Error Monitoring & Crash Reporting Basics

SMARTBEAR

Contents

How to Catch Bugs & When to Call an Exterminator

Software bugs share many similarities with the physical variety. While you may not notice the first few ants on the counter, problems quickly become apparent in just a few hours. The key to stopping an infestation is proactively monitoring and setting traps because catching bugs late can become a much more difficult and expensive endeavor.

Similarly, software bugs are much easier and cheaper to catch early in the development cycle. According to Celerity, the cost of a bug might be $100 if found in the gathering requirements phase, $1,500 if found in the quality assurance testing phase, and $10,000 if found in production. And in aggregate, software bugs cost the U.S. economy nearly $60 billion annually.

Let's look at how error monitoring and crash reporting can help you identify and fix bugs before they take a toll on your application and business.

What Do We Mean by Error Monitoring and Crash Reporting?

Errors are inevitable when developing web and mobile applications. While developers may insist, "it works on my machine," modern applications rely on a complex web of microservices and dependencies delivered to countless different types of browsers and devices. As a result, error monitoring and crash reporting have become significantly more difficult.

Most web and mobile applications include both front-end and back-end components, meaning effective error monitoring solutions must look across both to discover the root cause of a problem. Moreover, the frontend might involve hundreds of different browsers and devices, while the backend might have an intricate cloud services network.

Frontend

Front-end error monitoring involves identifying and fixing errors from the user's perspective. For example, you might have to troubleshoot a client-side JavaScript error in a React component or track down why a client-side API call is timing out. Front-end error monitoring aims to ensure everything looks good and functions properly.

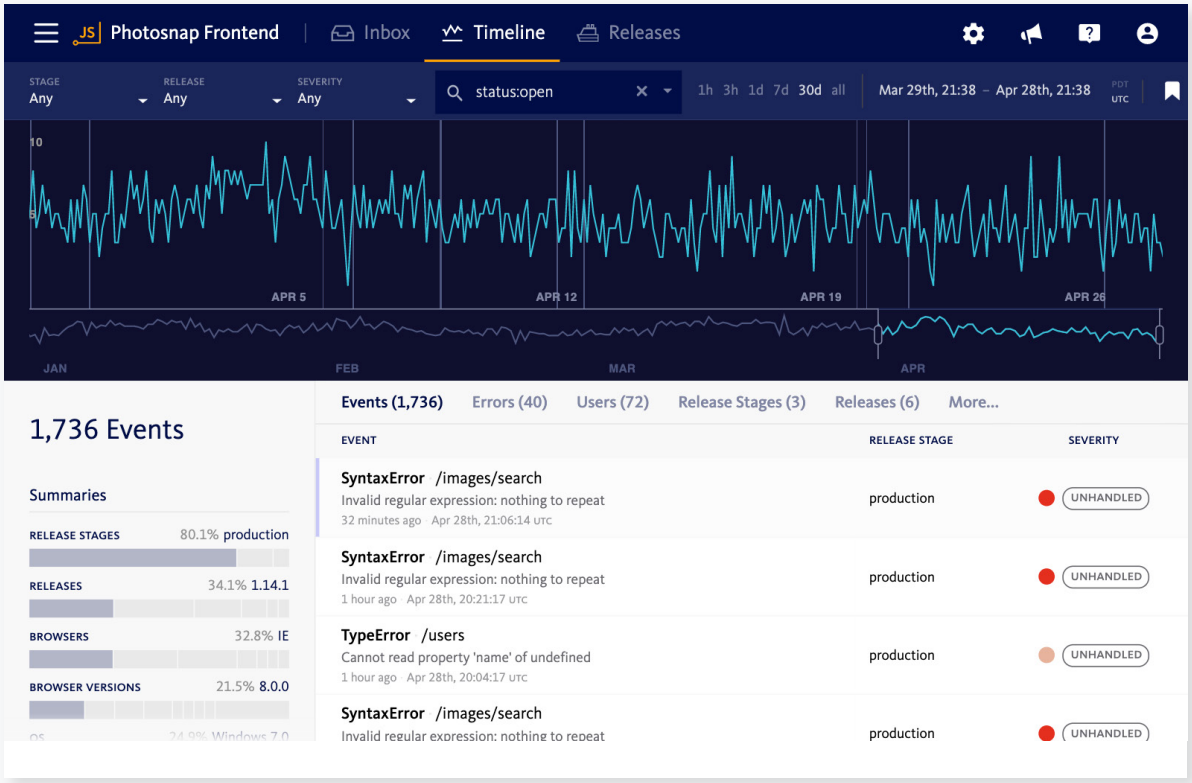
Synthetic Monitoring

Synthetic monitoring is the process of simulating user interactions to catch and fix errors before real users see them. The most common type of synthetic monitoring is end-to-end test automation using frameworks like Selenium or Appium.

