

## Урок 3. Sentry — инструмент мониторинга ошибок

Sentry — это система оперативного мониторинга ошибок. Пользовательский интерфейс представляет собой dashboard со списком ошибок и возможностью выполнять над ними различные действия.

Зайдем на сайт Sentry, в раздел Python: <https://docs.sentry.io/platforms/python/>. Как и написано в документации, для начала необходимо создать учётную запись и sentry-проект.

```
from flask import Flask, request
import sentry_sdk
from sentry_sdk.integrations.flask import FlaskIntegration
import logging

sentry_sdk.init(
    dsn="https://yourtoken@o1022719.ingest.sentry.io/5999691",
    integrations=[FlaskIntegration()],

    # Set traces_sample_rate to 1.0 to capture 100%
    # of transactions for performance monitoring.
    # We recommend adjusting this value in production.
    traces_sample_rate=1.0
)

app = Flask(__name__)

@app.route('/debug-sentry')
def trigger_error():
    division_by_zero = 1 / 0

@app.route('/test_type')
def test_type():
    user_id = request.args.get('user_id')
    user_id = float(user_id)

@app.route('/test')
def ll():
    raise IndexError
```

```
@app.route('/test_logging')
def test_logging():
    logging.error("error to log")

if __name__ == '__main__':
    app.run()
```

**ZeroDivisionError** trigger\_error New Issue ISSUE

Unhandled division by zero SBERBANK-7D-

Resolve Ignore Mark Reviewed Share Open in Discover

Details Activity User Feedback Attachments Tags Events Merged Issues Similar Issues

Event [ab67521520f14a59bdb4d4f993578484](#) | JSON (10.0 KiB)  
 Oct 1, 2021 11:13:48 PM UTC | [View Full Trace](#)

Tags

Chrome Version: 91.0.4472 CPython Version: 3.7.4 Mac OS X Version: 10.15.7

browser Chrome 91.0.4472 browser.name Chrome client\_os Mac OS X 10.15.7 client\_os.name Mac OS X device Mac

device.family Mac environment production handled no level error mechanism flask release 8384051605f5

runtime CPython 3.7.4 runtime.name CPython server\_name CAB-WSM-0014224 transaction trigger\_error

url <http://127.0.0.1:5000/debug-sentry>

Одно из ключевых преимуществ относительно логов — это хранение состояния переменных в момент ошибки: мы можем сразу понять, какая ситуация вызвала исключение. Особенно это актуально, если человек ещё не сильно погружен в проект и не знает структуру кода. В случае лога мы бы увидели только traceback, пошли бы на локальную машину и там дебажили проект, пробуя подавать на вход разные данные.

Ещё один очевидный плюс — удобная фильтрация ошибок. Так, например, в случае повторения ошибки, Sentry добавит тикету ошибки в счётчик +1. Таким образом, мы не будем засорять интерфейс кучей одинаковых ошибок. Ошибки, которые фиксирует Sentry, можно отправлять на почту, в Telegram-бота или Slack. В случае повторных ошибок оповещение отправлено не будет, и, если вы считаете, что этот баг не такой важный, ошибка будет просто накручивать счётчик.

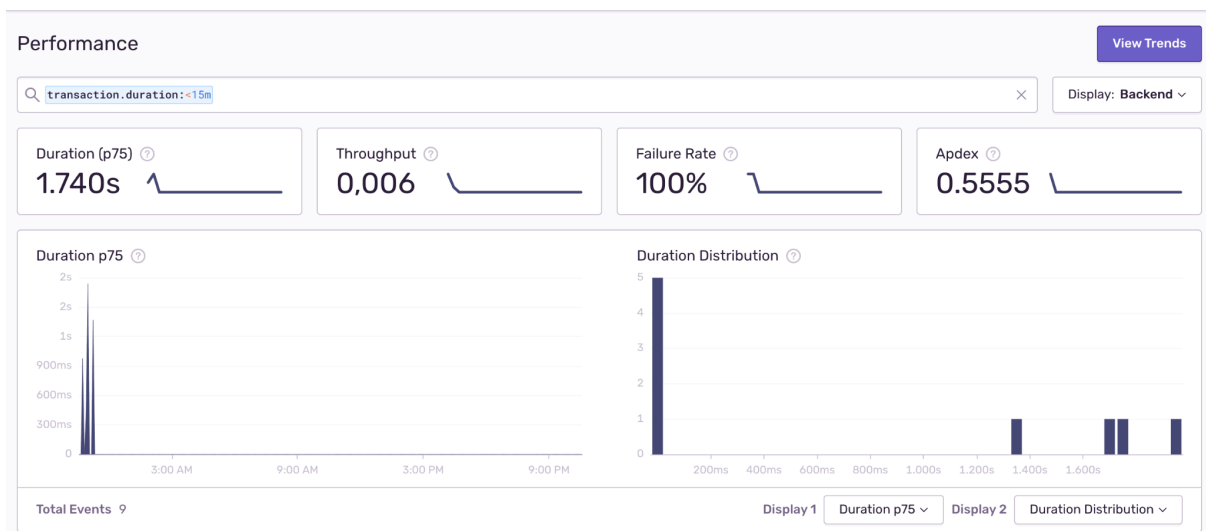
Здесь есть два решения: править все баги и не оставлять на потом или ставить autoreresolve через какой-то промежуток времени, чтобы ошибки закрывались и не скапливались.

