Code quality and performance audit

This document explains how we handled code quality and how we measure the performance of the ToDo List application.

Code quality

In order to maintain good quality code during the development of the application, we used the following tools:

- PHP Code sniffer to respect coding standards
- PHP Stan to identify errors
- Sonar Cloud for the general code review

PHP Code sniffer

This tool permits to detect violations of a coding standard using the command phpcs or automatically correct violations using the command phpcbf.

During the development of this application, we selected the PSR-12 standard (which implies PSR-1 and PSR-2 while extending the second) and we used the command phpcbf nameOfFolderHere --standard=PSR12 -p to apply this standard to our code located in the src folder.

PHP Stan

This tool scans each file looking for errors without having to actually run the application.

For instance, it highlighted a badly written class name.

Sonar Cloud

We used Sonar Cloud for the code review. It highlights security vulnerabilities, bugs and code smells.

Conclusion

Thanks to these three tools, we were able to respect PSR 12 coding standard as well as avoiding errors and security vulnerabilities.

Performance

We used BlackFire to study application performance. This tool can analyze memory consumption as well as the duration of each function called by our application, allowing us to see where we can make performance improvements.

Our application being rather simple, it runs fast without any improvements. Still, we saw through our BlackFire profiling that a lot of calls were executed to load classes. These calls could be avoided if we generate cached files for the composer autoloader by executing the following command: composer dump-autoload --optimize which convert PSR-0/4 autoloading to classmap to get a faster autoloader.