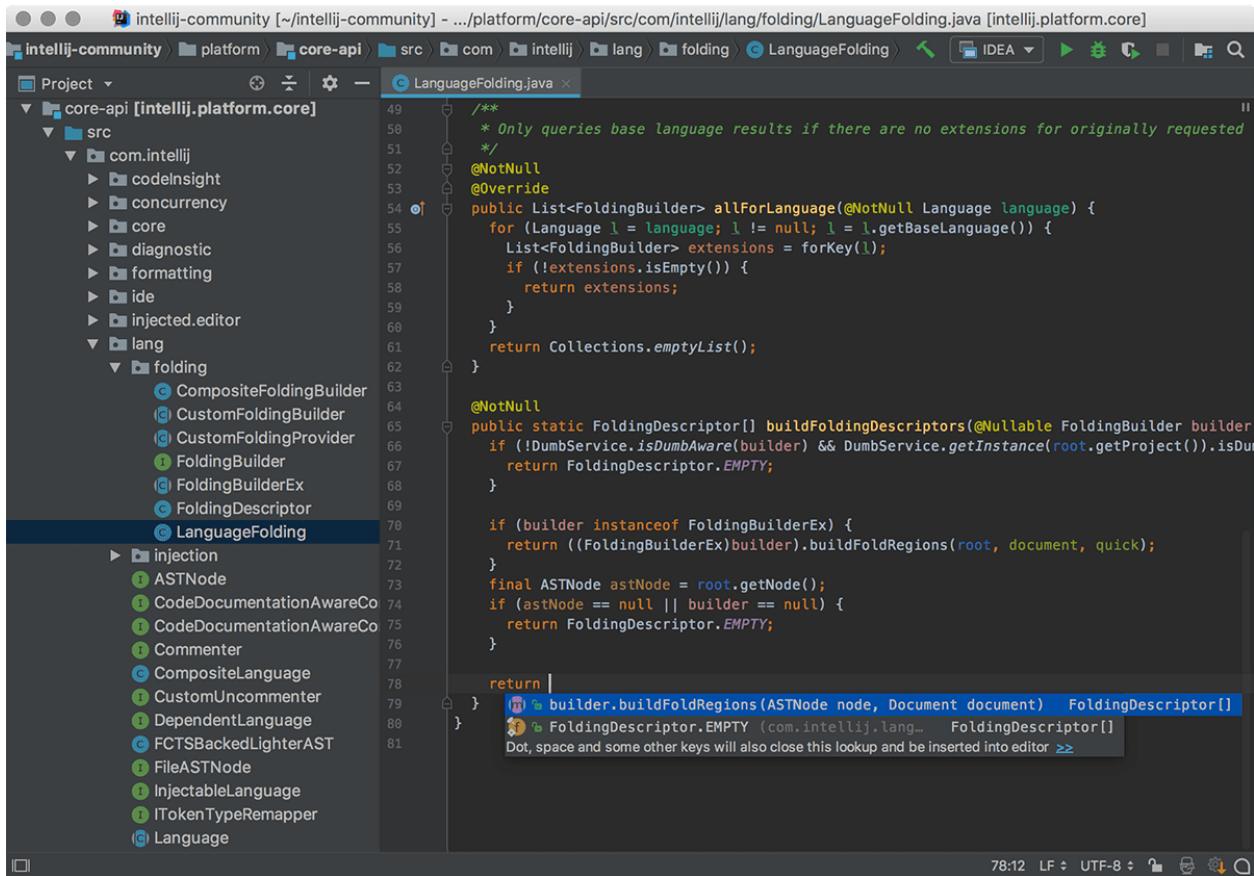
The screenshot shows the Visual Studio Code (VS Code) interface. On the left, the Extensions Marketplace sidebar is open, displaying a list of popular extensions such as Python, GitLens, C/C++, ESLint, Debugger for Chrome, Language Support for Java, vscode-icons, and Vetur. The main workspace shows two tabs: 'blog-post.js' and 'index.js'. The 'blog-post.js' tab contains code related to a Gatsby GraphQL application. The terminal at the bottom shows the command 'Gatsby Develop (gatsby-graphql-app)' and output indicating a successful compilation: 'info i [wdm]: Compiling...' and 'DONE Compiled successfully in 26ms' at 3:57:58 PM. Status bar details include 'Ln 6, Col 21', 'Spaces: 2', 'UTF-8', 'LF', 'JavaScript', and file status indicators.



**IntelliJ IDEA** - IntelliJ IDEA is a Java IDE by JetBrains, which creates a whole suite of developer tools. While not as popular as VS Code, IntelliJ is the #2 most-used IDE in our DevOps report and #9 in the [Top 20 Developer Tools for 2020](#). This developer tool has continued to gain traction year over year, climbing the ranks in our Top 20 Developer Tools report from #10 in 2019 and #11 in 2018.

IntelliJ IDEA offers a fast and intuitive experience for coding software. While IntelliJ is an IDE for Java, it also understands and provides intelligent coding assistance for a large variety of other languages such as SQL, JPQL, HTML, JavaScript, etc.

To enhance your DevOps workflow, IntelliJ IDEA supports build tools like Maven, Gradle, Ant, Gant, etc. to help automate compilation, packaging, running tests, deployment and other activities. To easily perform unit testing with ease, IntelliJ includes test runners and coverage tools for major test frameworks, including JUnit, TestNG, Spock, etc. Additionally, with a separate plugin, IntelliJ IDEA provides a dedicated tool window that lets you connect to locally running Docker machines to manage images, containers and Docker Compose services.



```

    /**
     * Only queries base language results if there are no extensions for originally requested
     */
    @NotNull
    @Override
    public List<FoldingBuilder> allForLanguage(@NotNull Language language) {
        for (Language l = language; l != null; l = l.getBaseLanguage()) {
            List<FoldingBuilder> extensions = forKey(l);
            if (!extensions.isEmpty()) {
                return extensions;
            }
        }
        return Collections.emptyList();
    }

    @NotNull
    public static FoldingDescriptor[] buildFoldingDescriptors(@Nullable FoldingBuilder builder
        if (!DumbService.isDumbAware(builder) && DumbService.getInstance(root.getProject()).isDumb)
            return FoldingDescriptor.EMPTY;
        }

        if (builder instanceof FoldingBuilderEx) {
            return ((FoldingBuilderEx)builder).buildFoldRegions(root, document, quick);
        }
        final ASTNode astNode = root.getNode();
        if (astNode == null || builder == null) {
            return FoldingDescriptor.EMPTY;
        }

        return [
            builder.buildFoldRegions(astNode, Document document) FoldingDescriptor[]
        ]
    }
}