Sentry сразу же интегрируется с модулем logging, то есть при вызове лога в коде он также будет отражен в интерфейсе Sentry.

Issues <span>▶</span>									
All Unresolved <span>5</span> For Review <span>5</span> Ignored Saved Searches <span>▼</span>									
<div> <input type="text" value="is:unresolved"/> <span>✕ ✎ + ⌵</span> <span>Sort by: Last Seen <span>▼</span></span> </div>									
<div> <input type="checkbox"/> <span>Resolve <span>▼</span></span> <input type="checkbox"/> <span>Ignore <span>▼</span></span> <input type="checkbox"/> <span>Mark Reviewed</span> <input type="button" value="Merge"/> <input type="button" value="..."/> </div>									
				GRAPH: 24h 14d	EVENTS	USERS	ASSIGNEE		
<input type="checkbox"/>	<b>TypeError</b>	test_logging	The view function for 'test_logging' did not return a valid response. The function either returned None or ende...		1	0	<span>▼</span>		
	New Issue	SBERBANK-7D-5	Unhandled	24hr ago   24hr old					
<input type="checkbox"/>	<b>error to log</b>	test_logging			1	0	<span>▼</span>		
	New Issue	SBERBANK-7D-4	24hr ago   24hr old	root					
<input type="checkbox"/>	<b>ValueError</b>	test_type	could not convert string to float: 'kkk'		2	0	<span>▼</span>		
	New Issue	SBERBANK-7D-3	Unhandled	24hr ago   24hr old					
<input type="checkbox"/>	<b>TypeError</b>	test_type	float() argument must be a string or a number, not 'NoneType'		1	0	<span>▼</span>		
	New Issue	SBERBANK-7D-2	Unhandled	25hr ago   25hr old					
<input type="checkbox"/>	<b>ZeroDivisionError</b>	trigger_error	division by zero		1	0	<span>▼</span>		
	New Issue	SBERBANK-7D-1	Unhandled	4d ago   4d old					

Что касается мониторинга производительности, в интерфейсе Sentry, во вкладке Performance, мы видим графики производительности наших роутов. С помощью мониторинга производительности Sentry отслеживает производительность приложений, измеряет такие показатели, как пропускная способность и задержка, а также отображает влияние ошибок в нескольких службах.

В интерфейс выводится четыре ключевых показателя: Duration (показывает длительность, быстрее которой выполняется определённая часть запросов), Throughput (пропускная способность, количество зарегистрированных транзакций в минуту), Failure Rate (интенсивность отказов, процент зарегистрированных транзакций, которые были зафиксированы в Sentry и имели неуспешный статус) и метрика Apdex (стандартная метрика, используемая для отслеживания и измерения удовлетворённости пользователей на основе времени отклика вашего приложения). Оценка Apdex показывает соотношение удовлетворительных, допустимых и неудовлетворённых запросов в конкретной транзакции или конечной точке. Чем выше метрика, тем удовлетвореннее наши пользователи.



### Плюсы Sentry:

- легко интегрировать в проект,
- поддержка большого количества инструментов,
- realtime-фиксация ошибок приложения,
- мониторинг производительности запросов,
- система оповещений по email, Telegram, Slack,
- отображение traceback и состояния переменных.