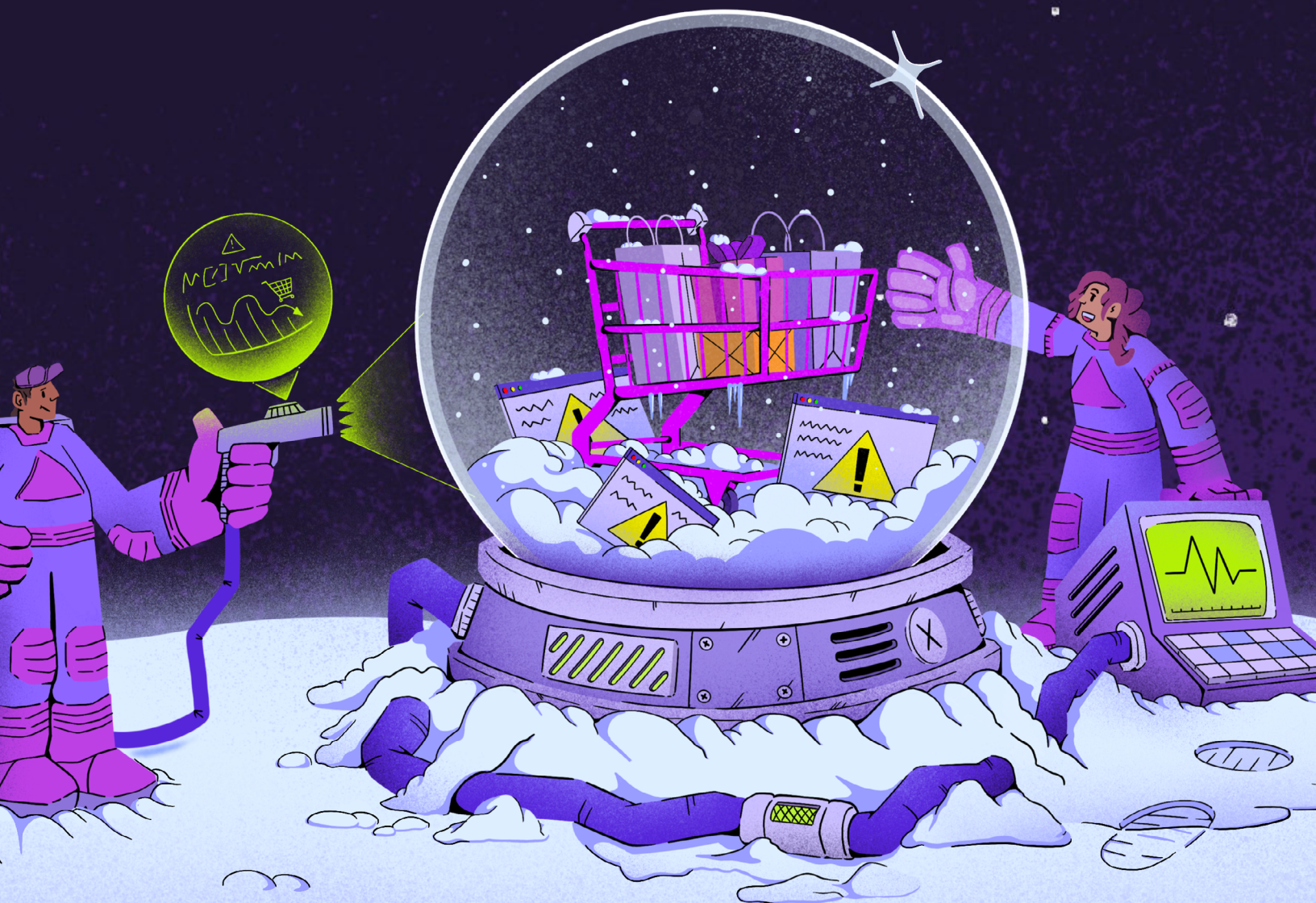




Holiday E-Commerce Checklist: A Developer's Survival Guide



Introduction

Nothing quite matches the terror of a checkout error at 11:59 PM on Black Friday.

