Crash report with Sentry using Expo

Working with crash reports in React Native can identify and fix issues that are causing app to crash in development and production.

Enable crash reporting in app by using a library such as Crashlytics or Sentry.
These libraries will automatically collect crash reports from the app and provide detailed information about the crash, such as the stack trace, device information, and user data.

What it will do ?

- Automatic Native Crash Error
- Offline storage of events
- Events enriched with device data
- Autolinking
- Breadcrumbs created for outgoing http request with XHR and Fetch; UI and system -events; and console logs
- Release health tracks crash free users and sessions
- Performance Monitoring creates transactions automatically for React Navigation v4 and above; creates spans automatically for XHR and Fetch
- Under the hood the SDK relies on our JavaScript SDK, which makes all functions available for JavaScript also available in this SDK
- RAM bundle support
- Hermes support

- Attachments enrich event by storing additional files, such as config or log files.
- User Feedback provides the ability to collect user information when an event occurs.

Set up a mechanism to receive crash reports in y development environment, such as by configuring the library to send crash reports to email or to a specific channel in team's communication tool.

Review the crash reports regularly to identify patterns and common issues. Look for issues that affect a large number of users or that are causing the app to crash frequently.

Reproduce the crash locally by following the steps provided in

the crash report, or by using the provided stack trace to identify the

specific line of code that is causing the issue.

Fix the issue by modifying the code and testing the app again.

Track progress of the issue, notify the team and close the issue once it has

been resolved.

It's worth mentioning that React Native also provides a way to catch and handle exceptions using try-catch blocks, this can help you to catch and handle the exception before it causes the crash.

Also, you can use debugging tools like react-devtools and redux-devtools to

debug and trace the code and state of the application, this way we can

identify the exact point where the crash is happening and also understand

what are the variables and state of the application when the crash happens.

https://docs.sentry.io/platforms/react-native/

Why sentry-expo?

Sentry treats React Native integrated with expo now.

- It's very easy to set up and use
- It scales to meet the demands of even the largest projects.
- We trust it for our projects at Expo.
- It is free for up to 5,000 events per month.
- It streamlines error-reporting code across iOS, Android, and web

Installing and configuring Sentry :

Step 1: Sign up for a Sentry account and create a project

https://sentry.io/signup/

dsrmurthysoftware@gmail.com/Welcome@123

Sign up for Sentry (it's free), and create a project in Dashboard by clicking start.

Select the platform to Monitor:

