***Software Testing Portfolio***

# **Introduction**

The project I have chosen for my portfolio is the ***Informatics Large Practical (ILP) project***, also known as the ***PizzaDronz app***. The concept of this app is to allow users to place pizza orders to several restaurants. These orders are then to be delivered close to Appleton Tower via a drone. The specifics of the PizzaDronz app are covered in more detail in the coursework specification document [1].

# **1. Analyse requirements to determine appropriate testing strategies [20%]**

## **1.1 Range of requirements, functional requirements, measurable quality attributes, qualitative requirements, etc.**

The requirements presented in the project requirement specification [2] are divided into three main categories: ***safety***, ***correctness,*** and ***liveness***, and all serve to address at least one of the ***9 measurable attributes*** specified in the documents. The small selection that best showcases the range of the requirements and will be discussed in this section of the portfolio is: requirement 22 (**efficiency requirement**), requirement 13 (**safety requirement**), requirement 18 (**correctness requirement**), requirement 14 (**liveness requirement**), and requirement 20 (**liveness requirement**). For the full description of these requirements please see the project requirement specification [2].

## **1.2 Level of requirements, system, integration, unit**

The level of requirements document [3] covers a more detailed outline of the level of each requirement outlined above. A few ideas in this document include requirement 14 being tested at ***unit level***, requirement 18 being tested at ***integration level***, and requirement 22 being tested at ***system level***.

## **1.3 Identifying test approach for chosen attributes**

A wider selection of testing approaches can be found in the testing approaches document [4]. However, the testing approach I have opted for is that of ***a combination of the proactive and reactive approaches***. This approach is best illustrated by the requirement in point 3 (**liveness requirement 2**). This requirement clearly requires some implementation details to be known, but the testing can be done, or simply refined, at a later stage of the development process.

## **1.4 Assess the appropriateness of your chosen testing approach**

When choosing which testing approach to use, a number of factors, including scheduling issues, risk factors, costs, etc. were considered. The approach that I ended up going for was the third approach mentioned in section 1.3, and a more rigorous justification as to why that was the case can be found in the approach justification document [5].

# **2. Design and implement comprehensive test plans with instrumented code [20%]**

## **2.1 Construction of the test plan**

The entire test plan is outlined in the coursework test planning document [6]. In this document, several sections are covered, including ***requirements being considered***, ***priority and pre-requisites***, ***scaffolding and instrumentation***, and ***process and risk***.

## **2.2 Evaluation of the quality of the test plan**

A more rigorous evaluation can be found in the ***process and risk*** section of the coursework test planning document [6]. The test plan document provides an optimistic plan, but some of the risks discussed include various ***scheduling*** and ***technology risks***, as well as ***limitations of the instrumentation***.

## **2.3 Instrumentation of the code**

A more detailed explanation of the instrumentation of the code can be found in the ***scaffolding and instrumentation*** section of the coursework test planning document [6]. Some of the techniques discussed in relation to the chosen requirements include scaffolding, generating simulators for the software environment, and more details about the contents of the different tests.

## **2.4 Evaluation of the instrumentation**

A more rigorous evaluation can be found in the ***process and risk*** section (particularly the ***evaluation of instrumentation*** subsection) of the coursework test planning document [6]. Some of the limitations discussed include ***how representative the synthetic data is of the real data***, ***how adequately the test suite verifies the requirements***, as well as other risks/ additional costs that ought to be considered.

# **3. Apply a wide variety of testing techniques and compute test coverage and yield according to a variety of criteria [25%]**

## **3.1 Range of techniques**

A more rigorous description of all the techniques used can be found in the ***testing techniques used*** section of the coursework testing evidence document [7]. Some of the techniques discussed include ***systematic functional testing***, ***structural testing,*** and ***boundary value analysis***. In addition, the ***different levels of testing*** (unit, integration, and system) have also been considered, as well as ***ways to test the measurable attributes***. The full test suite can be found in the ILP test package [8].

## **3.2 Evaluation criteria for the adequacy of the testing**

The full evaluation of the adequacy of testing can be found in the ***evaluation of adequacy of testing*** section of the coursework testing evidence document [7]. Some of the evaluation criteria discussed include ***class coverage, statement coverage, specs coverage*** and ***test coverage***.

## **3.3 Results of testing**

A detailed log of all the results of running the test suite can be found in the ***results of testing*** section of the coursework testing evidence document [7]. Some of the issues include ***the drone entering no-fly-zones whilst making order deliveries*** and ***issues with order validation***, and the main way these issues were fixed were through manual debugging and/ or some automated piece of code that allowed me to identify the source of the issue (more details in the supporting document).