Selenium enables developers to write code-based instructions that execute in headless web browsers, inside web views, or on simulators and raise an alert when new code changes break critical workflows. Meanwhile, Appium drives native apps, making it easy to run automated tests on simulators.

Other solutions, like TestComplete, provide a more comprehensive automated UI testing framework that supports desktop, web, and mobile applications. Unlike script-based testing solutions, the platform's scriptless Record and Replay or keyword-driven tests enable anyone to build robust UI tests that catch errors before real users see them. In addition, an AI-powered object recognition system helps make the tests more robust than other platforms.

BitBar and other solutions also enable these tests to run on real devices rather than headless browsers or simulators. That way, software teams can mimic actual customer devices and easily reproduce errors that might occur on specific devices. Developers also can integrate these real devices into continuous integration and deployment (CI/CD) processes to automatically run end-to-end tests before each major deployment to spot any issues.



Bugsnag makes it easy to see real-time events, errors, users, and releases. Source: Bugsnag

Real User Monitoring

Real user monitoring is monitoring real user sessions for errors. Many organizations rely on log monitoring solutions, like Splunk, or Application Performance Monitoring (APM) software, like New Relic, that monitors errors in production. These solutions cater to DevOps and infrastructure teams that monitor performance issues and adjust resources to accommodate traffic spikes or other developments.

More recently, application stability management (ASM) platforms, like Bugsnag, have emerged to help loop in developers. Rather than catering to DevOps or infrastructure teams, your engineering teams are the target audience for these solutions. They automatically detect when software breaks in the hands of a customer (for example, crashes or unhandled exceptions) and provide rich insights into application stability and how to improve the user experience.

A Combined Approach

Many organizations combine these approaches since it's impossible to catch every error before it reaches production. For example, you might use automated end-to-end tests to ensure new code doesn't contain any obvious errors before deploying it. Then, you might use an ASM platform to detect client-side errors in production and provide developers with the diagnostic information they need to fix them.

Backend

Back-end error monitoring involves identifying and fixing errors from the business or infrastructure perspective. For example, you might identify an N+1 query that leads to costly performance issues or discover unhandled exceptions causing a poor user experience. Back-end error monitoring aims to maximize performance and handle edge cases.

Some of the most common back-end errors include:

- **Application Errors:** Occur when a framework throws an error or your own code contains errors. For example, you may have a syntax error that causes an exception within a certain user workflow.
- **Dependency Errors:** Occur when a third-party package or dependency causes an error. For instance, an HTTP dependency might have an error that causes requests to time out or responses to be invalid.
- **Performance Errors:** Typically slow down an application from a user standpoint while increasing infrastructure costs. For example, a faulty piece of code might contain an N+1 query that leads to excessive database queries.
- **Network Problems:** Occur when internal or third-party services are unreliable or unreachable. For instance, a problem with DNS or CDN systems might lead to availability issues with an application.
- **Hardware Problems:** Occur when physical systems experience problems or there's a lack of cloud resources available. For example, a lack of CPU instances or insufficient memory could lead to performance issues or errors.

Third-Party Monitoring

According to Synopsys, modern applications use more than 500 open source libraries and components. In addition to potential security vulnerabilities, any errors or performance issues in these dependencies can lead to problems in applications that use them. As a result, effective error monitoring strategies must involve watching dependencies for potential problems to ensure they're up to date and minimize any issues.

In addition to dependencies, many applications rely on third-party APIs. Proper error handling can help mitigate many potential errors, but quickly catching these errors can help you spot edge cases. Open source tools like SoapUI enable developers to build functional and load tests to ensure third-party APIs work as expected while making it easy to create mock APIs to keep automated tests running smoothly.