There's never a good time for errors or performance degradations to show up, but during periods of peak traffic like the holidays, it's especially critical to get immediate answers about what's failing and how to fix it.

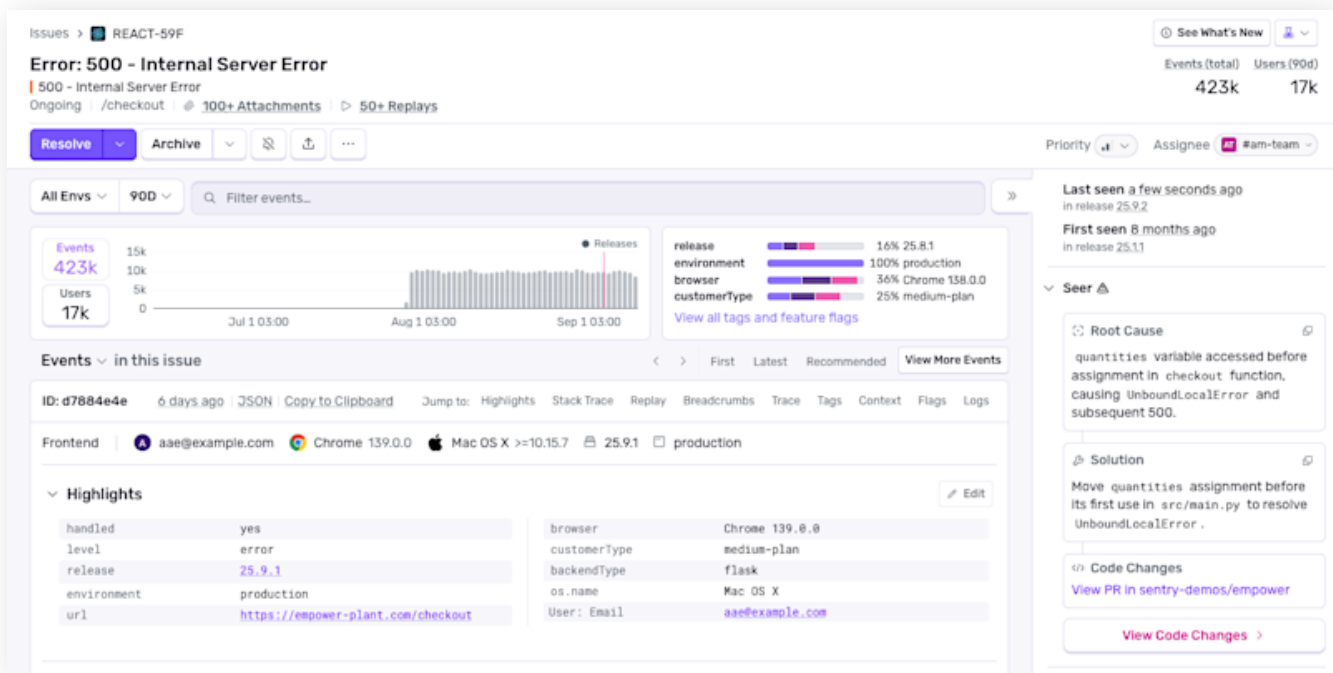
E-Commerce Holiday Readiness

Holiday readiness means setting up comprehensive error and performance monitoring before code freezes lock you out of making changes. The key is having these systems running and proven during normal traffic, so when Black Friday hits, you're not scrambling to figure out what's going on. Here's your holiday monitoring checklist, with the steps you need to take to survive (or even thrive) during peak shopping periods:

- ✓ **Implement error monitoring:** Triage and fix issues with detailed error context and debugging information.
- ✓ **Prioritize fixes with smart error alerting:** Configure alerts that reduce noise and surface revenue-threatening issues.
- ✓ **Collect user feedback and watch session replays:** Capture [session replay](#) data and real time user feedback when customers encounter problems.
- ✓ **Optimize site performance:** Monitor and identify performance bottlenecks using distributed tracing and logs before they impact conversion rates.
- ✓ **Eliminate bugs before they ship:** Keep errors from reaching production in the first place.

