Logging Best Practices

In this document, we will briefly cover some logging best practices.

- **Always Use A Logging Library:** This ensures that the corresponding information will be logged from the corresponding context (inside function, at endpoint, at route, etc) without the need to change the system to do it. [1]
- **Logging Levels Must Be Correctly Used:** Each logging level should have its own independent log and all the entries of that level should be logged there. [1]
- **Use Proper Logging Category:** Log entries can be categorized by application component (Eg - db, auth, payment, API), by functionality (Eg - user actions, network event, security, etc), by environment (Eg - production, development, etc) [1]
- **Write Meaningful Log Entries:** All log entities should be written with the assumption that they can be used reliably as a standalone resource to troubleshoot the system. Context (Which function, endpoint, database or API) and remediation information (how to fix the problem) must be there in log messages. [1]
- **Log in Machine Parsable Format:** The log entries should preferably be written in JSON so that it is easy to parse and search. [1]
- **Log Appropriately:** The correct way to log is to log a lot when the system is in development and analyse it, and then in production, to log more on those parts of the system that are causing trouble. [1]
- **Know Your Audience:** Log message, content, category and level should be tailored to the audience for which it is intended. This could either be an end-user trying to troubleshoot a problem or a developer trying to do the same. [1]
- **Do Not Log Sensitive Information:** Sensitive information such as credit card numbers, passwords, names, authorization tokens, etc. shouldn’t be logged. [1]

Here, we will go briefly into some logging levels:

- **Trace:** Should only be used in development to track bugs and not be committed to the VCS. [1]
- **Debug:** This level should be used to log anything that happens in the program. This type of log entry should be reduced in production and should only be extensively used during troubleshooting. [1]
- **Info:** This level should be used to log all actions that're user-driven or regularly scheduled system operations. [1]
- **Notice:** This level should be used in production to log any notable event that isn't an error. [1]
- **Warn:** This level should be used to log any event that could turn into an error. Eg - A database call taking too long, or a cache at near-capacity. This will allow for better troubleshooting as we will be able to understand how the system was behaving before the error. [1]
- **Error:** This level should be used to log all error events. [1]
- **Fatal / Emerg:** Should be used to log any event that involves a program exiting due to an error. It signifies the end of unsuccessful program execution. [1]

REFERENCES

[1] https://www.dataset.com/blog/the-10-commandments-of-logging/