Fortunately, many application performance monitoring and ASM solutions can help track down issues with dependencies. For example, Bugsnag provides a complete stacktrace for errors, leading developers straight to any dependency issues. GitHub and other source control tools also provide automated dependency alerts when they encounter potential security vulnerabilities in a codebase.

Infrastructure & Device Monitoring

APM solutions add an extra layer of support by monitoring applications and their underlying infrastructure for performance problems, networking errors, or hardware issues. By monitoring server loads and performance metrics, these solutions can help point out abnormalities and enable DevOps teams to fix problems before they become too big by scaling up server instances or making other changes.

Meanwhile, crash reporting tools can help notify development teams when problems occur on user devices, enabling a quick resolution. Since most users don't take the time to report crashes or other problems that occur within an application, these tools are the only way for development teams to identify and address these problems effectively.

Full-Stack, End-to-End Monitoring

Front-end and back-end components may be built with different technologies, but they rely on each other to deliver working software to the customer. When monitoring errors, a full-stack approach makes it much easier to trace bugs across the technology stack and identify the root cause of the problem without relying on separate solutions and teams.

Behavior-Driven Development (BDD) and Test-Driven Development (TDD) provide a good starting point for test automation by paving the way for a robust test suite. Using these methods, development teams can minimize the number of errors arising from miscommunications between business and technical teams while creating a robust base layer of automated tests.

When problems reach production customers, a full-stack error monitoring solution like Bugsnag might enable you to trace a JavaScript error in a React component across a Ruby on Rails back-end technology stack to diagnose the root cause of the problem. That way, there's no need to involve two different teams or coordinate the results from two different tools.

Full-stack error monitoring solutions also tie in infrastructure to provide a more complete picture. For instance, they might monitor hosts, containers, and serverless functions, enabling development teams to diagnose problems that arise or worsen by infrastructure issues (e.g., low memory or insufficient CPU cycles) rather than code-based issues.

Summary

Errors are an inevitable part of the software development lifecycle. With the growing complexity of modern applications and infrastructure, error monitoring and crash reporting solutions have become essential to maintain a high level of quality. We generally divide these solutions into front-end and back-end technologies, but full-stack error monitoring solutions can help troubleshoot problems across your technology stack.

How Error Monitoring & Crash Reporting Solve Your Bug Problems

Software teams are under constant pressure to increase development velocity without negatively impacting the user experience. That way, sales and marketing teams can onboard new customers while customer success teams keep customer attrition levels low. But, of course, the reality is often a struggle between product and marketing teams.

Error monitoring and crash reporting solutions are essential to bridging the gap between these two teams and efficiently achieving both goals.

Optimal User Experience

Most people agree that user experience is essential to success, particularly as customer expectations continue to rise.

According to Econsultancy, one in four visitors will abandon a website that takes more than four seconds to load and 64% of shoppers will shop elsewhere next time if they're dissatisfied with their visit to your site. And DevOps estimates that 96% of mobile app users write bad reviews of sub-par apps, with 42% of 0-star apps mentioning bugs.

Error monitoring enables you to quickly find and fix issues immediately after they happen. As a result, customers will rarely encounter deal-breaking errors and have a better overall user experience that leads to more engagement and higher satisfaction. Happier customers are more likely to refer their friends and colleagues, leave better reviews in app stores, and consume fewer customer support resources.

For instance, Lyft recognized that its existing mobile crash reporting tool hindered its ability to deliver a great user experience. Despite its best efforts to test and debug its apps, unreliable crash reports and inefficient debugging workflows led to production users seeing errors.