```

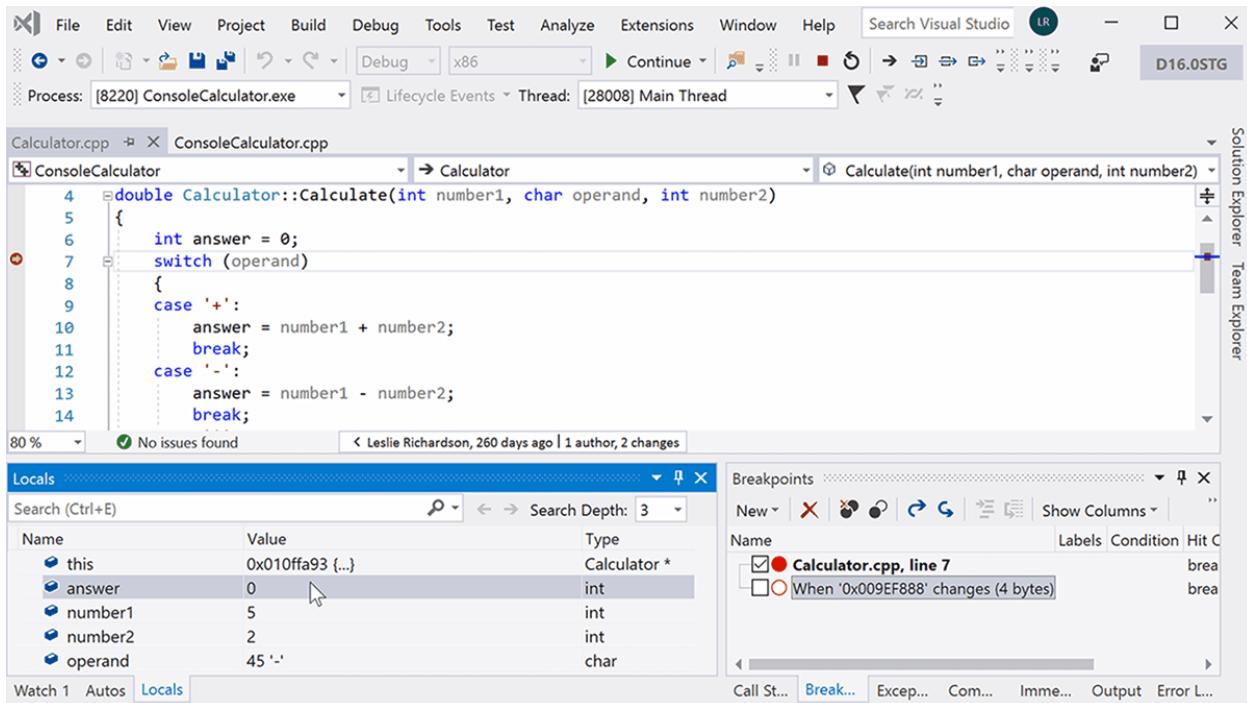


## Visual Studio

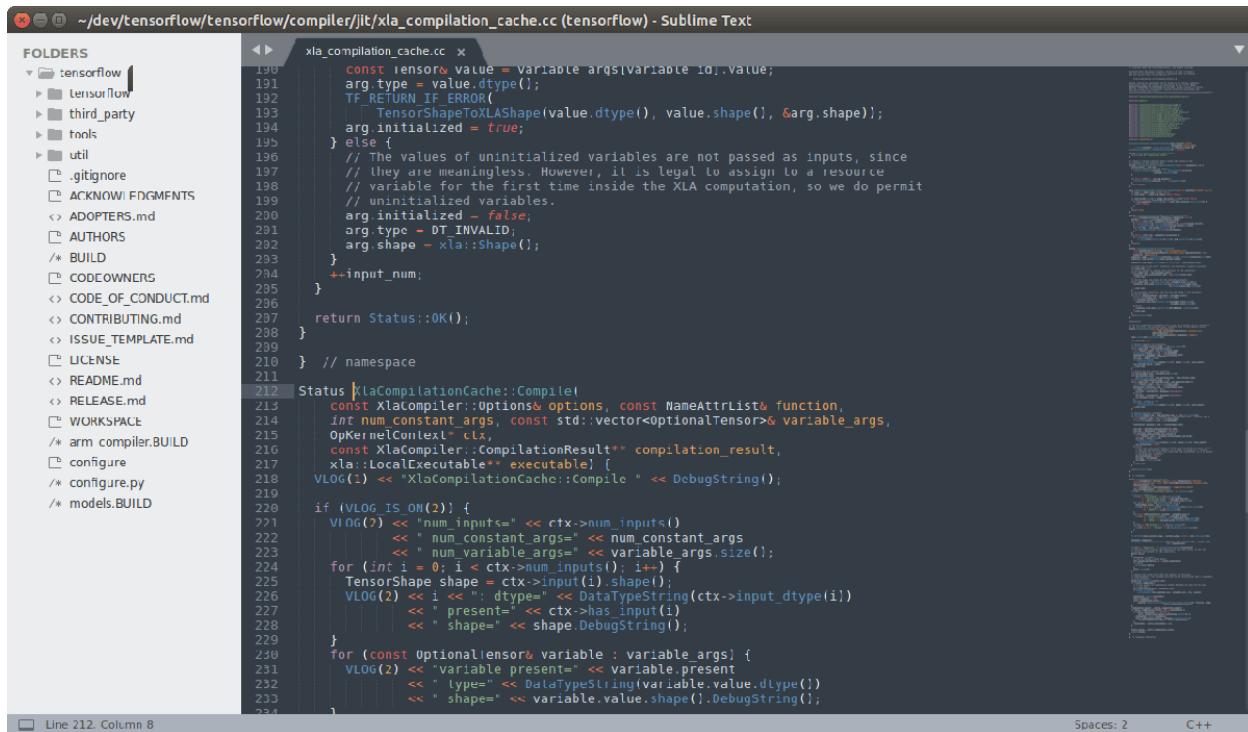
- Visual Studio, not to be confused with Visual Studio Code (VS Code), is another IDE built by Microsoft. It too is quite popular, coming in at #3 on our DevOps report and #4 on our [Top 20 Developer Tools for 2020](#). This integrated development environment includes tools and services for all platforms and languages.

Visual Studio offers features to help with various parts of the DevOps workflow: develop, analyze, debug, test, collaborate and deploy. Also, as a Microsoft tool, Visual Studio makes Azure development easier with project templates for Azure and direct deployment to

Azure. Plus, Visual Studio has a marketplace of extensions and integrations with other popular development tools.



[\*\*Sublime Text\*\*](#) - Sublime Text is a cross-platform text editor for code, markup, and prose. This mature IDE is stable and fast. It comes with autocompletion, syntax highlighting, and code folding. Sublime Text is the #4 most-used IDE in our DevOps report and #8 in the [Top 20 Developer Tools for 2020](#).



A screenshot of the Sublime Text code editor. The file being edited is `xla_compilation_cache.cc`, located at `~/dev/tensorflow/tensorflow/compiler/jit/xla_compilation_cache.cc`. The code is C++ and contains logic for handling variable arguments in XLA computations. The interface shows the file structure on the left and the code editor on the right. Status bar details: Line 212, Column 8, Spaces: 2, C++.

```

FOLDERS
tensorflow
  - Tensorflow
  - third_party
  - tools
  - util
  - .gitignore
  - ACKNOWLEDGMENTS
  - ADOPTERS.md
  - AUTHORS
  - BUILD
  - CODEOWNERS
  - CODE_OF_CONDUCT.md
  - CONTRIBUTING.md
  - ISSUE_TEMPLATE.md
  - LICENSE
  - README.md
  - RELEASE.md
  - WORKSPACE
  - arm compiler.BUILD
  - configure
  - configure.py
  - models.BUILD

xla_compilation_cache.cc
190   const tensorflow::Value* value = VariableArgs::Variable(0).Value();
191   arg_type = value.dtype();
192   TF_RETURN_IF_ERROR(
193     TensorShapeToXLAShape(value.dtype(), value.shape(), &arg.shape));
194   arg.initialized = true;
195 } else {
196   // The values of uninitialized variables are not passed as inputs, since
197   // they are meaningless. However, it is legal to assign to a resource
198   // variable for the first time inside the XLA computation, so we do permit
199   // uninitialized variables.
200   arg.initialized = false;
201   arg.type = DT_INVALID;
202   arg.shape = xla::Shape();
203 }
204 ++input_num;
205
206 return Status::OK();
}
} // namespace
Status XlaCompilationCache::Compile(
    const XlaCompiler::Options& options,
    const NameAttrList& function,
    int num_constant_args,
    const std::vector<OptionalTensor>& variable_args,
    OpKernelContext* ctx,
    const XlaCompiler::CompilationResult** compilation_result,
    xla::LocalExecutable** executable) {
VLOG(1) << "XlaCompilationCache::Compile" << DebugString();
if (VLOG_IS_ON(2)) {
    VLOG(2) << "num_inputs=" << ctx->num_inputs()
        << " num_constant_args=" << num_constant_args
        << " num_variable_args=" << variable_args.size();
    for (int i = 0; i < ctx->num_inputs(); i++) {
        TensorShape shape = ctx->input(i).shape();
        VLOG(2) << i << ". dtype=" << DataTypeString(ctx->input_dtype(i))
            << " present=" << ctx->has_input(i)
            << " shape=" << shape.DebugString();
    }
    for (const OptionalTensor& variable : variable_args) {
        VLOG(2) << "variable present=" << variable.present
            << " type=" << DataTypeString(variable.value.dtype())
            << " shape=" << variable.value.shape().DebugString();
    }
}

```



## Build

### Build Tools



[Jenkins](#) - Jenkins is an open source automation server, allowing organizations to accelerate their software development through automation. Jenkins manages and controls software delivery processes throughout the DevOps lifecycle, including build, test, operate and deploy. Set up Jenkins to watch for any code changes on GitHub, Bitbucket or GitLab, and automatically do a build with tools like Maven and Gradle. Utilize container technologies such as Docker and Kubernetes, initiate tests, and then take actions like rolling back or rolling forward in production.