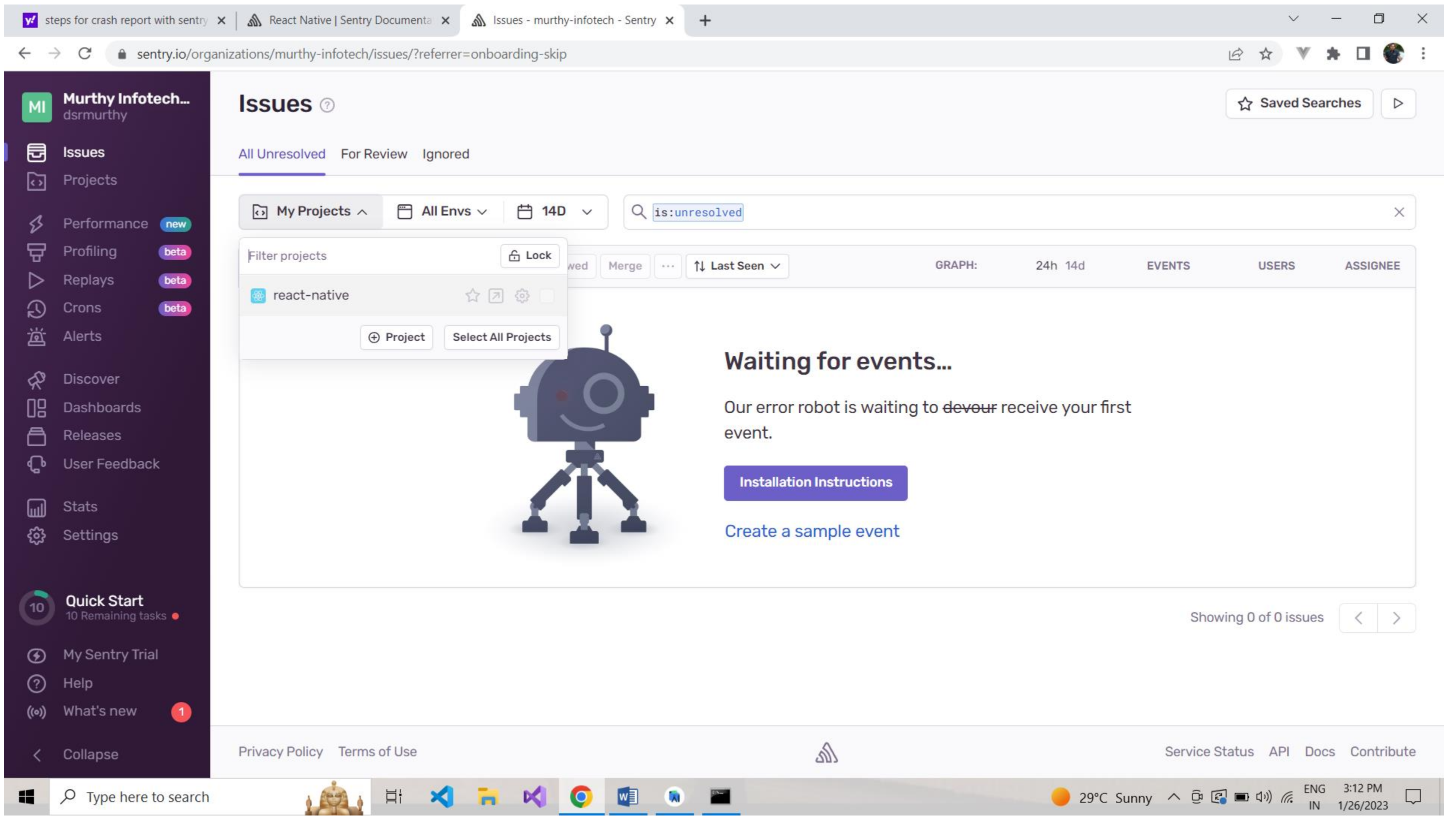
In Mobile tab , select React Native icon and click create project.

Prepare react-native sdk in your project.

```
C:\jest-app>npm install @sentry/react-native
```

2.Go to the Sentry API section, and create an auth token. The token requires the scopes: org:read, project:releases, and project:write. Save this, too.

Once you have each of these: organization name, project name, DSN, and auth token, you're all set!


steps for crash report with sentry | React Native | Sentry Documenta | Issues - murthy-infotech - Sentry

sentry.io/organizations/murthy-infotech/issues/?referrer=onboarding-skip

Murthy Infotech... dsrmurthy

Issues · Projects · Performance (new) · Profiling (beta) · Replays (beta) · Crons (beta) · Alerts · Discover · Dashboards · Releases · User Feedback · Stats · Settings

Quick Start — 10 Remaining tasks

My Sentry Trial · Help · What's new · Collapse

Issues

All Unresolved · For Review · Ignored

My Projects · All Envs · 14D · is:unresolved

Filter projects · Lock · react-native · Project · Select All Projects

Merge · Last Seen · GRAPH: 24h 14d · EVENTS · USERS · ASSIGNEE

Waiting for events...

Our error robot is waiting to ~~devour~~ receive your first event.