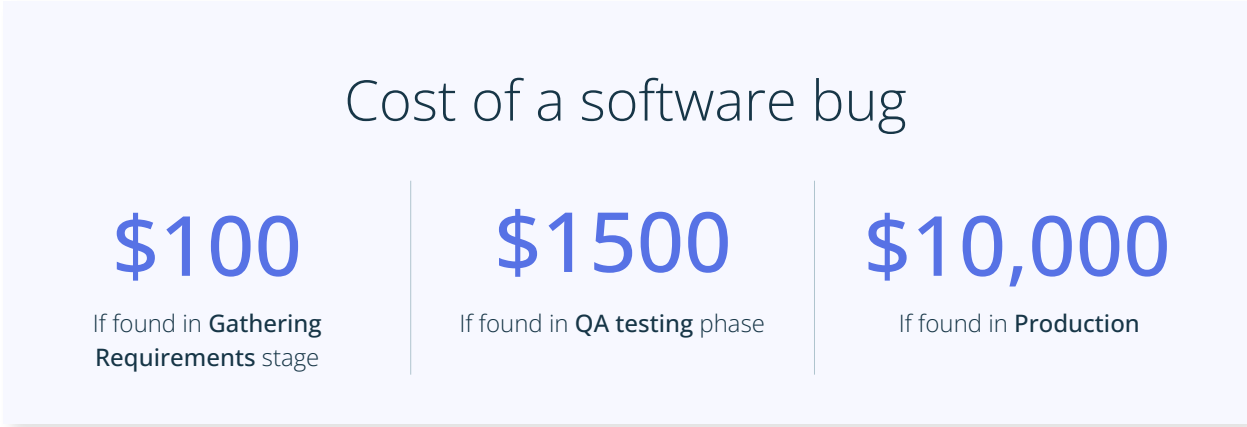
With Bugsnag's error monitoring, the company's developers decreased the time they spend troubleshooting errors by 30% and found them before they impacted users.

Better Team Productivity

Most developers will confirm that fixing bugs is stressful and time-consuming. Without detailed information about bugs or factors leading up to a crash, developers spend a lot of time trying to recreate bugs and track down the causes of an error. According to DevOps, investigating and fixing errors wastes 60% of developer time. And worse, it's usually in code they didn't even write, taking a toll on your team's morale.

Error monitoring solutions reveal bugs that impact users and help lead developers to the root cause. In addition, the best error monitoring solutions can send specific bugs to specific teams to fix without pinging everyone each time, resulting in less notification fatigue. In fact, according to DevOps, error monitoring solutions can deliver a 40% increase in developer productivity by providing them with the exact location and source of an error.

For instance, 99designs' engineering team needed to standardize its process for troubleshooting errors to more easily sift through the noise and fix errors in an efficient manner.



The cost of a software bug rises the longer it takes to fix. Source: Celerity

After implementing Bugsnag error monitoring in more than 25 projects, the team can quickly find errors that matter, saving time, improving product quality, and helping to onboard new engineers faster than ever.

Data-Driven Decisions

Product managers must balance the need to build and launch features with maintaining the quality of existing features. In practice, that means deciding between allocating time to features vs. squashing bugs. After all, it's not always practical to fix every bug that arises – especially if it doesn't impact many users or has a benign impact on user experience. To do this, they require high product visibility to make data-driven decisions.

Error monitoring solutions can help product managers make better data-driven decisions. By monitoring the rate of errors, they can determine thresholds where it makes sense to refocus efforts on bug squashing and focus on features until that point. Clear technical debt metrics also can help software teams experience less emotional stress from technical debt by quantifying what's an acceptable level and surfacing potential problems early.

Reduced Revenue Loss

Many errors lead to direct revenue loss for the business. For example, Knight Capital Group famously lost $440 million in 30 minutes due to a software bug in its trading algorithms. Faulty code that impacts an ecommerce checkout process or a software-as-a-service subscription functionality also can lead to direct revenue losses. And these bugs can quickly become costly when left unchecked for long periods.

Error monitoring solutions help prevent revenue loss or churn by quickly notifying development teams when problems arise and providing the background information necessary to debug them quickly. Faster response times also can reduce the impact of the problem significantly while showing customers that the business cares. And in the end, these efforts can keep business and technical teams on good terms.

Summary

Error monitoring solutions have several real-world benefits for software businesses. In addition to ensuring an error-free customer experience, they can help improve developer productivity and morale, enable product managers to make better data-driven decisions, and reduce revenue loss. Overall, these benefits typically translate to an extremely high return on investment for development teams adopting error monitoring tools.

Should I Set Up Traps or Call an Exterminator?

Software teams have several options when implementing error monitoring solutions. For instance, you might rely on unit and integration tests to ensure everything works before deployment and log files to surface any problems that arise afterward. And in many cases, companies build their own internal error monitoring systems.

The build vs. buy decision is a common problem in software businesses, especially because engineers have a penchant for building green field projects.