**Maven** - Maven is a build automation tool used primarily for Java projects but can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. The Maven project is hosted by the Apache Software Foundation.

Teams can use Maven's project object model (POM) and set of plugins to build projects with a unified build system. Once your team is familiar with how one Maven project builds, you'll know how all Maven projects build, saving time when trying to navigate numerous projects.



**Visual Studio** - Visual Studio for Windows and Mac has built-in compiler tools for all .NET languages. Use it to create builds immediately and test them in a debugger; run multi-processor builds for C++ and C# projects; customize different aspects of the build system. You can use its MSBuild command line tool to automate builds in your CI/CD pipeline, or use the CMake tool to run C++ builds for Windows and Linux.



**Gradle** - Gradle is an open source build automation system that helps teams build, automate and deliver better software faster. Developers can use Gradle to write in Java, C++, Python, etc., and package for deployment on any platform. Gradle's ecosystem of plugins and integrations help teams scale automation. Model, integrate and systematize the delivery of your software from end to end, and scale-out development with fast builds. Gradle offers many features from compiling avoidance to advanced caching, to enabling continuous delivery.



## Test

### Testing

## Key Findings

Testing frameworks are integral to a successful DevOps strategy as they help provide high-level guidelines for creating and designing test cases. Testing frameworks provide a

combination of practices and tools designed to help development and QA teams test more effectively.

Testing tools help enterprise teams improve test speed, increase accuracy, reduce maintenance costs, and lower risk of error. With an efficient DevOps workflow, developers can use these tools during continuous delivery to reduce oversight by QA.

## Tools



**JUnit** - JUnit is an open source unit testing framework for Java. JUnit is useful for developers who work in test-driven environments because it helps find bugs early in the code, which makes code more reliable. Unit testing also forces developers to spend more time reading code than writing. This results in more readable, reliable and bug-free code, which builds confidence during development.

```
15 public class JunitAnnotationsExample {  
16  
17     private ArrayList<String> list;  
18  
19     @BeforeClass  
20     public static void m1() {  
21         System.out.println("Using @BeforeClass , executed before all test cases ");  
22     }  
23  
24     @Before  
25     public void m2() {  
26         list = new ArrayList<String>();  
27         System.out.println("Using @Before annotations ,executed before each test cases ");  
28     }  
29  
30     @AfterClass  
31     public static void m3() {  
32         System.out.println("Using @AfterClass ,executed after all test cases");  
33     }  
34  
35     @After  
36     public void m4() {  
37         list.clear();  
38         System.out.println("Using @After ,executed after each test cases");  
39     }  
40  
41     @Test  
42     public void m5() {  
43         list.add("test");  
44     }  
45 }
```

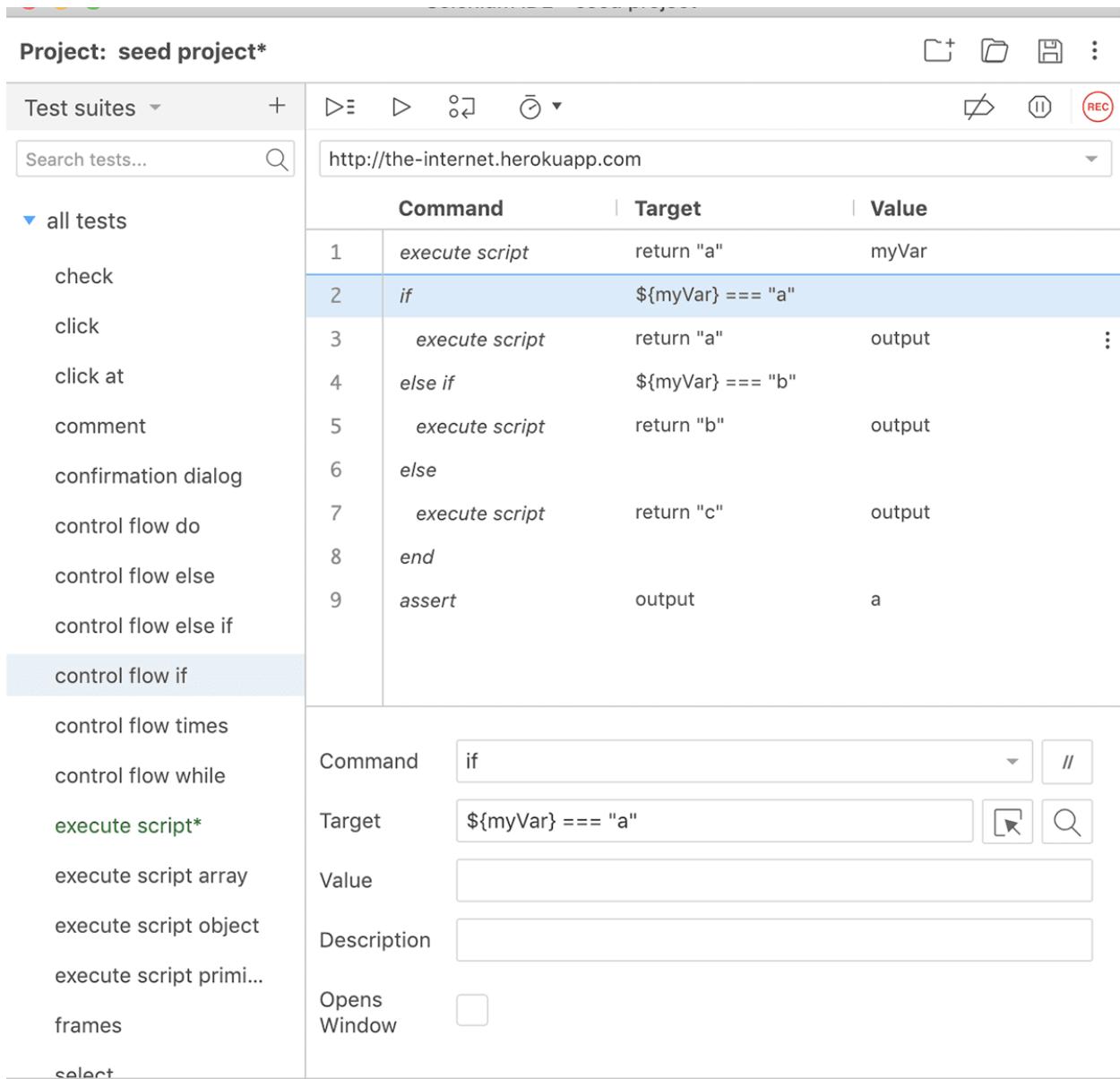


**Selenium** - Selenium is a suite of tools for automating web browsers. It provides a playback tool for authoring functional tests without the need to learn a test scripting language.

Selenium WebDriver is a collection of language-specific bindings to drive a browser. It helps QA teams create robust, browser-based regression automation suites and tests, and scale/distribute scripts across many environments.

Selenium IDE is a Chrome and Firefox add-on that will do simple record-and-playback of interactions with a browser. It helps QA teams create quick bug reproduction scripts and scripts for automation-aided exploratory testing.

Selenium Grid is ideal for QA teams who want to scale by distributing and running tests on several machines while managing multiple environments from a central point. This makes it easy to run tests against a variety of browsers and operating systems.



