Software Testing 2022: Your Portfolio

Outline of the Software Being Tested

The software being tested is a system for making lecture recordings interactive by inserting flashcards into videos whilst the student is watching them. The system has several components:

- **Database:** this is where the flashcard and user data, and flashcard review history are stored.
- **API:** the API is used within the system for data retrieval from the database. The API is also used from external websites to retrieve and embed flashcards for their videos.
- **Website:** users can review flashcards here later-on.
    - **Flashcard:** this component is both used (embedded) by external websites in the lecture videos and by the website of the system (for review sessions).
- **Authentication:** the system integrates a third-party authentication provider (Auth0).

Link to repository on GitHub. Here is an early prototype of the system.

Learning Outcomes

Please refer to the auditor guidance for the documents naming convention.

1. Analyze requirements to determine appropriate testing strategies [default 20%]

    1.1. Range of requirements, functional requirements, measurable quality attributes, qualitative requirements, …

    The portfolio includes a statement of requirements (see document LO1 1.1 1.2 EVI 1) which comprises a wide range of requirements, both functional (R1, R2, R5, R7) and qualitative (R3, R4, R6, R8). For qualitative requirements, the document presents measurable quality attributes which will be used for the test specification in LO2. Furthermore, the requirements address a wide range of properties: safety, robustness, accessibility, availability, performance, and correctness. The portfolio also showcases a requirements analysis (LO1 1.1 SC 1) which was conducted prior to creating the statement of requirements.

    1.2. Level of requirements, system, integration, unit.

    The requirements are applicable at all different levels. This is documented in the statement of requirements. The applicability of a requirement at a certain level is explicitly demonstrated by using appropriate terminology in the requirement description (e.g. "…should integrate" or "the … unit should…").

    1.3. Identifying test approach for chosen attributes.

    The portfolio identifies a wide range of testing approaches for the requirements at hand, including end-to-end testing, penetration testing, performance testing, and more. For qualitative requirements, we use the respective provided measurable attribute identified for LO1.1. We point out potential testing techniques (e.g. combinatorial, structural) where we are able to at this early stage. This is evidenced at LO1 1.3 1.4 EVI 1. Where the appropriateness of the approach is not sufficient, an alternative approach is provided.

    1.4. Assess the appropriateness of your chosen testing approach.

    In document LO1 1.3 1.4 EVI 1, we justify the choice of the chosen testing approaches. To do so, we first assess the appropriateness of every approach and consider its limitations as well. We found that we arrived at appropriate testing approaches which are adequate given the limited time available.

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

2.1. Construction of the test plan

As part of the portfolio, we constructed a test plan (see document LO2 2.1 2.2 2.3 2.4 EVI 1) which aims to be consistent with the requirements and test approaches identified in section 1 and further aims to ensure adequate testing. We first give each requirement a justified priority rating and identify its A&T needs. This leads to identified tasks (either research, scaffolding/implementation, or testing tasks) which are used to construct the rest of the test plan. We assigned each task to a lifecycle phase which facilitated schedule planning.

2.2. Evaluation of the quality of the test plan

The evaluation of the quality plan is also provided in the same document (see Risk Assessment / Evaluation of the Test Plan). We consider various risks (technological, personnel, scheduling) and provide mitigation approaches to ensure adequate testing. Considering further risks such as critical requirement risks would certainly make the evaluation even more thorough. However, we are convinced that the evaluation has been carried out as thoroughly as the provided time budget allowed us for; this was only possible by prioritising most relevant risks to our system.

2.3. Instrumentation of the code

As part of the test plan we created scaffolding and instrumentation. The process of doing so is presented in the same document (Implementation of Instrumentation of Scaffolding). We first identified tasks which require scaffolding or instrumentation. Then, for each task, we planned the process and implemented it, pointing to the results in the repository.

2.4. Evaluation of the instrumentation

The evaluation of the instrumentation is provided in the same document. For each instrumentation task, we first assess the adequacy of its implementation, and how good it is. We then provide pointers as to how we could improve the instrumentation to test the requirements more adequately if the time budget allowed us for doing so.