Implement error monitoring

Every error in your critical experiences like sign-in and checkout is potential revenue walking out the door. Holiday readiness starts with making sure you're capturing errors in prod so you don't have to wait for customer complaints to tell you what's wrong. Use [Sentry's error monitoring](#) to automatically track exceptions across your checkout and payment systems, complete with the context you need to fix issues quickly.



- **Prioritize what matters:** [Grouped issues](#) keep related failures together, so you can focus on high-impact errors like payment failures instead of sifting through noise.
- **Understand the full picture:** Get the connected context you need to understand what went wrong and why, with readable stack traces, [tags](#), and [breadcrumbs](#) of user actions so you can see what happened leading up to an error.
- **Use AI and automation to resolve faster:** Use [suspect commits](#) to tie issues directly to the most likely commit and author. Sentry's AI agent [Seer](#) uses rich context to identify actionable issues in real-time and generate fixes automatically.

Prioritize fixes with smart error alerting

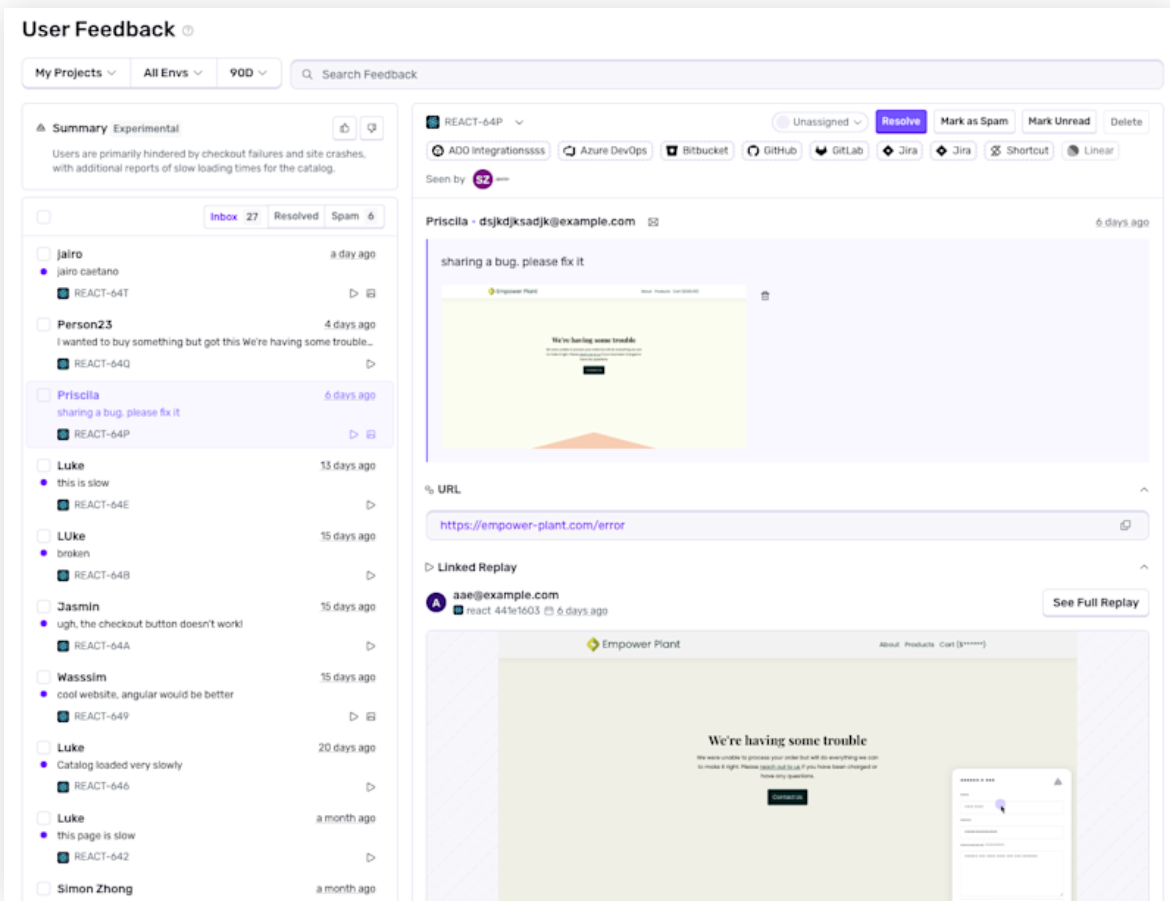
If you can't separate noise from critical issues, you risk missing the problems that actually cost you money. Sentry's [Alerts](#) are built to focus on user impact. With flexible rules and filters, you can cut through the noise and make sure the right people see the right problems fast.