The screenshot shows the Selenium IDE interface with the following details:

**Project: seed project\***

**Test suites** + ▶ ⌂ ⌂ ⌂ ⌂ REC

Search tests...

http://the-internet.herokuapp.com

**all tests**

	Command	Target	Value
1	execute script	return "a"	myVar
2	if	<code>#{myVar} === "a"</code>	
3	execute script	return "a"	output
4	else if	<code>#{myVar} === "b"</code>	
5	execute script	return "b"	output
6	else		
7	execute script	return "c"	output
8	end		
9	assert	output	a

**control flow if**

Command: if  
Target: `#{myVar} === "a"`  
Value:  
Description:  
Opens Window:

**control flow times**

**control flow while**

**execute script\***

**execute script array**

**execute script object**

**execute script primitive**

**frames**

**select**



**Jest** - Jest is a JavaScript testing framework that works with Babel, TypeScript, Node, React, Angular and Vue. Jest is fast and requires little configuration. It uses snapshot testing to keep track of large objects; snapshots live either alongside your tests or embedded inline.

```
IL ./module2.spec.js
< some other test (3ms)
> skipped 1 test

> some other test

expect(value).toMatchSnapshot()

Received value does not match stored snapshot 1.

- Snapshot
+ Received

Object {
-   "bar": 40,
+   "bar": 55,
}

12 |
13 | test('some other test', () => {
> 14 |   expect(data2).toMatchSnapshot()
15 | });
16 |

  at Object.<anonymous>.test (module2.spec.js:14:17)

1 snapshot test failed.
snapshot Summary
1 snapshot test failed in 1 test suite. Inspect your code changes or press `u` to update them.

Interactive Snapshot Progress
1 snapshot remaining, 1 snapshot updated, 2 snapshots skipped

Usage
Press u to update failing snapshots for this test.
Press s to skip the current test.
Press q to quit Interactive Snapshot Mode.
Press Enter to trigger a test run.

Usage: Press w to show more...
```



**PHPUnit** - PHPUnit is a programmer-oriented testing framework for PHP. It is an instance of the xUnit architecture for unit testing frameworks. Many modern PHP frameworks come with PHPUnit integration, including Laravel, Symfony and CakePHP. CMS's including Wordpress and Drupal also use it for testing.

```
unit.Ignore;
unit.Test;

JUnitAnnotationsExample {

    ArrayList<String> list;

    class
    static void m1() {
        em.out.println("Using @BeforeClass , executed before all test cases ");

        void m2() {
            = new ArrayList<String>();
            em.out.println("Using @Before annotations ,executed before each test cases ");

            ass
            static void m3() {
                em.out.println("Using @AfterClass ,executed after all test cases");

                void m4() {
                    .clear();
                    em.out.println("Using @After ,executed after each test cases");

                    void m5() {
                        .add("test");
                        rtFalse(list.isEmpty());
                        rtEquals(1, list.size());
                    }
                }
            }
        }
    }
}
```



## Release

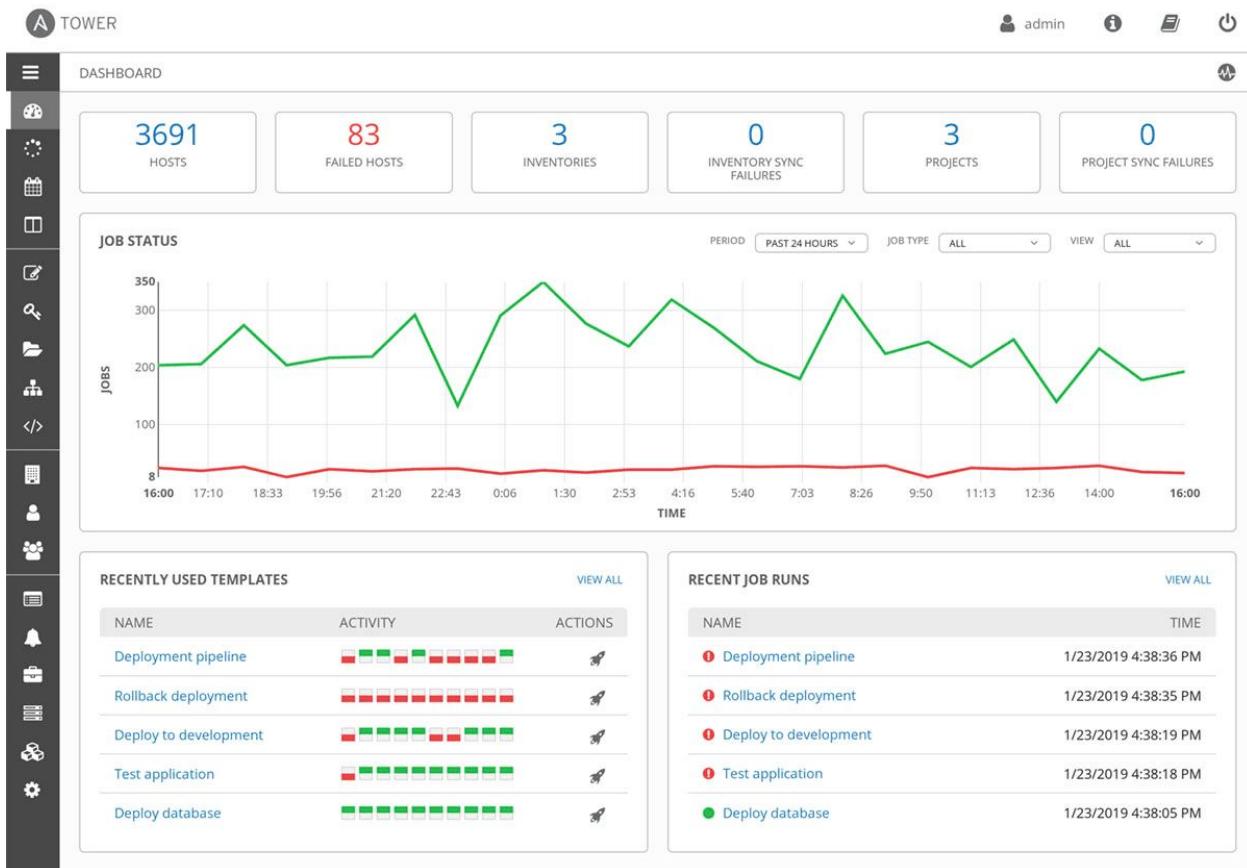
### Configuration Management



**Ansible** - Ansible provides simple solutions for automating a variety of systems from infrastructure, applications, networks, containers, security, and more. Ansible configurations are straight-forward data descriptions of your infrastructure that make it easy for everyone to understand. Managing systems requires nothing more than a password or SSH key—no install agent software required—avoiding the common

