Quality Management Plan Version 0.1

CSC 480/HCI 521: Software Design

Table of Contents

1. Quality Management Approach

Our goal is to ensure that scantrons are evaluated correctly and consistently. There should be a low margin of error (TBD), ideally as close to zero errors as possible. It is very important that a students grade is correct, and that the scantron answers are scanned correctly! Each component of this system shall be tested thoroughly and the system should display no incorrect data. The testing that shall be done should ensure that each component works as expected.

1.1 Synopsis

Creating a software which calculates student's grades is highly beneficial considering the tedious task at hand, however, extensive testing is needed to ensure the system requirements are operating correctly and efficiently.

1.2 Testing Plan

We will use **Spock** to unit test Java code, we will use **PyTest** to test Python code, and we will be using **Travis Continuous Integration** to help streamline our pull requests and code changes.

Two people will check pull requests. A member of QA to run various tests and a different member of the team pushing the changes, to ensure the code changes are up to the teams standards.

Possible tests

Possible tests to be chosen from for testing each component. We most likely will not create each possible test for each component as that would be an insane amount of tests

Almost all code should have unit testing at minimum. All other options for testing are listed below:

1. Specification Based Testing
2. Control Flow Based Testing
    2.1. Minimal Multiple Condition Testing

(This test subsumes branch testing which subsumes statement testing, making those tests unnecessary if this is done.)

    2.2. Path Coverage Testing
    2.3. Data Flow Testing
3. Class Testing
    3.1. Category Partition Test
    3.2. Source Code Test
    3.3. Polymorphism Test
4. Class Scope Test
    4.1. State Transition Matrix
    4.2. All Events Criterion
    4.3. Flattened Class Scope Test
    4.4. Class Interaction Test
5. Unit Tests
6. Integration Tests
7. System Tests
8. Acceptance Test
9. Ongoing Tests and Future Testability (Excitor Not MVP)

Testing each Use Case In **SRS**.

Each use-case must work as expected. Using use-cases as a basis will allow us to test every potential scenario of the system. This will also enable us to test how the different components interact, not just the components acting alone that unit testing will accomplish. This would be

1.3 Components

Not enough info yet

1.3.1 Scanner - The physical scanner where a professor will scan his/her scantron sheets.

1.3.2 Web App - The web page that will allow the user to control various aspects of their tests and grades.

1.3.3 Database - Where all the data and information will be stored.

1.3.4 Data Analyzer - The program that will “read” the scantrons and compute data and statistics.

1.4 Definitions

Term	Definition
SRS	SMARTron Software Requirements Specification **Last Referenced:** Feb 14th 2019 https://github.com/csc480-19s/SMARTron/blob/master/Project%20Artifacts/srs_v0.4.pdf

2. Quality Properties, Metrics, and Criteria

Property	Definition	Metric	Criterion
Correctness	The degree of quality and level of error-free the system is.	Based on the execution of test cases based off of the specifications provided by the requirements team..	• The system is considered correct if it runs on Bastians Macbook Pro • All test cases marked as having high criticality must pass. • Along with testing the Component must be successfully reviewed by a third party member from either the Database or Engine teams to be considered correct.
Usability	The degree to which the system is able to used with ease by the user	To determine usability, the system must follow usability guidelines • Satisfaction • Efficiency	• The system is considered to be usable if the stakeholder's satisfied with the usability of the design • Efficiency: The system should complete faster than previous method of grading exams, with less effort. The system will also grade tests as stated in the accuracy property. • Success rate: The user success rate on specific tasks. Tasks will be based on system requirements • Time: The less time it takes to complete a task, is better for the user
Security	All the measures that are taken to protect data, or to ensure that only users with appropriate permission have access.	Ability of the system to determine authorization and be usable only by said authorized users as well as only send data to authorized users. The system must prevent information loss.	• The system is considered Secure if the data is only accessible by the authorized user that requested the data. • The system has to retain data until user indicates otherwise so that results can be altered by the user.
Traceability	The ability to link product requirements back to stakeholders' rationales and towards corresponding design artifacts, code, and test cases.	Based off of the traceability matrix are all use cases matched with a test case.	• The system is considered to conform to traceability if all of the requirements dictated by the stakeholder are reflected by the code.
Accuracy	The degree to which the result of a measurement, calculation, or specification conforms to the correct value or a standard.	Ability of the system to grade scantron sheets with precision and to detect defects in the sheets.	• System is considered accurate when 90% of well marked scantron sheets are graded correctly. • Scantron sheets that are not well marked or have some type of defect are not expected to be graded accurately but must be detected as defects for manual review.

https://en.wikipedia.org/wiki/List_of_system_quality_attributes

3. Testing

4. Results

5. Conclusion