CASE STUDY

Brave Software and TeamCity

Building a CI/CD Solution They Can Rely On

brave

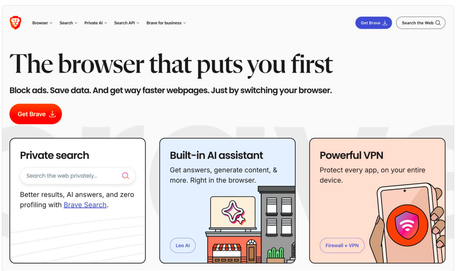
Industry	Software / Browser
Location	International
Company size	< 200
Products used	TeamCity

brave.com/↗

The setup

Brave Software is best known for its privacy-first browser (with over 115 million monthly active users) and Brave Search (a private search engine).

Brave Search operates with a fully independent index – it builds and serves its own search database instead of depending on larger search providers' indexes, handling billions of queries per month. This is something only three companies in the world can claim – Google, Bing, and Brave Search.



The company has fewer than 200 employees, with a small DevOps team with many responsibilities, one of which is CI/CD. Their stack isn't simple – the browser combines C++, TypeScript, Java, Swift, Rust, and Python, and relies on tools like GN, Ninja, Siso, and Remote Build Execution to keep pull request builds fast.

This is a large-scale, multiplatform, performance-sensitive system that must work every day across hundreds of builds.

The problem

The previous CI system used for the browser was unreliable, with bugs going unfixed and plugins going unmaintained.

The instability disrupting development and the effort needed to keep the system going became too much.

With a CI setup for Linux, Windows, macOS, Android, and iOS, the team needed a stable platform that would provide the basics and get out of their way.

> **Wojciech Knapik**
> Staff DevOps Engineer, Brave
>
> Without a high chance of quick upstream fixes, we had to switch plugins, give up functionality, or implement and maintain ever more workarounds, including reimplementing node provisioning.

Looking for a new tool

The team looked at alternatives, both self-hosted and managed, but none proved to be the right fit.

Most options lacked the flexibility they needed for a complex, multiplatform browser build. Others were too expensive, no longer supported, or didn't provide the needed features.

While Brave generally prefers open-source solutions, there was no competing open-source tool that could meet their requirements.

> **Wojciech Knapik**
> Staff DevOps Engineer, Brave
>
> Reliability and feature richness were the primary criteria. We also wanted to make sure that when we do run into bugs, there will be someone there to look at our reports.

The solution

That's when they decided to move to TeamCity, starting with PR builds.

The rollout went platform by platform, enabling builds in TeamCity and disabling them in the old system.

Configuration is managed entirely via Kotlin DSL, with changes going through pull requests, reviews, and automated checks.

> **Wojciech Knapik**
> Staff DevOps Engineer, Brave
>
> A must-have requirement was a robust DSL for versioning and reviewing configuration changes.

At the infrastructure level, Brave moved toward ephemeral EC2 agents provisioned via TeamCity cloud profiles, with persistent pre-populated workspace volumes for incremental builds.

They also built advanced orchestration logic using TeamCity's REST API, allowing a PR trigger build to determine and fan out downstream builds dynamically.

The system now works without requiring extra attention, so the team can focus on building the CI cycles on top of it.

Early results

When asked what they liked most about TeamCity, the answer was simple: "We can assume things will work and won't randomly break."

That reliability has changed the DevOps team's day-to-day reality. Instead of spending their time fixing CI issues, they're putting that time into making the system better by supporting new platforms and products, splitting large test suites across multiple machines, dockerizing build steps, and more.

The team is taking full advantage of the various security-related features of TeamCity. Google SSO takes care of authentication; the GitHub App and AWS connections provide ephemeral, short-lived credentials with minimal permissions, replacing long-lived tokens and node roles; and the built-in Let's Encrypt support completes e2ee in transit for the service.

At the same time, they openly point out what still needs work in TeamCity. To address the issues that the team mentioned, recent and upcoming versions of TeamCity are focused on further optimizing UI performance. To stay up to date on further TeamCity development, subscribe to our blog.

Final thoughts

Brave's TeamCity journey has only just begun. But the basics are now in place: pull requests are enforced, configuration is managed clearly in code, incremental builds run on short-lived cloud machines, and there's a stable platform underlying all of it.

The team can now focus on improvements to get the most out of the system. To learn more about past build and release challenges Brave's team overcame, check out "Building and releasing the Brave browser to tens of millions of users", a blog post by Mihai Plesa (VP of DevOps at Brave).

Also, if you're looking for new opportunities, the Brave team is hiring!

Similar Customer Stories

Junie

Junie

Danila Savenkov, Junie Team Lead

If we tried running this locally, it just wouldn't be realistic. A single evaluation would take a full day. That's why we use TeamCity: to do everything in parallel, isolated environments, and to ensure the results are reproducible.

Read the case study→

PICNIC

Picnic

Ivan Babiankou, Staff Software Engineer, Picnic

We were looking for a managed solution for all our CI use cases. Additionally to that, we needed self-hosted agents to control which software we're running, and which exact tooling is in use. TeamCity Cloud with self-hosted agents provided a tailor-made solution that our team of more than 300 engineers happily uses and that pushes our productivity to the next level.

Read the case study→

Gradle

Gradle

Piotr Jagielski, VP of Engineering, Gradle Build Tool

We've relied on TeamCity as our CI system of choice for over a decade. It provides all the features we need out of the box. We also appreciate its reliability and like Kotlin DSL for configuring our build pipelines.

Read the TeamCity case study→