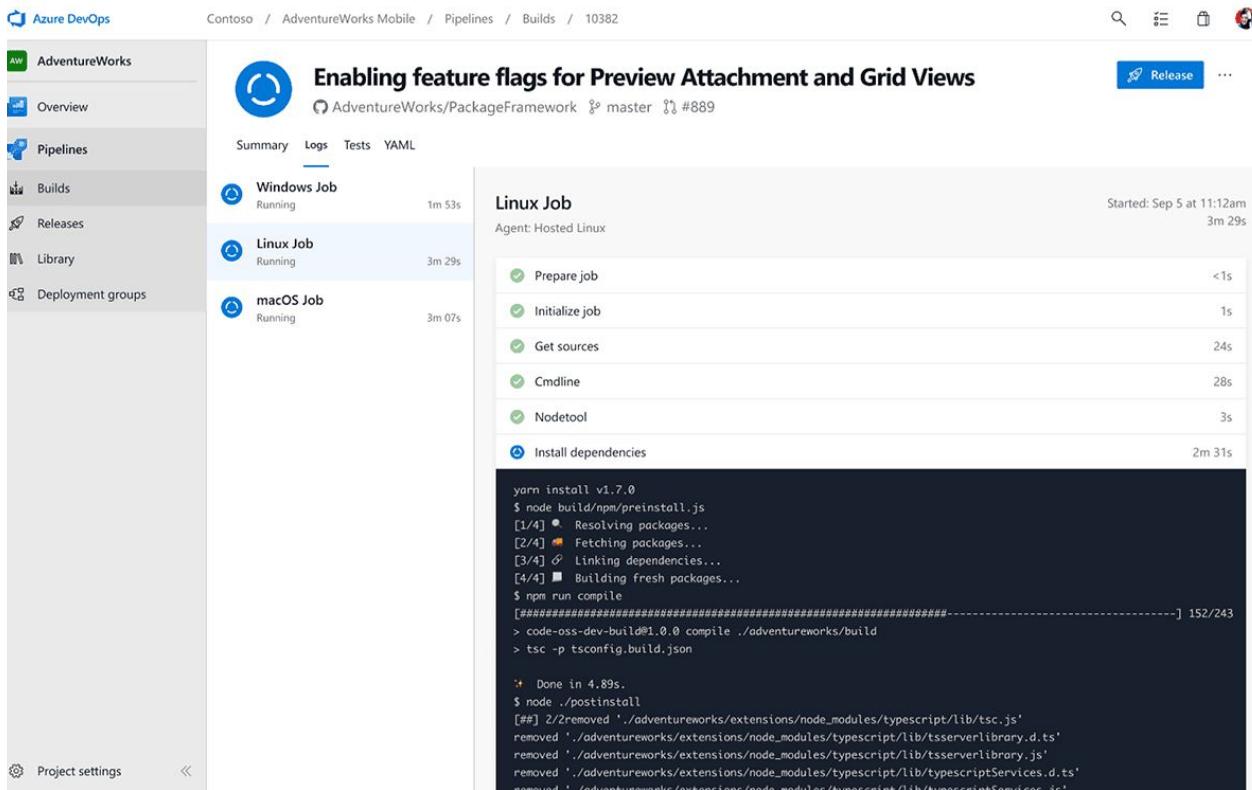
automation system problem of "managing the management." Ansible relies on OpenSSH, which is the most secure remote configuration management system.

Not only does Ansible rank as the #1 configuration management tool on this DevOps report, in Stack Overflow's 2019 Developer Survey, [62% of developers who use Ansible love it.](#)



[Azure DevOps](#) - Azure DevOps is a set of DevOps solutions created by Microsoft. Simplify configuration management in the cloud with Azure Automation. Manage resource configuration across your entire system to enforce desired states, roll out configuration updates, and automate resolution of unexpected changes and issues.

Author and manage PowerShell configurations, import configuration scripts, and generate node configurations—all in the cloud. Use Azure configuration management to monitor and automatically update machine configuration across physical and virtual machines.



The screenshot shows the Azure DevOps Pipelines interface for the AdventureWorks project. The pipeline consists of three parallel jobs: Windows Job (Running, 1m 53s), Linux Job (Running, 3m 29s), and macOS Job (Running, 3m 07s). The Linux Job details are expanded, showing steps like Prepare job, Initialize job, Get sources, Cmdline, Nodetool, and Install dependencies, each with its duration. The logs for the Linux job show the command-line output of the build process, including yarn install v1.7.0, node build/npm/preinstall.js, and npm run compile.

```

yarn install v1.7.0
$ node build/npm/preinstall.js
[1/4] 🚀 Resolving packages...
[2/4] 🛡 Fetching packages...
[3/4] ⚙ Linking dependencies...
[4/4] 🏺 Building fresh packages...
$ npm run compile
[########################################] 152/243
> code-oss-dev-build@1.0.0 compile ./adventureworks/build
> tsc -p tsconfig.build.json

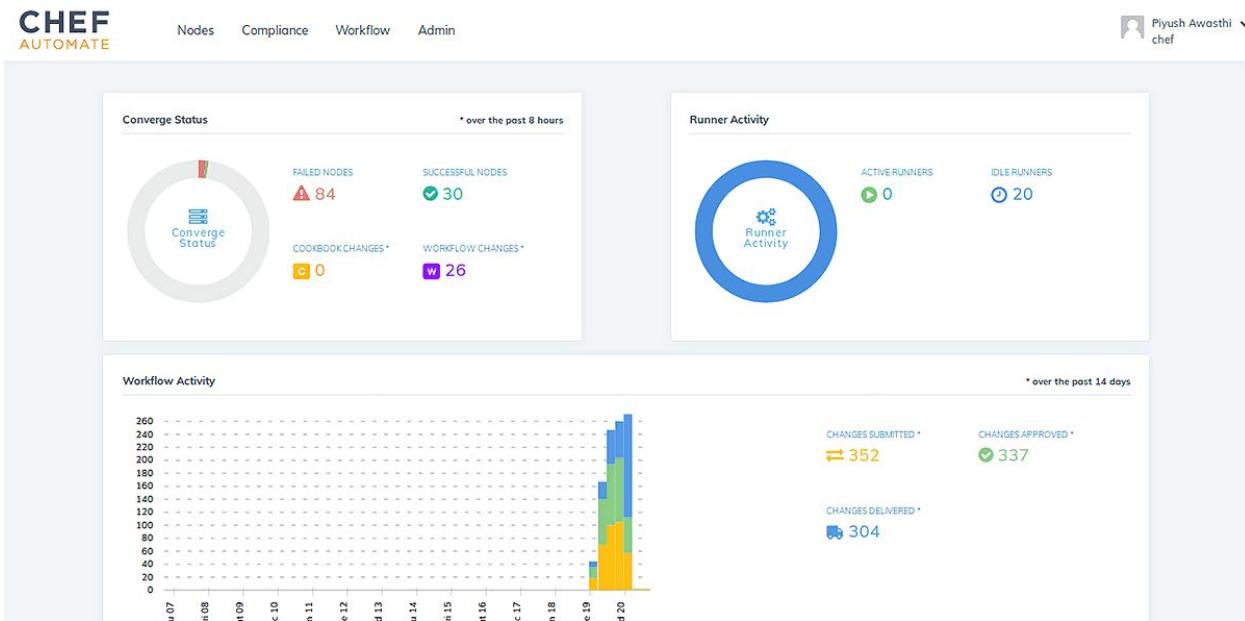
⚡ Done in 4.89s.
$ node ./postinstall
[#] 2/2 removed './adventureworks/extensions/node_modules/typescript/lib/tsc.js'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverlibrary.js'
removed './adventureworks/extensions/node_modules/typescript/lib/typescriptServices.d.ts'
removed './adventureworks/extensions/node_modules/typescript/lib/tsserverClientServices.js'

```



**Chef** - Chef is a configuration management tool for dealing with machine setup on physical servers, virtual machines and in the cloud. It enables continuous automation across IT processes, helping enterprise teams work more efficiently. Chef Automate leverages Chef, Habitat, and InSpec to create a pipeline that can cross both internal and external boundaries, standardizing environments and processes locally within the data center and up in the cloud.

DevOps teams that turn configuration into code are able to leverage the same tools and processes for their applications to efficiently prepare environments to run applications.



## Continuous Integration/Delivery (CI/CD)

### Key Findings

Continuous integration (CI) is a DevOps practice that involves merging code several times a day in a shared repository, and then from that repository, or production environment, building and automated testing are carried out. CI helps enterprise development teams detect errors quickly, reduce integration problems, and avoid compounding problems.

Continuous delivery (CD) adds another layer onto CI, enabling software to be released to production at any time, typically by automatically pushing changes to a staging system. Enterprise development teams have found that the DevOps practice of CD helps ensure every change is releasable and lowers the risk of each release. This allows enterprises to gain a competitive edge by delivering value more frequently and creating tighter customer feedback loops.