3. Apply a wide variety of testing techniques and compute test coverage and yield according to a variety of criteria [default 20%]

3.1. Range of techniques

The test specification in LO2 2.1 2.2 2.3 EVI 1, alongside with the testing documentation LO3 3.1 EVI 1, are evidence that the implementation of the planned testing went reasonably well. The scaffolding proved to be very useful. For some tests, we noticed during the test implementation that our implementation would not allow for adequate testing (e.g. T1). For those cases, we documented in the **History** column a change of approach and implementation.

3.2. Evaluation criteria for the adequacy of the testing

The evaluation criteria used for each testing method are documented in LO3 3.4 EVI 1. The choice of these criteria was motivated by the general applicability of them. While test-method-specific criteria might lead to a more detailed and overall more insightful evaluation, we argue that keeping the criteria this high-level and applicable to different kinds of testing methods, we streamlined the evaluation process and made it more efficient. This was a valuable trade-off given the limited time available.

3.3. Results of testing

The testing results indicate an overall passing and adequate system which fulfills the requirements according to the implemented tests. For some tests, we had to implement direct system changes to make tests pass. This is documented in the evidence. More detailed work that communicates the results effectively is evidenced in LO3 3.3 EVI 1.

3.4. Evaluation of the results

The application, along with detailed results, of the chosen evaluation techniques from (3.2) is documented in LO3 3.4 EVI 1. Overall, we used well-chosen tests which led to usable (4) results. Test-specific evaluation criteria were fulfilled with the results. Generally, the results were difficult to evaluate with some of the evaluation criteria (e.g. precision) which is explained in the document. This suggests that next time we should perhaps choose different evaluation criteria.

4. Evaluate the limitations of a given testing process, using statistical methods where appropriate, and summarise outcomes. [default 20%]
    4.1. Identifying gaps and omissions in the testing process
    Any omissions or deficiencies in the testing carried out and how it could be improved are documented in LO4 4.1 EVI 1.
    4.2. Identifying target coverage/performance levels for the different testing procedures
    We are aware that in a typical production environment we would want to strive for a "good-practice" coverage of 80-90% as achieving 100% line coverage can be complex for more sophisticated systems. We did not set any specific target levels for the tests due to the anticipated failure of computing test coverage given the system setup (see comment above). In LO4 4.1 EVI 1 we comment on why setting such a target level would not make much sense for how the test suite was developed. This is a case where the measurement of measurable attributes of the code was more challenging than meeting functional requirements, we stated during LO1. This is how we justify not investing time and cost into the aspect of test coverage.
    4.3. Discussing how the testing carried out compares with the target levels
    Given the points discussed in 4.1 and 4.2, we can deduct that it is very difficult to tell whether we met any testing targets or not due to the lack of a tool to calculate overage for us. We could attempt a manual approach, but we did not consider this an approach worth the trade-off for development time. This being said, we can say that our work probably fell short in this regard. Note that we are aware of the typical target levels and what we would need to achieve in a production setting (see 4.2).
    4.4. Discussion of what would be necessary to achieve the target levels.
    The portfolio provides a plan with actionable steps which should help achieve the target levels, in document LO4 4.4 EVI 1. While these steps seem useful, they would be time-consuming and cost-expensive to implement. Therefore we decide to not carry them out for this portfolio.
5. Conduct reviews, inspections, and design and implement automated testing processes. [default 20%]
    5.1. Identify and apply review criteria to selected parts of the code and identify issues in the code. [default 20%]
    We identify and apply review criteria in document LO5 5.1 EVI 1. Review techniques found for appropriate were a classic checklist inspection and abstract-driven reading. Note that we did not exhaustively inspect important artefacts but rather chose two different types (testplan and system unit) to demonstrate necessary skills within the given time limit.
    5.2. Construct an appropriate CI pipeline for the software
    The description of an appropriate CI pipeline for the software is provided in document LO5 5.2 EVI 1. This includes specific details on how we would construct the pipeline for GitHub.
    5.3. Automate some aspects of the testing
    The discussion and planning of embedding of testing in the pipeline is evidenced in LO5 5.3 EVI 1.
    5.4. Demonstrate the CI pipeline functions as expected.
    A description of how the pipeline would operate and what issues we think it would identify is provided in document LO5 5.4 EVI 1.