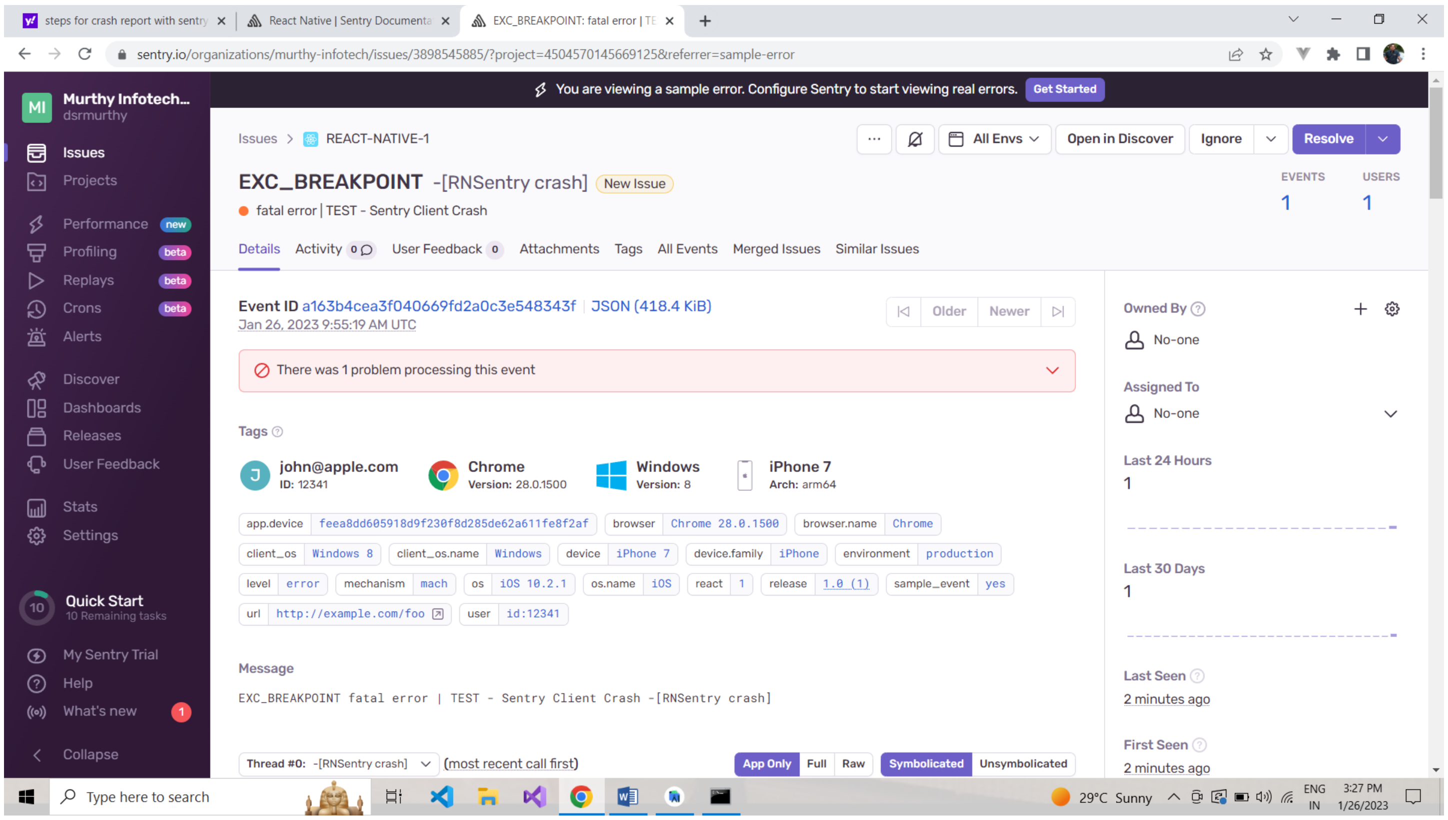
Installation Instructions

Create a sample event

Showing 0 of 0 issues

Privacy Policy · Terms of Use · Service Status · API · Docs · Contribute

Sample event :


steps for crash report with sentry | React Native | Sentry Documenta | EXC_BREAKPOINT: fatal error | TE

sentry.io/organizations/murthy-infotech/issues/3898545885/?project=4504570145669125&referrer=sample-error

You are viewing a sample error. Configure Sentry to start viewing real errors. Get Started

Issues > REACT-NATIVE-1

All Envs · Open in Discover · Ignore · Resolve

EXC_BREAKPOINT -[RNSentry crash] New Issue

fatal error | TEST - Sentry Client Crash

EVENTS 1 · USERS 1

Details · Activity 0 · User Feedback 0 · Attachments · Tags · All Events · Merged Issues · Similar Issues

Event ID a163b4cea3f040669fd2a0c3e548343f | JSON (418.4 KiB)