### Tools



**Jenkins** - Jenkins is the #1 CI/CD tool according to our DevOps report; it allows organizations to accelerate their software development through automation. Jenkins manages and controls software delivery processes throughout the DevOps lifecycle, including build, test, operate and deploy. Set up Jenkins to watch for any code changes on

GitHub, Bitbucket or GitLab and automatically do a build with tools like Maven and Gradle. Utilize container technologies such as Docker and Kubernetes, initiate tests and then take actions like rolling back or rolling forward in production.



**GitLab** - GitLab prides itself on being one application for the entire DevOps lifecycle. Therefore, GitLab CI/CD is part of the single GitLab application, providing a seamless user experience from planning to deployment. With GitLab CI/CD you can execute builds on Unix, Windows, macOS, and any other platform that supports Go. Plus, it splits builds over multiple machines for fast execution. GitLab CI/CD also offers real-time logging, flexible pipelines, versioned pipelines, autoscaling, build artifacts, Docker support, container registry, and more.



**Azure DevOps** - Azure DevOps is a set of DevOps solutions created by Microsoft which allow developers to go from code to cloud by automating each part of the DevOps process with continuous integration and continuous delivery.

With end-to-end solutions on Azure, teams can implement [DevOps](#) practices in each of the application lifecycle phases: plan, develop, deliver, and operate. Being tightly integrated with both Visual Studio and VS Code makes Azure DevOps easier for developers to work on their CI/CD pipelines.



**Travis CI** - Travis CI is a hosted continuous integration service used to build and test software projects hosted on GitHub. One of the key advantages of using Travis CI over Jenkins is that it's faster to set up: simply login with GitHub, test a project and push to GitHub. If your organization is using GitHub to work on open source projects, this is a great option. Travis CI also integrates with popular communication tools like Slack to keep your development teams up-to-date on build statuses.



## Deploy

### Deployment

## Key Findings

Continuous Deployment is an advanced DevOps practice for enterprises with a mature DevOps process. It goes one step further than Continuous Delivery, by pushing code changes to production automatically, rather than just a staging system. Whether your development teams are at this stage or not, you'll need a tool to help with your deployment strategy.

## Tools



**Jenkins** - Jenkins is not only a build tool, but it's also one of the most widely adopted solutions for continuous delivery. There are tons of plugins that enable Jenkins to integrate with virtually any tool, including all of the best-in-class solutions used throughout the continuous delivery process. Jenkins plugins allow developers to deploy Docker images, deploy to Kubernetes clusters, or across the network. And using its Pipeline plugin, developers can set up pipelines as code and deploy to any service like AWS or Azure using their APIs.



**Azure DevOps** - Azure DevOps is a set of DevOps solutions created by Microsoft. Azure Pipelines is one of the services, and it's used to automate builds and deployment to many cloud providers, especially Azure.

If your enterprise is developing a .NET, Java, Node, PHP, or a Python app, Azure Pipelines can help you set up a highly customizable continuous integration (CI) and continuous delivery (CD) pipeline.



**AWS** - AWS CodeBuild is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to

deploy. There are no servers to provision and scale, or software to install, configure, and operate.

AWS CodeBuild belongs to a family of AWS Code Services, which you can use to create complete, automated software release workflows for continuous integration and delivery (CI/CD). You can also integrate CodeBuild into your existing CI/CD workflow. For example, you can use CodeBuild as a worker node for your existing Jenkins server setup for distributed builds.



**GitLab** - GitLab CI/CD not only tests and builds projects, it can also be used to deploy them in your infrastructure. GitLab provides a full history of your deployments for each environment. It also keeps track of your deployments, so you always know what is currently being deployed on your servers. If you have a deployment service such as Kubernetes associated with your project, you can use it to assist with your deployments.

GitLab helps automate the release and delivery of applications, shortening the delivery lifecycle, streamlining manual processes, and accelerating team velocity. With zero-touch continuous delivery (CD) built right into the pipeline, deployments can be automated to multiple environments like staging and production, and the system automatically knows what to do without being told - even for more advanced patterns like canary deployments. With feature flags, built-in auditing/traceability, on-demand environments, and GitLab pages for static content delivery, you'll be able to deliver faster and with more confidence than ever before.



## Operate

### Containers



**Docker** - Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Docker containers combine software and its dependencies into a standardized unit for software development that includes everything it needs to run: code, runtime, system tools and libraries. Enterprise teams use Docker

containers to ensure applications always run the same, and to make collaboration as simple as sharing a container image.

Not only does Docker rank as the #1 container tool on this DevOps report, in Stack Overflow's 2019 Developer Survey, Docker came in as the #3 "[most common platform](#)," with 35% of professional developers reporting they've developed on the Docker platform. Docker also ranks as the #2 "[most loved platform](#)" in the same report.



**Kubernetes** - Kubernetes is an open source container-orchestration system for automating application deployment, scaling, and management. Originally a Google product, Kubernetes is now maintained by the Cloud Native Computing Foundation.

Docker and Kubernetes aren't direct competitors. Kubernetes is a container orchestrator for container platforms like Docker. All the major cloud providers support it, so if your enterprise is moving applications to the cloud, Kubernetes is a solid choice. It provides a common framework to run distributed systems so development teams have consistent infrastructure from development to production for every project. Kubernetes can manage scaling requirements, availability, failover, deployment patterns, and more.

While Kubernetes is a robust tool, it is also notoriously complex and may add unnecessary or unwanted overhead to your DevOps toolchain. Many public cloud services like AWS and Azure provide some orchestration capabilities that may serve your enterprise's needs.



**AWS** - Amazon Web Services (AWS), as the name implies, is an Amazon service that provides on-demand cloud computing platforms and APIs. If you're using other AWS developer tools to host code, build, test, and deploy your applications to AWS, then consider extending your DevOps toolchain to run containers on AWS.

If your enterprise chooses AWS, you'll set up one of two container orchestrators: Amazon Elastic Container Service (ECS) or Amazon Elastic Kubernetes Service (EKS). If you're familiar with AWS constructs and APIs, ECS is a great place to run your containers. ECS is deeply integrated with AWS services such as Identity and Access Management (IAM), Amazon Virtual Private Cloud (VPC), and Amazon Route 53. If you use Kubernetes, EKS is a secure, reliable, and scalable way to run Kubernetes.

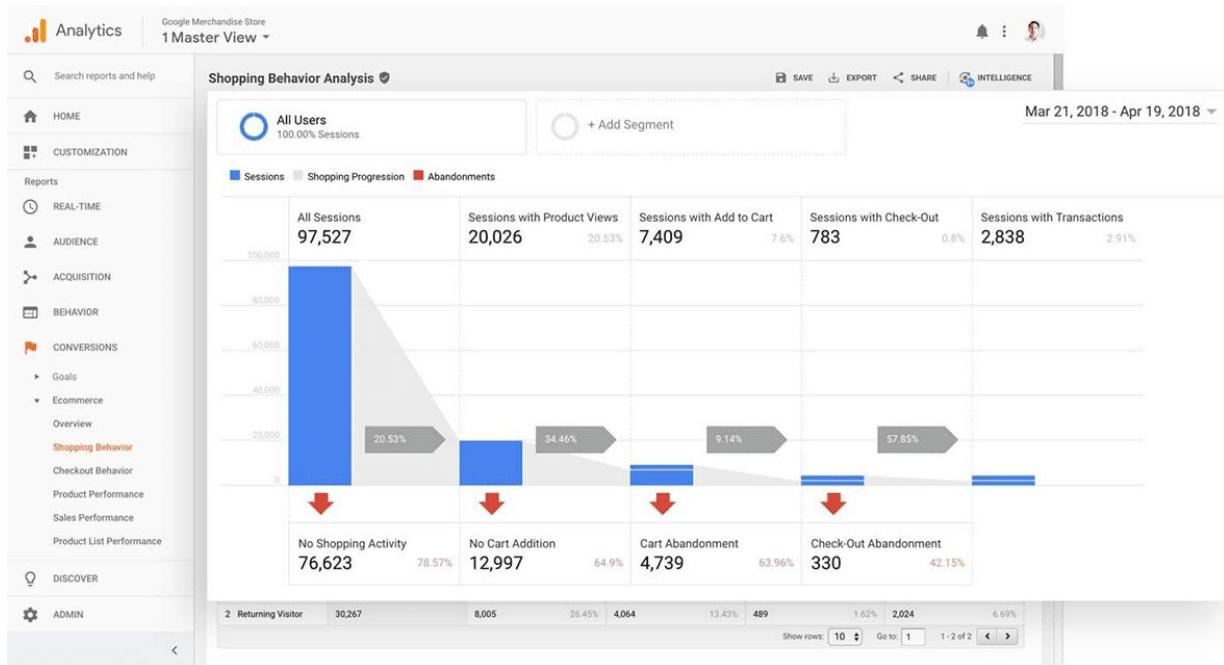


## Monitor

### Analytics/Monitoring



**Google Analytics** - Google Analytics is a powerful analytics and monitoring tool that enables enterprise teams to collect, configure and analyze crucial data. This tool provides insights into user interactions with websites, web applications, Android and iOS apps. Developers can utilize Google Analytics' robust APIs to build comprehensive reports and custom configurations.