- **Focus on impact:** Create alerts when issues match specific criteria (like when the triggering event is fatal), macro-level metrics cross specific thresholds (like when an issue was seen 100 times in 24 hours), or when an HTTP request doesn't return a successful response.
- **Adapt [thresholds](#) to traffic:** Pair fixed thresholds with dynamic thresholds to flag anomalies compared to historical baselines, so you catch problems even as traffic patterns shift.
- **Route alerts by severity:** Critical payment errors should page the on-call engineer immediately. Medium-severity issues might go to Slack. Environment filters prevent staging or dev errors from cluttering production alerts.

Done right, alerts help you see which fires are worth fighting first so you never miss the ones that are burning revenue.

Understand the user experience with session replays and user feedback

Most customers will just leave your site before telling you when something breaks. Sentry's [Session Replay](#) shows you a video-like playback of user interactions on your site, in relation to network requests, DOM events, and console messages. By combining Session Replay with the [User Feedback Widget](#), which lets you collect feedback directly from end users, you can understand critical details about errors and the UX of your site.

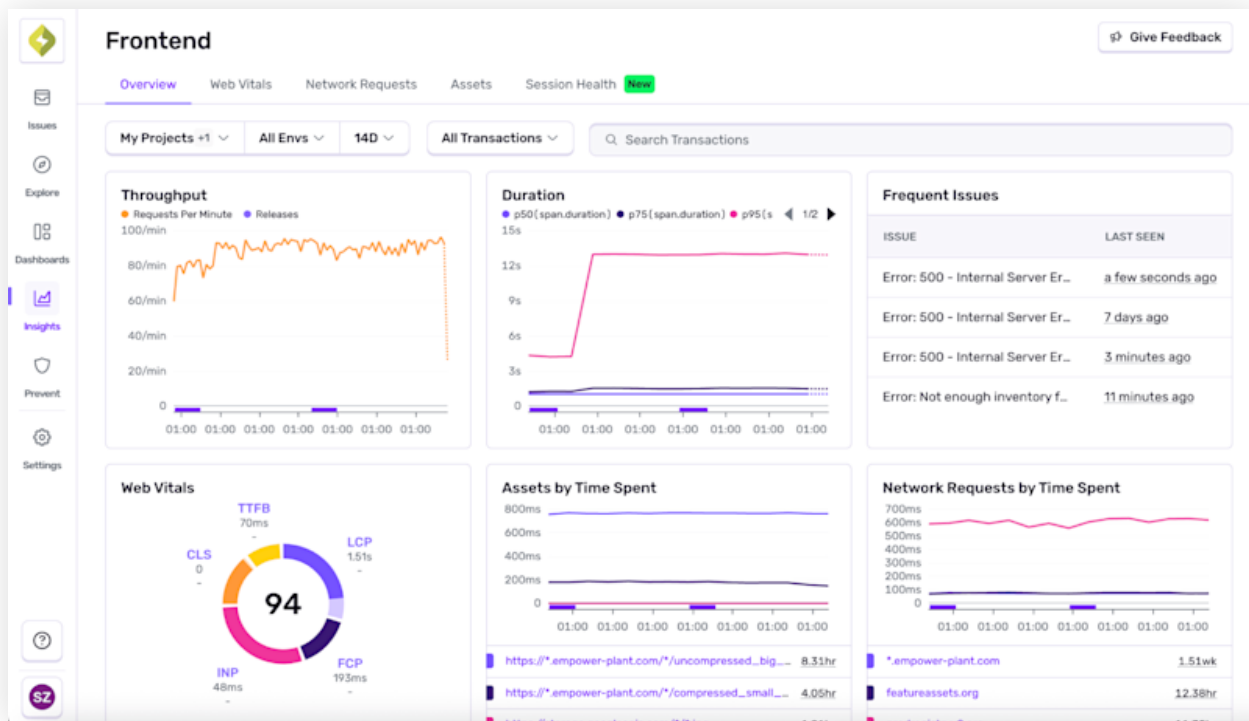


- **Use visual context to debug issues:** Replays show the exact steps that led to an issue, helping you reproduce bugs without guesswork or endless back-and-forth.
- **Connect feedback to real behavior:** User feedback submissions can be paired with replays to see what the customer experienced leading up to the submission.
- **Route feedback efficiently:** Critical problems like payment failures should go straight to engineers, while general design suggestions can wait for the next product review cycle.