Factors to Consider When Building

Many software teams begin with homegrown error monitoring solutions built with open source tools, log monitoring scripts, and services like Zapier to stitch everything together. While these solutions work for a minimum viable product (MVP) or early-stage business, error monitoring needs tend to grow more complex over time, making in-house solutions harder to maintain.

Homegrown error monitoring solutions have a few key benefits:

- **Customization:** Building an error monitoring tool from scratch opens the door to limitless customization. For instance, you can build your key performance indicators and metrics, and build tight integrations with your application.
- **Integrations:** Building an error monitoring solution might be the right choice if your software relies on complex integrations. Building your own might be best if off-the-shelf error monitoring solutions don't offer the integrations you need.

But, there are several drawbacks to keep in mind:

- **Complexity:** Error monitoring solutions tend to become increasingly complex and difficult to maintain over time. For example, a monolithic MVP might be simple to monitor for errors, but if the product evolves into a microservices architecture, error monitoring quickly can become a much larger and more difficult task.
- **Opportunity Cost:** Developer hours spent building and maintaining error monitoring solutions aren't focused on the core business. If having superior internal technology isn't a competitive advantage, your development team's efforts may be better spent building and maintaining the business' core software products.

Features: Off-the-shelf error monitoring solutions tend to have more features than the homegrown variety since they're purpose-built with feedback from hundreds or thousands of software businesses. Using homegrown solutions, your team might miss out on features that could help catch more bugs and fix them faster.

What to Look for When Buying

Error monitoring and crash reporting have become a niche software-as-a-service market. Rather than wasting time building in-house tools, software teams can focus on their core competencies and outsource error monitoring to purpose-built solutions. And in many cases, the result is a more efficient development team and a high return on investment.

The key things to look for when buying an off-the-shelf solution include:

- **Features:** Many error monitoring solutions offer basic features, like surfacing exceptions and providing a stack trace. While these features are a great start, the best error monitoring solutions go a step further and offer product managers the metrics they need to make decisions and enhanced diagnostic capabilities `to developers
- **Integrations:** Software teams should ensure the error monitoring solution they choose integrates with their programming languages, frameworks, and infrastructure. Otherwise, they will have to spend valuable resources building custom integrations and keeping them up to date with the platform's APIs.
- **Dev Tools:** Error monitoring tools also should integrate with your software team's existing development tools. For example, the error monitoring solution might create bug issues in Jira automatically or notify the appropriate engineers in Slack to simplify and streamline development workflows.
- **Support:** The installation of error monitoring solutions can be a massive undertaking, depending on the solution and project details. In particular, enterprises with large and complex projects may want to consider error monitoring solutions with the backing of robust onboarding and support teams to avoid any problems.

Summary

Many software teams stitch together their own error monitoring solutions during the early stages of a project. However, as a project grows more complex, off-the-shelf error monitoring solutions often provide the most cost-effective and feature-full option. When choosing an off-the-shelf solution, you should look beyond the price point to the features, integrations, and onboarding and support to find the best fit for your business.