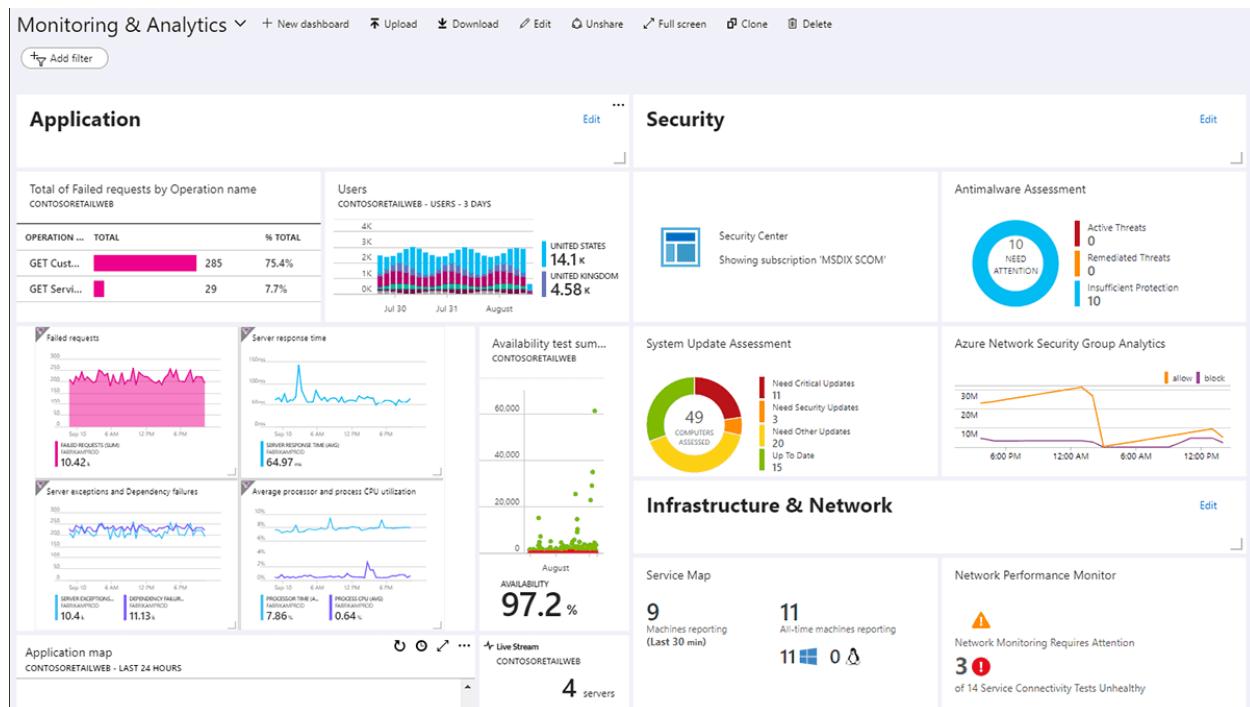


**Grafana** - Grafana is an open-source analytics and monitoring solution that supports dozens of databases natively, to pull in data from all your sources. It's extremely

visual, from heatmaps to histograms, graphs to geomaps, and all kinds of dashboards. Grafana also allows teams to set up alerts and notifications via Slack, etc.

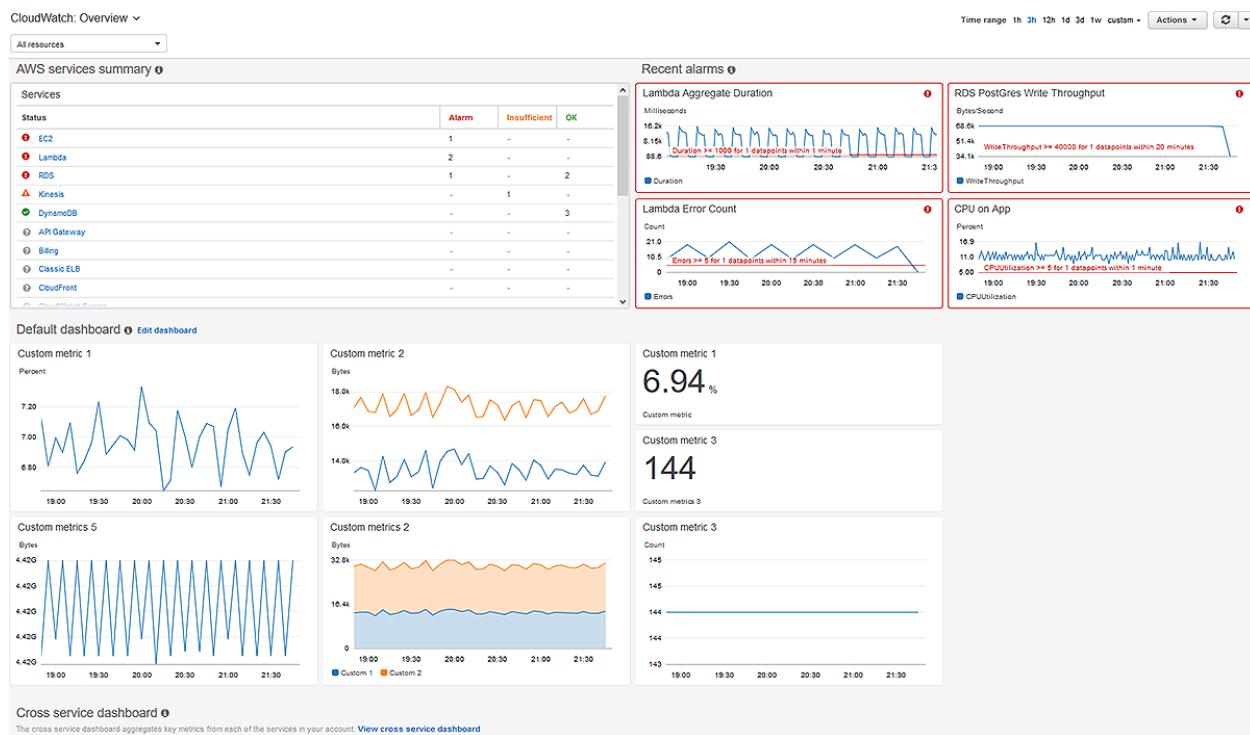


[Azure](#) - Azure Monitor is a Microsoft tool that provides full observability into your applications, infrastructure, and networks. Implement full-stack monitoring, get actionable alerts, and gain insights from logs and telemetry with Azure Monitor.



**AWS** - Amazon CloudWatch is a monitoring and analytics service built for DevOps engineers, developers, site reliability engineers (SREs), and IT managers. CloudWatch provides the data enterprise teams need to take action. Use this tool to monitor applications, respond to performance changes, optimize resources, and get an overview of operational health.

CloudWatch collects monitoring and operational data in the form of logs, metrics, and events, providing insights into AWS resources, applications, and services that run on AWS and on-premises servers.