Once you've fixed issues, keep feedback loops open. Replays and reports together give you a continuous view of user experience, so you can spot and resolve new frustrations before they cost you more customers.

Optimize site performance

A 100-millisecond delay can [reduce conversions by 7%](#). During holiday traffic spikes, slow pages don't just frustrate shoppers – they push them directly to competitors. [Real User Monitoring \(RUM\)](#) data shows how performance issues like poor [LCP](#) and [FCP](#) directly impact abandonment rates when customers are trying to check out their cart full of limited-time product deals.



Start by capturing real-world performance data. Use [Sentry's Real User Monitoring](#) to track how actual shoppers experience your site during peak conditions, and head to [Insights -> Frontend](#) in Sentry to surface any metrics that fall outside recommended thresholds.

- **Review [Time to First Byte \(TTFB\)](#):** Drill into transactions causing slow initial responses by drilling into high-TTFB spans. Surface specific requests like database queries or third-party calls that delay the start of page rendering, so you can prioritize optimizations that actually improve user-perceived performance.

- **Pinpoint bottlenecks using [distributed tracing](#):** Look at API performance, slow database queries, or problematic third-party scripts that are likely to experience degraded performance with higher traffic.
- **Use [trace-connected logs](#) for debugging and optimization:** Application and server logs are your ground truth during traffic spikes. Stream structured logs in real time (live-tailing) to spot anomalies, set up alerts on specific log patterns (e.g. repeated failed payment_status), and visualize trends over time with [dashboards](#).
- **Optimize asset loading:** Cache static assets (images, JS, CSS) for faster repeat visits, and utilize lazy loading to reduce initial load times.