Jan 26, 2023 9:55:19 AM UTC

Older · Newer

There was 1 problem processing this event

Tags

john@apple.com ID: 12341 · Chrome Version: 28.0.1500 · Windows Version: 8 · iPhone 7 Arch: arm64

app.device	feea8dd605918d9f230f8d285de62a611fe8f2af
browser	Chrome 28.0.1500
browser.name	Chrome
client_os	Windows 8
client_os.name	Windows
device	iPhone 7
device.family	iPhone
environment	production
level	error
mechanism	mach
os	iOS 10.2.1
os.name	iOS
react	1
release	1.0 (1)
sample_event	yes
url	http://example.com/foo
user	id:12341

Message

```
EXC_BREAKPOINT fatal error | TEST - Sentry Client Crash -[RNSentry crash]
```

Thread #0: -[RNSentry crash] (most recent call first) · App Only · Full · Raw · Symbolicated · Unsymbolicated

Owned By: No-one

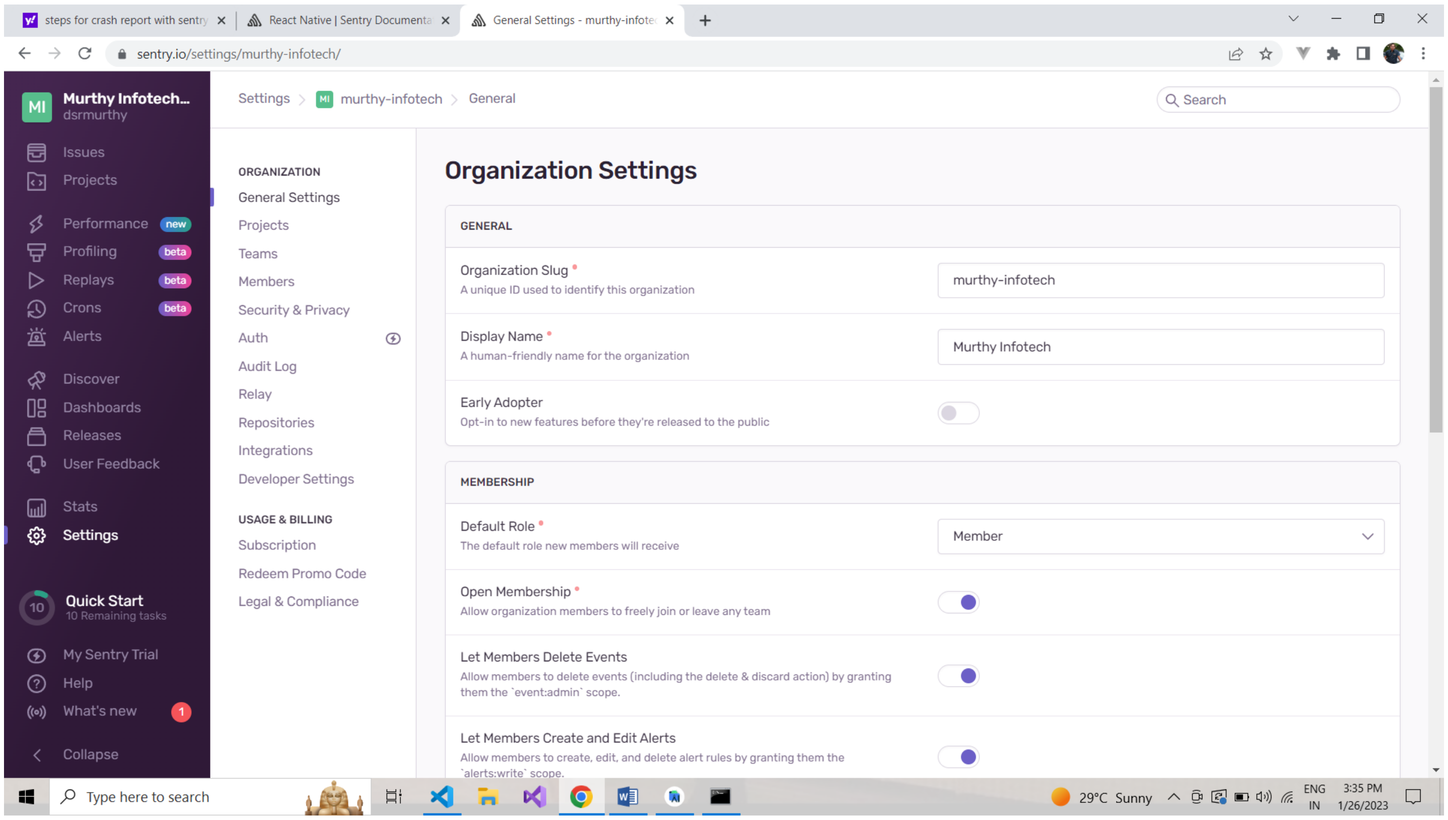
Assigned To: No-one

Last 24 Hours: 1

Last 30 Days: 1

Last Seen: 2 minutes ago

First Seen: 2 minutes ago



Organization Settings

GENERAL

Organization Slug *
A unique ID used to identify this organization
murthy-infotech

Display Name *
A human-friendly name for the organization
Murthy Infotech

Early Adopter
Opt-in to new features before they're released to the public

MEMBERSHIP

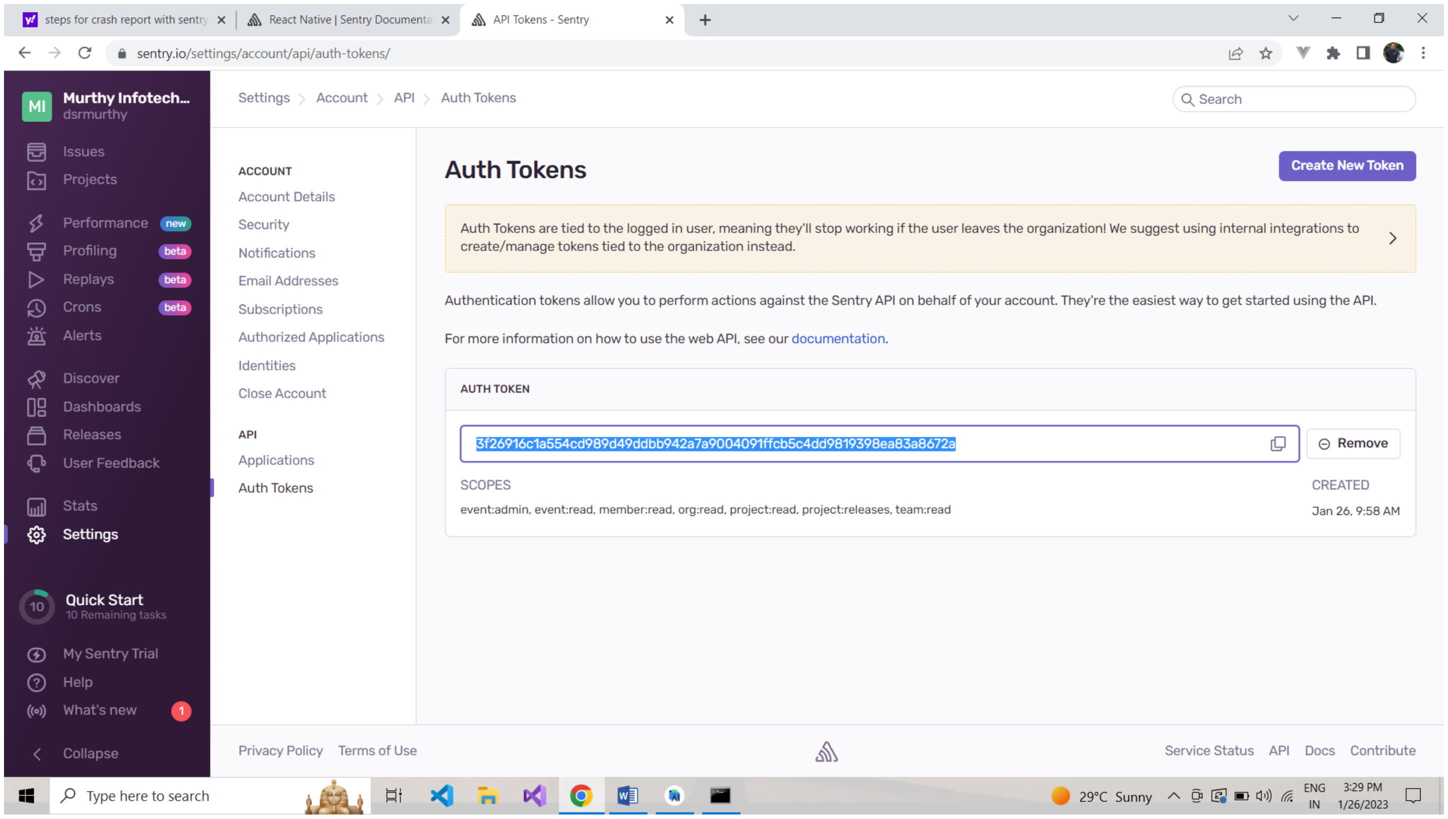
Default Role *
The default role new members will receive
Member