Call the Pros: How Bugsnag Fixes Your Bug Problem

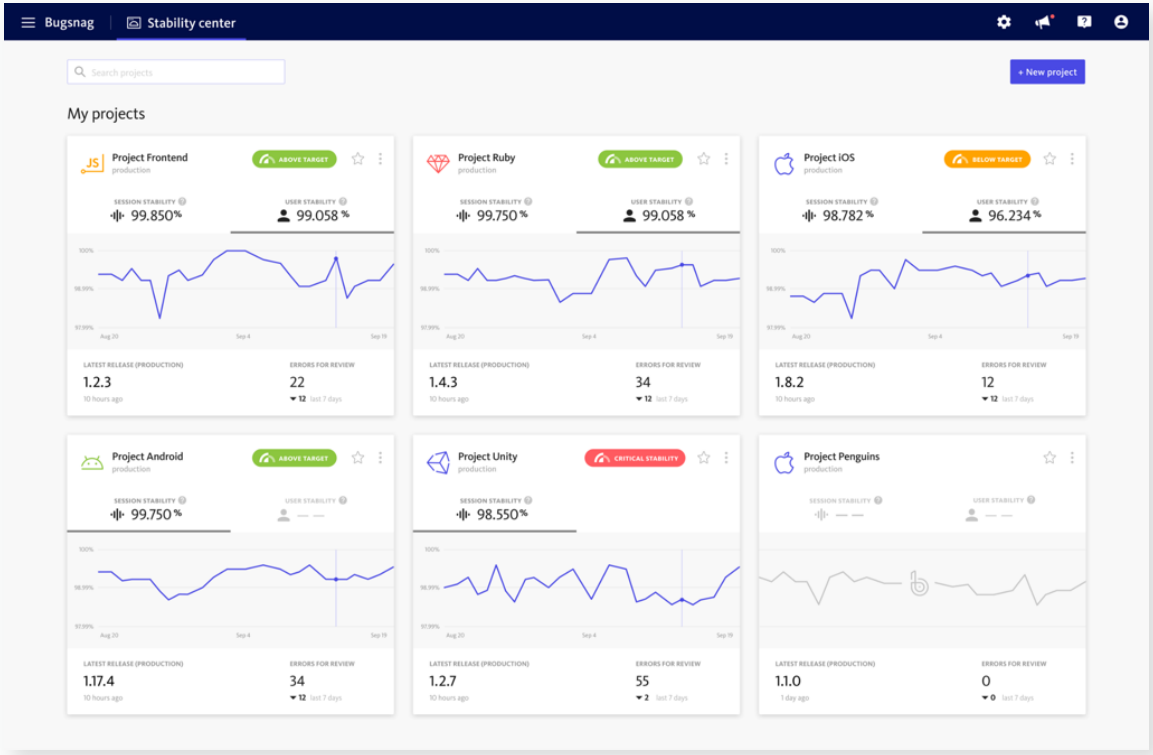
Bugsnag provides developers with the error context they need to quickly fix bugs and get back to building features. At the same time, the platform provides product managers with the confidence to promote new features to production and supports data-driven decisions about when to switch between building features or squashing bugs.

Let's look at Bugsnag's unique three-step process to fix bugs in record time and then explore how you can implement the solution in just one day.

Fix Bugs in Record Time

Most error monitoring solutions take a simple approach involving surfacing errors and providing basic diagnostics. While these capabilities help software teams identify errors and save developers a bit of time, product managers still struggle to determine when to focus on bugs vs. features, and extensive troubleshooting is still necessary.

Bugsnag provides a much more effective solution thanks to its unique three-step approach to quickly find, prioritize, and fix bugs.



Stability Scores make it easy to see application health at a glance. Source: Bugsnag

Step #1: Stabilize

Bugsnag's Stability Scores provide unparalleled visibility into each release. After setting stability targets, or the proportion of crash-free app sessions, the platform will only notify you when stability drops below these critical levels. That way, your team isn't wasting time on bugs that don't matter and can instead focus on building new features.

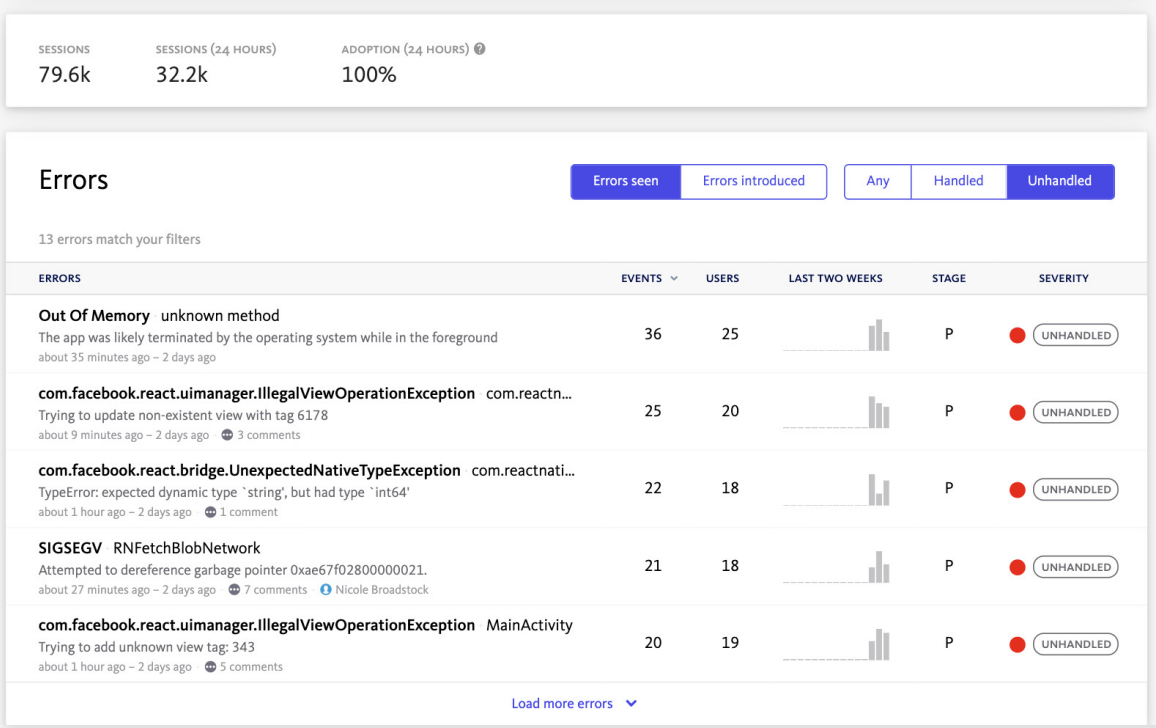
Bugsnag's Releases Dashboard also makes it easy to see when a release is ready for production and whether teams are trending toward stability targets. These objective metrics can help them make data-driven decisions while keeping the entire team on the same page regarding technical debt and quality assurance requirements.