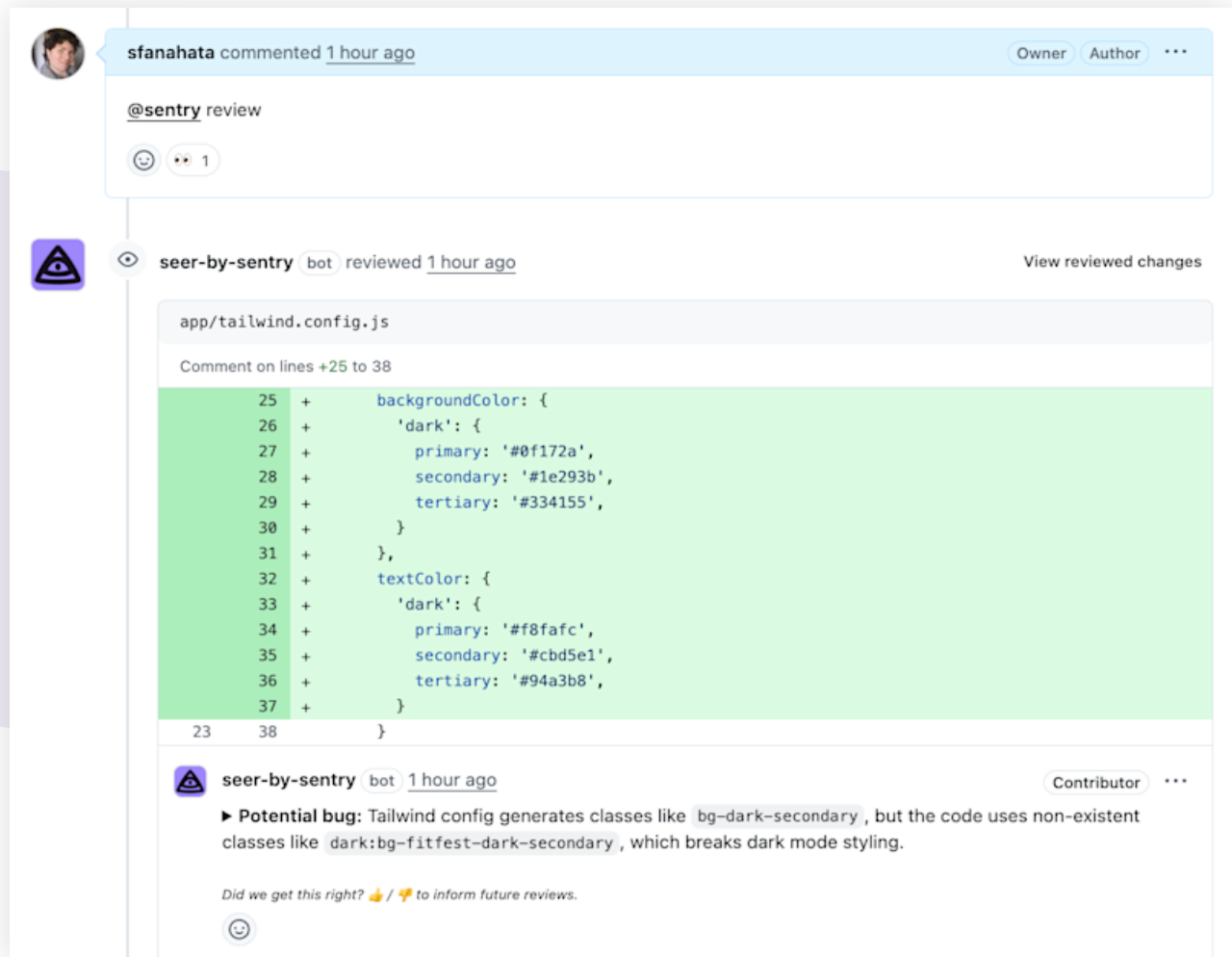
Once fixes are in place, validate by setting up [metric alerts](#) to compare before and after state.

Eliminate bugs before they ship

The best time to fix bugs is before they make it to prod. Manual code reviews are great, but when holiday deadlines are breathing down your neck, thorough reviews can become “LGTM” approvals.

Your team’s good, but nobody’s catching everything. AI can hunt down the edge cases humans miss and warn you before they blow up in prod.

When you create a pull request and set it to Ready for Review, [Sentry AI code review](#) will check for errors in your code. If no error is found, you will see a 🎉 emoji as a reaction to your PR description. Otherwise, it will add comments to your PR.



- **Predict errors:** Sentry's AI code review goes beyond standard PR review, pulling error context from Sentry and alerting you to high-impact bugs in your code.
- **Automate code review:** AI analysis identifies potential bugs, logical inconsistencies, and risky implementation patterns in critical code paths like checkout flows and payment processing.
- **Intelligent test generation:** Automatically creates comprehensive unit tests tailored to your specific code changes, including edge cases that developers commonly overlook.

AI-assisted review and testing is here, and it actually plays nice with your workflow. Error prediction, test generation, PR review. Just comment `@sentry review` or `@sentry generate-test` in your GitHub PR and watch it work. Think of it as preventative debugging, so your next holiday deploy doesn't turn into a production horror story.

Holiday without the debilitating anxiety

Don't wing it this holiday season. The tools and practices in this guide give you the visibility you need when everything's on fire and customers are trying to spend money on your site. Set this stuff up during normal traffic when you have time to test and fix issues. Trying to implement monitoring during Black Friday is like trying to install smoke detectors while your house is burning down.

When problems inevitably happen during peak shopping, you'll know exactly what's broken and how to fix it before your revenue disappears. And if it's too late for this season.....there's always next year.