Open Membership *
Allow organization members to freely join or leave any team

Let Members Delete Events
Allow members to delete events (including the delete & discard action) by granting them the `event:admin` scope.

Let Members Create and Edit Alerts
Allow members to create, edit, and delete alert rules by granting them the `alerts:write` scope.

Crated auth. Token: From auth



Auth Tokens

Auth Tokens are tied to the logged in user, meaning they'll stop working if the user leaves the organization! We suggest using internal integrations to create/manage tokens tied to the organization instead.

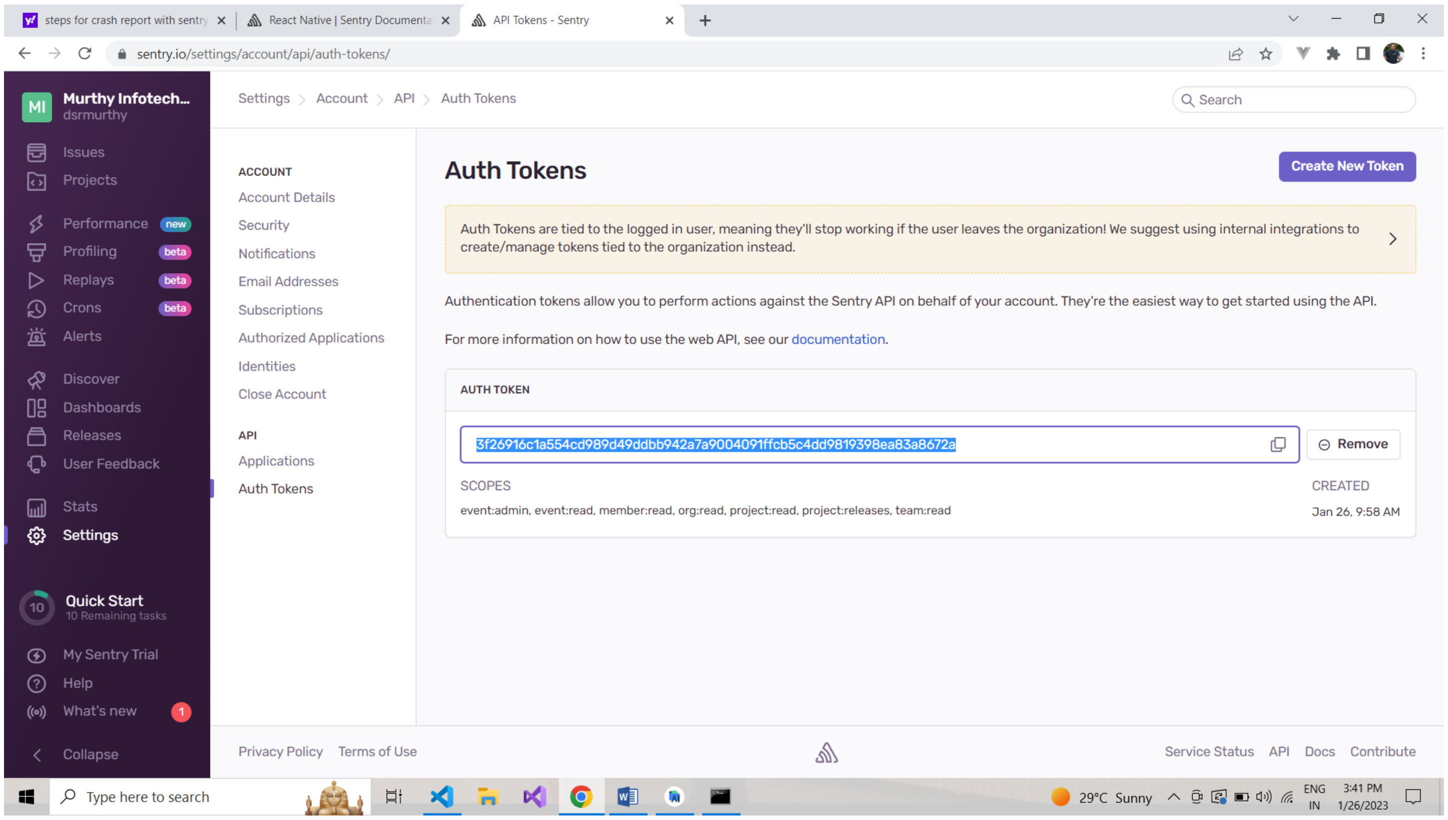
Authentication tokens allow you to perform actions against the Sentry API on behalf of your account. They're the easiest way to get started using the API.

