Software Test Plan

[Type the document subtitle]

[Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.]

Jeff Steel

Table of Contents

Background

- A brief background and history of the testing project, which should include:
    - Brief description of the purpose of the AUT (Application Under Test)
    - Definitive version information for the AUT
    - Any other relevant supporting information

About a 4 month ago the company approached by Laura's landscaping with project to create an application to add to her business workflow. This included that all documents, booking … be recorded and sorted via a software solution.

At this state in the project the initial application has been build and is ready for TDD.

The TDD will most focus on the application core features (database, events, data mutations…) as opposed to the GUI at this stage.

Test Objectives

- What's the purpose of the test planning process and the software test plan?
- What are the quality and reliability goals of the product?

The purpose of the testing plan is to insure that the developers are aware of what is needing attention and resources in order for the product to meet the determined standard of both the company and client.

During the development process the dev, analyst, business teams were given certain checkpoints and a architectural solution to how and when the code should be doing what.

The point of the testing plan is to insure that the specified features have been implemented and are operation as expected. This becomes especially important when data is being mutated and stored.

Definitions

- This section lists the common terms and their meaning that applies to the project being tested or developed.
- Here's a list of a few common terms and very loose definitions

    - Build. A compilation of code and content that the programmers put together to be tested.
    - Test release document (TRD). A document that the programmers release with each build stating what's new, different, fixed, and ready for testing.
    - Alpha release. A very early build intended for limited distribution to a few key customers and to marketing for demonstration purposes. It's not intended to be used in a real-world situation.
    - Beta release. The formal build intended for widespread distribution to potential customers
    - Spec complete. A schedule date when the specification is supposedly complete and will no longer change.
    - Feature complete. A schedule date when the programmers will stop adding new features to the code and concentrate on fixing bugs.

- Bug committee. A group made up of the test manager, project manager, development manager, and product support manager that meets weekly to review the bugs and determine which ones to fix and how they should be fixed.

Assumptions and Exclusions

- This section lists the assumptions that have been made for the purposes of the testing project. It also lists the specific aspects of the AUT to be tested.
  - For example, the following assumptions have been made:
    - The testing project will test:
      - The functionality of the Order form
      - The interface between Order form and Customer form
      - The interface between Order form and Suppliers form
    -

- Items to be tested:
  - The launching of the application
  - The GUI features
    - Labels and inputs are align
    - Buttons are in functionality order
    - The correct inputs have mapped default values that will NOT cause problems with queries, searches, sorting …
  - The correct amount of pages / tabs that are correct in their functionality and content the core data they need to operate
  - The database is connected and can be disconnected with appropriate responses and error handling i.e. the application does not crash and the current data is not lost.
  - The data base is compressing data
  - The data base is storing and able to sort the various data
  - The database server will be live during the entire testing process.
  - Access will be available to the test rig outside of normal working hours.

- This section also lists the items that will not be included in the testing project.
  - For example, the following exclusions have been made:
    - The testing project will not test:
      - Performance aspects of the Order form
      - Load aspects of the Order form
      - The interface between the Order form and Main form
    - It is not necessary for a User Representative to be present during testing.
    - Testing tool support will not be necessary for this testing project.

- Items not tested
  - Alll non / future moduels / feature i.e.
    - All data formatters for emails, reports…
  - Automated booking features

- Customer automation
- Advanced GUI layouts w/ automation tools…
- Networking
    - Https
    - Ssh
    - Shell
- Deployment
    - System independence
    - Dependencies
    - Network adapters

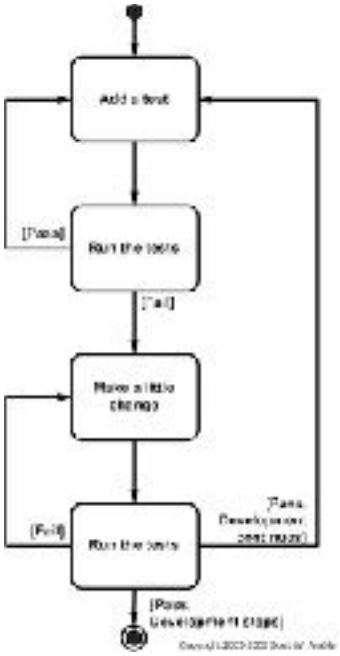
Test Phases

- This section identifies the stages for testing a particular project or AUT.
- The two very important concepts associated with the test phases are the entrance and exit criteria.
    - **Entrance Criteria**
        - This section lists the entry criteria for the testing project.
        - For example, the following entry criteria have been specified:
            - The test environment has been set up (and tested).
            - The amended (to reflect changes to the AUT) Requirements Specification Document has been received.
            - The final copy of the amended AUT has been received from the Development Team and has been installed in the test environment.
            - The latest version of the operating system required for testing has been installed in the test environment.
            - The test project has been appropriately resourced.
    - **Exit Criteria**
        - This section lists the exit criteria for halting the testing process.
        - For example, the following entry criteria have been specified:
            - No outstanding defects
            - Observed defects below an agreed-on profile, such as:
                - No critical defects, and less than n severe defects
                - N hours testing time without observing a defect
                - A complete run (or rerun) through the test script with no defects
            - Some budgetary criterion, such as:
                - The assigned testing budget has been exceeded
                - The assigned testing effort had been exceeded
            - Some scheduling criterion, such as:

    - Testing must stop to allow planned delivery of AUT
    - Testing must stop to allow planned development of AUT (e.g., Unit Testing must be complete by a certain date to allow integration of the AUT to begin)
- Testing must stop because the AUT is unfit for this level of testing; that, the AUT is of such poor quality that further testing would be a waste of effort).