## **3.4 Evaluation of the results**

The more in-depth evaluation of the testing results can be found in the ***evaluation of testing results*** section of the coursework testing evidence document [7]. Some of the key points discussed relate to how ***the application of the evaluation criteria*** from section 3.2, and the ways in which they improve confidence in the testing.

# **4. Evaluate the limitations of a given testing process, using statistical methods where appropriate, and summarise outcomes [20%]**

## **4.1 Identifying gaps and omissions in the testing process**

All gaps and omissions in the testing process are discussed in more detail in the ***gaps and omissions*** section of the coursework testing evaluation document [9]. The main omissions identified are ***lack of unit and integration tests for certain components*** and ***lack of automated tests for command line input validation***.

## **4.2 Identifying target coverage/performance levels for the different testing procedures**

Here, we will consider the target levels for each of the four evaluation criteria mentioned in subsection 3.2 of the portfolio, as well as why these target levels are adequate. More details can be found in the ***target levels*** section of the coursework testing evaluation document [9].

## **4.3 Discussing how the testing carried out compares with the target levels**

All details on how the actual levels produced by our test suite compare with the target levels can be found in the ***how well does the testing meet the target levels*** section of the coursework testing evaluation document [9]. The main point made here was that although, in some cases, the actual values fell below the target levels, this was made up for by other means of manual testing to ensure that we can confidently say the system works as expected.

## **4.4 Discussion of what would be necessary to achieve the target levels**

A more detailed discussion on how to achieve/ exceed the target levels can be found in the ***improvements to achieve/ exceed target levels*** section of the coursework testing evaluation document [9]. The main ideas discussed here include ***additional tests being added to the test suite*** and ***generating more testing data***.

**5. Conduct reviews, inspections, and design and implement automated testing processes [15%]**

## **5.1 Identify and apply review criteria to selected parts of the code and identify issues in the code**

A more detailed description of the review criteria and issues identified can be found in the ***review criteria*** section of the coursework inspection and review document [10]. The review criteria used include ***checklist review*** for ***functionality*** and ***ad-hoc review*** for ***code quality***, and some of the issues identified include ***issues with Maven dependencies*** and ***small delays in creating the output files after execution finished***.

## **5.2 Construct an appropriate CI pipeline for the software**

A detailed plan of the CI pipeline for the given software can be found in the ***construct the pipeline*** section of the coursework inspection and review document [10]. Some of the actions include ***setting up the version control system (VCS)***, ***choosing and configuring a continuous integration (CI) tool***, ***configuring the test suite***, and ***deployment/ post-deployment steps***.

## **5.3 Automate some aspects of the testing**

We would ideally integrate all the tests in our current test suite with the CI pipeline. However, that may not be feasible, especially if more tests will be created in the future. Therefore, tests used to verify the safety-critical requirements should be prioritised in the CI pipeline. Additionally, having a combination of unit, integration, system, and other tests (using the tools described in coursework inspection and review document [10]) as part of our pipeline should also be a priority, since this would ensure more rigorous testing.

## **5.4 Demonstrate the CI pipeline functions as expected**

A detailed explanation of how the CI pipeline functions as expected can be found in the ***demonstrate the pipeline works as expected*** section of the coursework inspection and review document [10]. Some of the aspects discussed include ***handling failed tests***, ***handling syntax errors*** and ***handling coding style violations***.

**References**

1. ILP Coursework Specification, “ilp-coursework”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ilp-coursework.pdf>
2. ILP Coursework Requirements Specification, “ILP Project Requirements”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Project%20requirements.docx>
3. ILP Coursework Requirements Level, “ILP Requirements Level”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Requirements%20Level.docx>
4. ILP Coursework Testing Approaches, “ILP Testing Approaches”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Testing%20Approaches.docx>
5. ILP Coursework Testing Approach Justification, “ILP Testing Approach Justification”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Testing%20Approach%20Justification.docx>
6. ILP Coursework Test Planning Document, “ILP Planning document”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Planning%20document.docx>
7. ILP Coursework Testing Evidence Document, “ILP Testing Evidence”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Testing%20Evidence.docx>
8. ILP Project test package, <https://github.com/AndreasHiropedi/ILP_project/tree/main/src/test/java/uk/ac/ed/inf>
9. ILP Coursework Testing Evaluation Document, “ILP Testing Process Evaluation”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Testing%20Process%20Evaluation.docx>
10. ILP Coursework Inspection and Review Document, “ILP Inspection and Review”, <https://github.com/AndreasHiropedi/ILP_project/blob/main/software-testing-cw-documents/ILP%20Inspection%20and%20Review.docx>