5.3 - Automate some aspects of the testing

The testing of this project is fully automated leveraging a GitHub Actions CI pipeline. A Maven build stage is triggered upon every time master is commited to (either directly or through a merge), or whenever a merge request with master as the target is pushed to. This executes all tests discussed in this project (along with some from the ILP coursework) automatically. Any test failures results in the pipeline to fail and prevents faulty merges into the master branch - enforcing testing.

If the level of testing of this project was to be extended, I would keep roughly this ratio of test types in the pipeline. Unit and integration tests are well suited to CI because they are fast and provide immediate feedback on code changes. Less system level tests should be included within the general CI pipeline, as they are generally slower - maybe have a separate action that only runs on attempted merges to the master branch.

More expensive testing, such as large scale performance testing, exhaustive random testing, or long-running stress tests should be intentionally excluded from the CI pipeline. These tests would be more appropriate to be ran less frequently, as they are much more resource intensive.

Overall, the automated testing strategy I've implemented ensures that core functional requirements are continuously validated when changes are made to critical branches, whilst maintaining a fast and reliable CI pipeline suitable for the iterative development cycle that this project has used so far.