Step #2: Prioritize

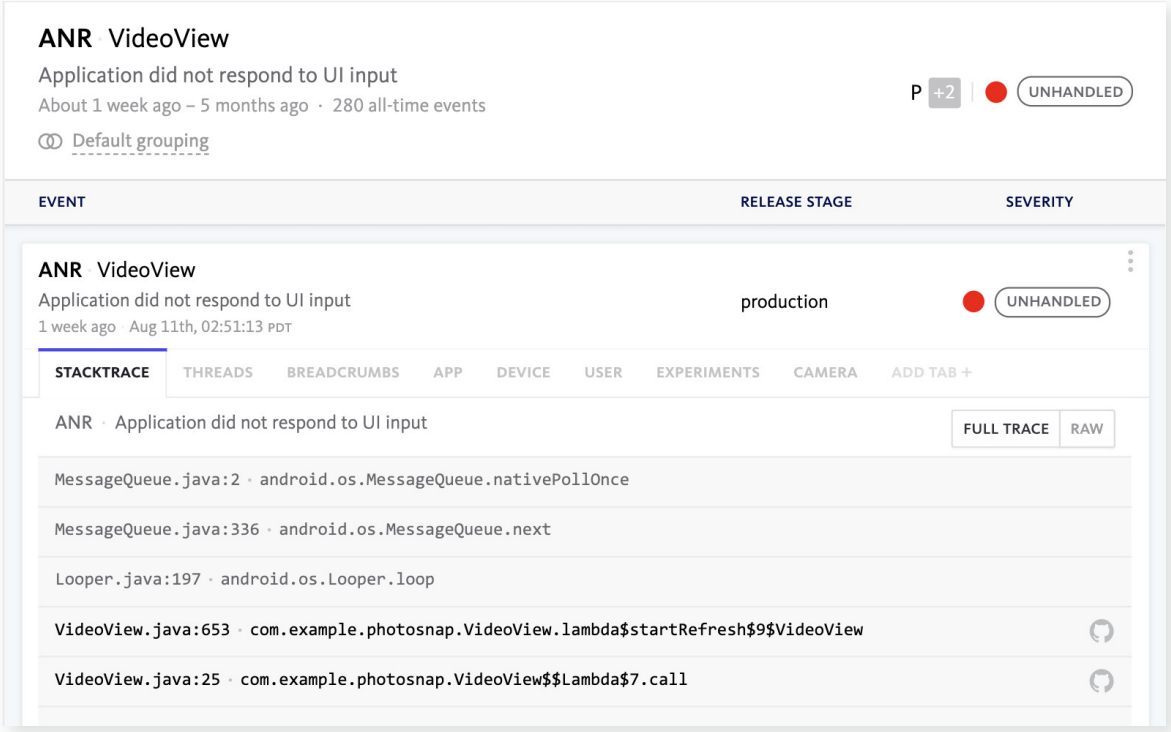
Bugsnag makes it easy to prioritize bugs. When the Stability Score drops below an acceptable level, the platform automatically shows you the top five errors occurring in that release, so you know where to start investigating problems. A simple search bar also provides immediate access to specific errors that may need a quick fix.

