



Sentry Project Structure

The leader in application performance monitoring and error tracking. Sentry's codebase is a masterclass in managing a massive, long-lived monolith that serves millions of users, featuring a robust Django backend and a sophisticated React application.

Updated 2025-12-30

#sentry #monitoring #django #react #python #typescript #open-source

PNG

PDF

Copy

Prompt

Project Directory

- src/sentry/ The core Python...
- api/ Comprehensive R...
- models/ Core data models
- migrations/
- templates/ Server-side HTM...
- static/app/ The main React ...
- components/
- views/
- stores/ Global state ma...
- tests/ Massive automat...
- bin/ CLI and mainten...
- config/ Server configur...
- package.json Frontend depend...

Repository Info

Repository - [getsentry/sentry](#)

Stars - 38k+

License - BSL-1.1

Last Analyzed - December 2025

Tech Stack

Backend - Python (Django)

Frontend - React

Language - TypeScript / Python

Database - PostgreSQL, ClickHouse, Snuba

Queuing - Kafka & Redis

Architecture Notes

Sentry's architecture is a testament to the longevity of the Django/React pattern. While it appears as a monolith, it utilizes a highly distributed data plane called 'Snuba' (built on ClickHouse) to handle its massive analytical workload. The primary application logic remains in Python, providing a stable and well-understood foundation, while the frontend is a state-of-the-art React application that handles complex data visualization. Its codebase is notable for its incredible testing culture and rigorous standards for code quality.

Key Directories

src/sentry/ - Contains over a decade of business logic for error processing and management

static/app/ - A high-scale React application that serves as the primary interface for developers

tests/ - A critical part of the repo, ensuring stability for a mission-critical platform

config/ - Manages the complex configuration required to run Sentry in various environments

Why This Structure?

Sentry is perhaps the best example of how to scale a 'traditional' web stack to handle billions of events. It shows that you don't always need a microservices architecture to achieve massive scale. It's a gold standard for professional software engineering, particularly in the Python and React ecosystems.