Testing Document

Describes the test cases that have been implemented • Includes discussion on how these test cases were derived • Includes discussion on why these test cases are sufficient • Includes test coverage discussion (next week's topic) • No particular format required for this course

We don't have any test cases because nothing has been implemented yet!

Test Plan

Introduction:

This document explains our strategies on how we test our application, the resources we have used, the test environment that our tests were performed in, and the limitations of the testing and the schedule of testing activities.

Developing test cases

As new classes are created, and new methods need to be implemented we will use the Test-Driven Development (TDD) approach to implement our methods. Tests dictate what a piece of code does because the code will need to pass whatever test we are trying to account for. TDD is developed in multiple stages. We begin by writing a single test. Next, we compile it, and at this point the test should not compile because we have not implemented any code. Now we will implement the code, but only enough for the test to compile. At this stage when we run the test, it should fail. Next, we will complete implementing the code so that the test will pass. By this stage we want to clean up the code and make it more read-able and user friendly for other developers to work with in the future. This process repeats for new tests.

Test cases that have been implemented

Currently there have not been any test cases implemented.

Why are these test cases sufficient?

Assumptions while testing the application

List of test cases included in testing the application

List of features to be tested

- Creating any number of scenarios, the teacher would like to create
- Enabling the teacher to create a scenario as long or short as they would like
- Being able to edit the size of a scenario (allowing the teacher to make changes to their existing scenario)
- Making buttons functional
- Ensuring users can interact with GUI properly

What sort of approach to use while testing the software

List of deliverables that need to be tested

The resources allocated for testing the application

Any risks involved during the testing process

Risks of TDD are typically going to be a result of not accounting for tests that should be included when implementing a piece of code. We know what we want to test, but is that all we want to test, or have we left something critical out?

A Schedule of tasks and milestones to be achieved

Test Scenario

A test scenario considers what area in the application should be tested. A test scenario is a test case but unlike a test case, a test scenario requires multiple steps. The key thing to consider is that we execute a sequence of tests in an order. In other words, each test is dependent on the output of the previous test. As a tester, you may put yourself in the end user's shoes and figure out the real-world **scenarios** and use **cases** of the Application Under**test**.

Test Case

Test cases are used to ensure that a program either passes or fails in terms of functionality. Currently we do not have any test cases developed.

Components included in a test case:

- Test case ID
- Product module
- Product version
- Revision history
- Purpose
- Assumptions
- Pre-conditions
- Steps
- Expected outcome
- Actual outcome
- Post-conditions

Requirements Traceability Matrix (RTM)

The Requirements Traceability Matrix (RTM) is a table that is used to trace the requirements during the Software Development Cycle. The requirements of the matrix are associated with a test case so that the testing can done as per the requirements. The main goal of the matrix is to:

- Make sure the software is developed as per the mentioned requirements.
- Helps in finding the root cause of any bug.
- Helps in tracing the developed documents during different phases of SDLC.