Bugsnag's Search and Segment capabilities also promote code ownership by enabling teams to segment errors by areas they're responsible for maintaining. For example, you can bookmark a segment of errors on the latest release and from a specific context (for example, the shopping cart). You can also quickly surface errors that impact a specific customer segment (for example, VIP customers).

Unlike many other error monitoring platforms, Bugsnag enables team members to customize alerts for specific segments, such as those occurring in their code base. That way, product managers don't need to worry about notification fatigue, and developers can access the information they need through their preferred communication channels.



Easily create alerts for specific features or segments to avoid notification fatigue. Source: Bugsnag



Bugsnag provides unparalleled diagnostic data to help fix bugs fast. Source: Bugsnag

Step #3: Fix

Bugsnag's intuitive and comprehensive diagnostic tools make tracking and fixing bugs easier. With full-stack observability and rich diagnostics, developers quickly can access fully readable stack traces, user action breadcrumbs, environmental data, and other information they need to reproduce bugs and implement fixes quickly.

In addition, Bugsnag automatically groups errors by root cause, meaning developers don't have to worry about two people working on the same fix. The platform also integrates with the most popular programming languages and frameworks to provide rich detail and trace errors across the front-end and back-end technology stack.

Easy to Implement

Bugsnag is fast and straightforward to implement. With personal assistance from a specialized onboarding team, you can easily accomplish a same-day migration from a logging system or homegrown tool. The platform's 50-plus pre-built integrations mean developers can be confident everything will work out of the box without the need for building custom integrations.

At the same time, Bugsnag packages everything into a single, lightweight SDK with minimal dependencies to avoid increasing your app's size or loading time. For example, JavaScript developers easily can import the pre-built NPM package into their application and call Bugsnag's notify function when handling errors to access unparalleled insights.

In addition to easy code-level integrations, the platform supports over 40 application-level integration with development tools, ranging from PagerDuty and Jira to Slack and Twilio. That way, development teams quickly can assimilate Bugsnag into their existing workflows with automatic issue creation, alerting, and other functionalities.

Finally, organizations with privacy, security, and compliance requirements also can leverage an on-premise version of Bugsnag. The self-hosted version helps meet PCI or other compliance standards by hosting Bugsnag on a single machine or clustered servers. And you'll have the support of Bugsnag's engineering team to ensure proper installation and maintenance.

Summary

Bugsnag is a comprehensive Application Stability Management (ASM) platform that takes a unique three-step approach to ensure application stability and help software teams balance feature development with bug troubleshooting. Bugsnag gives you cutting-edge features, and the experienced customer success team from SmartBear supports the platform to help you with everything from onboarding and installation to unique on-premise requirements.

The Bottom Line

Error monitoring solutions help improve productivity, create robust user experiences, and provide clarity for product managers. From front-end UI issues to back-end dependency errors, several different potential problems might arise. Software teams can mitigate these issues by combining end-to-end test automation, error monitoring, and application stability management solutions like Bugsnag.

While many early-stage organizations opt for open source and homegrown tools, Bugsnag offers a highly effective and scalable solution that keeps your development team working full speed on tasks that matter most. The platform's unique combination of actionable high-level metrics, detailed low-level diagnostics, and customizable alerts makes it the best solution for businesses to improve quality, reduce cost, and keep developers happier.

Start a free trial, or request a demo today!

FAQs

How much does Bugsnag cost?

Bugsnag offers a free, open source plan for one user, as well as paid plans for small to medium-sized teams starting at $35 a month for five seats and 150,000 monthly events. If you're a larger organization requiring more events, seats, and features, contact us to discuss the right plan.

How does the free trial work?

The free trial provides unrestricted access to all standard features for 14 days, and you don't need a credit card to sign up. At the end of the trial, you can select a plan to continue using Bugsnag without having to set up the account again.

What platforms does Bugsnag support?

Bugsnag supports over 50 platforms and technologies. You can see the entire list here and sort by server, mobile, browser, or desktop categories. We also provide documentation and onboarding support to help get you started.

Real-time direction on application stability. Confidently make your next move without digging for information.

Start 14-Day Free Trial

Request a Demo