React Native: Developing Android and iOS Apps

Cheat Sheet: Error Handling and Debugging

Type	**Description**	**Code>Code Example**

try-catch-finally	Encompassing any code that could possibly throw an error or exception, known or unknown, can be embedded within a try-and-catch block. This will ensure the error is handled and the user experience is not hindered. In the final block, you can include code that needs to be executed irrespective of whether there is an error or not.	`    try {`
		`      const response = await`
		`fetch('https://jsonplaceholder.typicode.com/posts/1');`
		`      // Check if the response is successful`
		`      if (!response.ok) {`
		`        throw new Error('Failed to fetch data');`
		`      }`
		`      const result = await response.json();`
		`      setData(result); // Set the fetched data`
		`    } catch (err) {`
		`      // Handle error`
		`      setError(err.message);`
		`    } finally {`
		`      setLoading(false); // Hide loader`
		`    }`

eas-cli	The command installs the Expo Application Services Command Line Interface globally, enabling users to build, deploy, and manage their Expo applications directly from the command line.	`npm install -g eas-cli`
eas login	The command authenticates the user with their Expo account, allowing access to Expo services for building and managing applications.	`eas login`

EAS Configure	The command sets up the necessary configuration files and settings for building an Expo app with the EAS Build service, preparing the project for a build process.	`eas build:configure`
EAS Build Android	The command initiates the build process for an Expo application targeting the Android platform, using the configuration specified in the "preview" profile.	`eas build -p android --profile preview`