For more information on how to use the web API, see our documentation.

AUTH TOKEN

3f26916c1a554cd989d49ddbb942a7a9004091ffcb5c4dd9819398ea83a8672a

SCOPES	CREATED
event:admin, event:read, member:read, org:read, project:read, project:releases, team:read	Jan 26, 9:58 AM



Auth Tokens

Auth Tokens are tied to the logged in user, meaning they'll stop working if the user leaves the organization! We suggest using internal integrations to create/manage tokens tied to the organization instead.

Authentication tokens allow you to perform actions against the Sentry API on behalf of your account. They're the easiest way to get started using the API.

For more information on how to use the web API, see our documentation.

AUTH TOKEN

3f26916c1a554cd989d49ddbb942a7a9004091ffcb5c4dd9819398ea83a8672a

SCOPES: event:admin, event:read, member:read, org:read, project:read, project:releases, team:read

CREATED: Jan 26, 9:58 AM

Step 2: Installation

In your project directory, run:

```
$ npx expo install sentry-expo
```

sentry-expo also requires some additional Expo module packages. To install them, run:

```
$ npx expo install expo-application expo-constants expo-device expo-updates @sentry/react-native
```

Step 3: Code

Initialization

Add the following to your app's main file such as App.js:

```
import * as Sentry from 'sentry-expo';

Sentry.init({
  dsn: 'YOUR DSN HERE',
  enableInExpoDevelopment: true,
  debug: true, // If `true`, Sentry will try to print out useful debugging information if something goes wrong with sending the event. Set it to `false` in production
});
```

Usage

Depending on which platform you are on (mobile or web), use the following methods to access any @sentry/* methods for instrumentation, performance, capturing exceptions and so on:

For React Native, access any @sentry/react-native exports with Sentry.Native.*

For web, access any @sentry/browser exports with Sentry.Browser.*

```
// Access any @sentry/react-native exports via:
// Sentry.Native.*
// Access any @sentry/browser exports via:
// Sentry.Browser.*

// The following example uses `captureException()` from Sentry.Native.* to
capture errors:

try {
  // your code
} catch (error) {
  Sentry.Native.captureException(error);
}
```

Step 4: App Configuration

Configuring sentry-expo is done through the config plugin in your app.json or app.config.js.

Configure a postPublish hook

Add expo.hooks property to your project's app.json or app.config.js file:

```
{
  "expo": {
    ... your existing configuration
    "hooks": {
      "postPublish": [
        {
          "file": "sentry-expo/upload-sourcemaps",
          "config": {
            "organization": "your sentry organization slug here",
            "project": "your sentry project name here",
            "authToken": "your auth token here"
          }
        }
      ]
    }
  }
}
```
The authToken value can be generated from the Sentry API page.

Add the Config Plugin

Add expo.plugins to your project's app.json or app.config.js file:

```
{
  "expo": {
    "plugins": ["sentry-expo"]
    ... your existing configuration
  }
}
```

Source Maps

With the postPublish hook in place, now all you need to do is run expo publish and the source maps will be uploaded automatically. We automatically assign a unique release version for Sentry each time you hit publish, based on the version you specify in app.json and a release id on our backend -- this means that if you forget to update the version but hit publish, you will still get a unique Sentry release.

