Software Testing 2024/2025 Portfolio - Test Limitations

Omissions and deficiencies in testing

Testing suites often present with gaps and omissions and this may be for a variety of reasons; this includes things like limited time and resources, a complex system functionality and priority on functionalities other than those with which there are gaps in their testing. These factors did of course contribute to deficiencies of this PizzaDronz testing suite, evident in the under-testing of areas such as output validation or flight path optimisation. Additional testing in these areas would of course improve the system reliability and verify that all requirements and expectations discussed in the Informatics Large Practical are met.

It is also important to consider the deficiencies in the functionalities that are already looked at in the current testing suite.

Input validation

The input validation functionality is well tested in this testing suite. It focuses on handling incorrect inputs and throwing the correct errors and warning messages when needed, however the testing does not account for malicious inputs. If this was a commercial application, security testing would be vital in order to prevent users being able to provide inputs containing hostile data (like an injection attack). The specification does mention PizzaDronz running locally, however success with the application could push for expansion. This could certainly open up the system and its vulnerabilities to hackers. Incorporating rigorous security testing for input validation would ensure the system is more robust against this exploitation.

Drone pathing system

The testing suite does cover all recognised partitions in the coordinate system however there could still be improvements: repeated testing, especially in testing the `closeness` of the drone, could help identify inconsistencies or errors that may not appear in the current single instance tests. This would further help meet some of the non-functional requirements seen in the specification such as the safety of the drones operation (if something like a cumulation of rounding errors across thousands of movement instructions resulted in the drone crashing into a person or building, it is clear to see that this requirement has not been adequately met).

Order validation

The order validation testing in the testing suite does face limitations, mostly because of its reliance on real data from the REST server. If more functionalities or requirements were to be added, such as the need to be able to mark orders containing non-pizza items, the current test suite would be unable to adapt to this due to the reliance on the REST server. The introduction of a dedicated test REST server API in which the developer has access to customise specifically for testing purposes would resolve this omission and allow for easy growth of the project.

System functionality and runtime

The lack of specific flight path generation or optimization testing means that full system tests are not fully stress-tested and consistently perform well under the time limit. Stress testing at the system level is a critical missing element in the current suite, as it would assess how the system handles larger-than-usual datasets or increased complexity. However, implementing such tests is challenging because the system relies entirely on retrieving data from a REST server. To enable stress testing, a dedicated test REST server would need to be implemented, simulating high loads or extreme data scenarios for evaluation.

Testing suite documentation

The current implementations of tests have very little notation and documentation due to the time constraint on the project. Adding detailed documentation would make sure that other developers and testers could better understand the test plan. If more work was to be carried out on this project, the testing suite would also need to be updated; well-written documentation would certainly make this process easier.

Coverage

The test suite does achieve 100% coverage and the target, as discussed when setting the test evaluation criteria in ***TestResults.pdf***, should be maintained at the maximum as to ensure reliability in a system. Any changes to the application, whether that be updates to code or a change in functionality, must avoid deprecating parts of the system, as this could reduce coverage and would weaken robustness overall. To prevent this, it is crucial to implement constant automated testing and regular coverage calculations (as discussed in ***ciPipeline.pdf***).

Performance

The current system's performance does meet the specified requirements, but it may not be fully optimised in terms of time or space efficiency. Improvements would include refining the end-to-end system tests to handle longer-distance paths to and from restaurants. As mentioned in the gaps in the order validation functionality testing, introducing a dedicated test REST server would allow for more comprehensive performance testing under conditions like this. It would give developers the ability to identify bottle necks and areas of the system that are reducing performance.