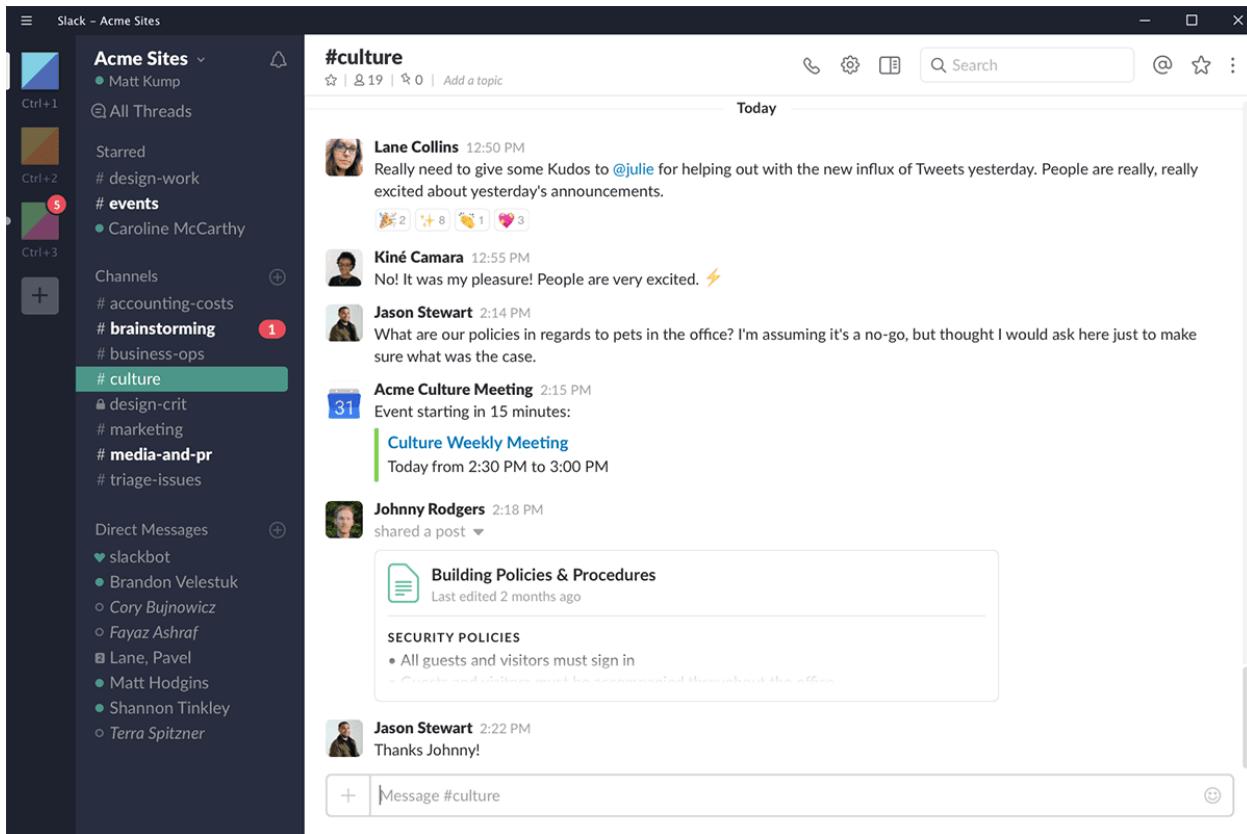


## Communication

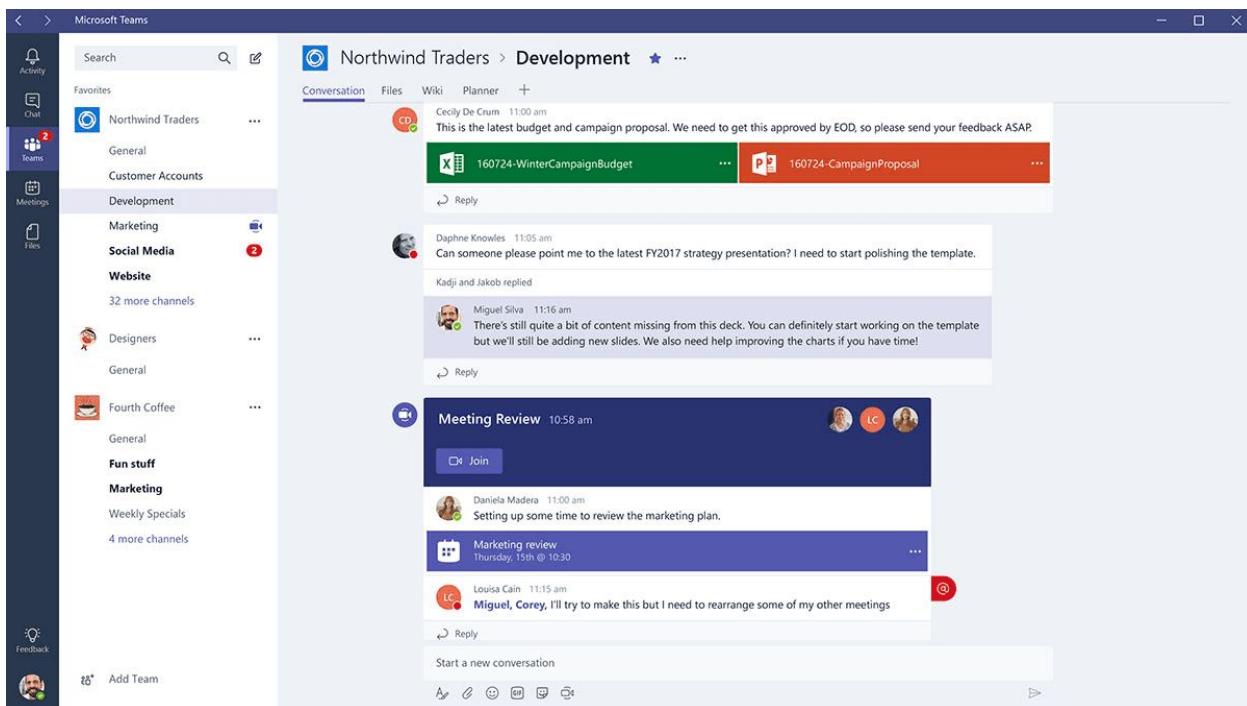


**Slack** - Slack is a cloud-based instant messaging platform, with video calling, that logs the history of conversations and organizes them into channels based on topics. It has taken over as a real-time alternative to email within enterprises. Development teams use Slack to get access to key information and experts company-wide; review and collaborate on projects; track real-time performance data from multiple systems to resolve issues faster; streamline and automate workflows by connecting to other services and platforms.

Slack integrates with popular project management and issue tracking tools like Jira, Glitch, Trello and Trello to monitor new bugs and tasks as they come in, and to respond without context switching. Slack also integrates directly with GitHub, GitLab and Bitbucket.



**Microsoft Teams** - Microsoft Teams is a communication platform that combines chat, video meetings, file storage, and application integration. If your enterprise is reliant on the Microsoft suite of tools, this is an ideal hub for team collaboration in Office 365. In Teams you can access, share, and edit Word, PowerPoint, and Excel files in real time.



## Conclusion

Leading enterprises are undergoing DevOps transformations to gain a competitive advantage by delivering their technologies faster and more securely. A DevOps transition involves many facets, including evolving employee mindsets, teaching new skills, and introducing the appropriate tools. This report focused summarizing the best DevOps tools from 2,700 trailblazers who have already successfully implemented them.

Share this report with other stakeholders to showcase the importance tools play in a DevOps transformation. From our report findings, it's clear that high-performing software engineers and IT leaders choose useful and usable tools to improve productivity and deliver value during the DevOps transformation. They also automate and integrate tools into their toolchains, freeing up time to spend on new development, and debunking the argument that it's too time-intensive or expensive to implement tools.

Contact [Sales@GitKraken.com](mailto:Sales@GitKraken.com) to get started with GitKraken DevOps tools.



**Git GUI**



**Glo Issue Boards**



**Timelines**