This hook can also be used as a postExport hook if you're self-hosting your updates.

Uploading source maps at build time

Note: Disregard the following if you're using the classic build system (expo build:[android|ios]).

With expo-updates, release builds of both iOS and Android apps will create and embed a new update from your JavaScript source at build-time. This new update will not be published automatically and will exist only in the binary with which it was bundled. Since it isn't published, the source maps aren't uploaded in the usual way like they are when you run expo publish (actually, we are relying on Sentry's native scripts to handle that). Because of this you have some extra things to be aware of:

Your release will automatically be set to Sentry's expected value- ${bundleIdentifier}@${version}+${buildNumber} (iOS) or ${androidPackage}@${version}+${versionCode} (Android).

Your dist will automatically be set to Sentry's expected value: ${buildNumber} (iOS) or ${versionCode} (Android).

The configuration for build time source maps comes from the android/sentry.properties and ios/sentry.properties files. For more information, see Sentry's documentation. Manual configuration is only required for bare projects, the sentry-expo config plugin handles it otherwise.

Configuration for expo publish and npx expo export for projects is done via app.json, whether using bare workflow or not.

Skipping or misconfiguring either of these can lead to invalid source maps, and you won't see human-readable stacktraces in your errors.

Uploading source maps for updates

This requires SDK 47, with sentry-expo@6.0.0 or above and expo-updates@0.15.5 or above.

If you're using EAS Update, or if you're self-hosting your updates (this means you run npx expo export manually), you need to take the following steps to upload the source maps for your update to Sentry:

Run eas update. This will generate a dist folder in your project root, which contains your JavaScript bundles and source maps. This command will also output the 'Android update ID' and 'iOS update ID' that we'll need in the next step.

Copy or rename the bundle names in the dist/bundles folder to match index.android.bundle (Android) or main.jsbundle (iOS).

Next, you can use the Sentry CLI to upload your bundles and source maps:

release name should be set to ${bundleIdentifier}@${version}+${buildNumber} (iOS) or ${androidPackage}@${version}+${versionCode} (Android), so for example com.domain.myapp@1.0.0+1.

dist should be set to the Update ID that eas update generated.

AndroidiOS

```
node_modules/@sentry/cli/bin/sentry-cli releases \
   files <release name> \
   upload-sourcemaps \
   --dist <Android Update ID> \
   --rewrite \
   dist/bundles/index.android.bundle dist/bundles/android-<hash>.map
```

For more information, see Sentry's instructions for uploading the bundle and source maps.

The steps above define a new 'dist' on Sentry, under the same release as your full app build, and associate the source maps with this new dist. If you want to customize this behavior, you can pass in your own values for release and dist when you initialize Sentry in your code. For example:

```
import * as Sentry from 'sentry-expo';
import * as Updates from 'expo-updates';

Sentry.init({
  dsn: 'YOUR DSN',
  release: 'my release name',
  dist: 'my dist',
});
```

These values should match the values you pass to the sentry-cli when uploading your source maps.

Testing Sentry

When building tests for your application, you want to assert that the right flow-tracking or error is being sent to Sentry, but without really sending it to Sentry servers. This way you won't swamp Sentry with false reports during test running and other CI operations.

sentry-testkit enables Sentry to work natively in your application, and by overriding the default Sentry transport mechanism, the report is not really sent but rather logged locally into memory. In this way, the logged reports can be fetched later for your own usage, verification, or any other use you may have in your local developing/testing environment.

For more information on how to get started, see sentry-testkit documentation.

If you're using jest, make sure to add @sentry/.* and sentry-expo to your transformIgnorePatterns.

Error reporting semantics

In order to ensure that errors are reported reliably, Sentry defers reporting the data to their backend until the next time you load the app after a fatal error rather than trying to report it upon catching the exception. It saves the stacktrace and other metadata to AsyncStorage and sends it immediately when the app starts.

Disabled by default in dev

Unless enableInExpoDevelopment: true is set, all your dev/local errors will be ignored and only app releases will report errors to Sentry. You can call methods like Sentry.Native.captureException(new Error('Oops!')) but these methods will be no-op.