Test Strategy

- This section describes the approach that the test team will use to test the software both overall and in each phase.
- The test team needs to decide if it's better to use black-box testing or white-box testing or a mix of both techniques and when to apply each and to which parts of the software.
- Will you be using automating tools? If so, do they need to be developed or can existing solutions be purchased? What is the name of the testing tool?



http://agiledata.org/essays/tdd.html

TDD is an iterative development process. Each iteration starts with a set of tests written for a new piece of functionality. This process is usually accomplished with inbuilt functions that take a input and return the output. It is then compared using in language comparison operators to an expected output.

Java uses Unit to accomplish this, it is also in built into Eclipse IDE.

TDD can be used as both Black box and White Box, if the developers are using TDD it is usually White box.

If testers are using TDD the most common is to compare business teams input to the expected business output. This is black box testing.

Resource Requirements

- Planning the resource requirements is the process of deciding what's necessary to accomplish the testing strategy.
- For example:
    - **People**. How many, what experience, what expertise? Should they be full-time, part-time, contract, students?
    - As the system is designed around pre deterrent requirement i.e. by the clients business and related business work flows, customer bases … the people who preferably testing the system are the developing companies analyst team along side the business team who were tasked with getting and meeting with the client, this preferably should be done by the team that initially dealt with the client
    - After the testing is complete the analyst should analyse and consult with project manages which in tern at later time would then call in technical team in order to plan and resolve the bugs / issues fourd
    - **Equipment**. Computers, test hardware, printers, tools.
    - Is most if not all the application is software ensuring this it will operate on the clients system, both hardware and installed software. This could include
        - Java se and ee runtimes
        - Related framework dependencies
        - Permissions
        - ~ Most of this could be tested in Docker or an live demonstration on the clients computer
    - **Office and lab space**. Where will they be located? How big will they be? How will they be arranged?
    - Most of the testing would depend on how the companies eco systems work, this should be discussed with the team manager along with the required testers and preferably and system/'s admin to ensure that all systems are configured correctly.
    - This is best planned ahead of time in order for all team/'s to be aware.
    - **Software**. Word processors, databases, custom tools. What will be purchased, what needs to be written?
    - Jira has an excellent documenting system inbuilt it the software which can be easily distributed to the related teams.
    - More humble and on hand tools could include pre printed documentation for had writing notes.
    - Also word / excel templates could be used instead as they have a pre determined design that the team/'s would be aware of.

- **Outsource companies**. Will they be used? What criteria will be used for choosing them? How much will they cost?
- The only time outsource to 3rd parties should be if the in-house team is not specialised in the a certain area/'s i.e. Microsoft internals along side Mac osx internals, that would be required by the application/'s
- **Miscellaneous supplies**. Disks, phones, reference books, training material. What else might be necessary over the course of the project?
- Work phones and related communication devices are very useful as they allow for on call sprints when a certain event occurs, in this case a few dev team members could be on call when the test team runs their tests incase of error and or guidances

Roles and Responsibilities

- Planning the tester assignments identifies the testers responsible for each area of the software and for each testable feature.
- This section specifies the names, roles, and responsibilities of the staff involved in the testing project.

    - Test Team Leader – Jame Ashley
        - Responsible for managing the day-to-day task of the Test Team.
        - Produces test plan format
        - Reporting of the results
        - To present the documents to the team manager/
    - Test Analyst – Leroy Jenkins
        - Responsible for developing test cases and test data
        - Arrange documentation layout
        - To compare the test modules for validity and compare business results against application output
    - Tester – Steve Job
        - Liase with User Representative
        - Performs testing
        - Write blackbox tests
        - Compare to developer white box testing

Test Cases

- The test planning process will decide what approach will be used to write them, where the test cases will be stored, and how they'll be used and maintained.
- This document is created separately and you need to specify the test case document name here so the testing team can use it.

see testCases.xmls in folder

Bug Reporting / Problem Reporting

- This section documents the manner in which defects observed during testing are recorded and reported to the appropriate staff.
- Exactly what process will be used to manage the bugs needs to be planned so that each and every bug is tracked from when it's found to when it's fixed – and never, ever forgotten.
- For example, all defects observed during System Testing will be recorded on the appropriate Test Result Record Form. All such defects will be reviewed during the Post Test Review Meeting, where the Test Team Leader and the Development Team Leader will discuss the scope of remedial work to the AUT, timescales for correction, and any test case(s) that need to be rerun as a result of the remedial work.

To report bugs / issues that were discovered during the testing process would have a multi step reporting system. This is so developers can reproduce the bugs to pin point the issue/'s. How ever not all bugs are code logical bugs, devs could of simply or not so simple missed a criteria and or not understood what the criteria meant. Which includes the following:

- Time and location of test
- The state of the code, error messages, line numbers, stack calls…
- The test case along with and state, config files, program states I.e. after storing data …..
- A listing of similar bugs, preferably in a excel spread sheet which points to a detailed template filled in by the testers and team involved

Sign-Off

- <signature >