Group 3 Team 5

DB Edit Software

Test plan

Version <1.0>

<April 2020>

Document Control

Approval

The Guidance Team and the customer shall approve this document.

Document Change Control

|  |  |
| --- | --- |
| Initial Release: | 1.0 |
| Current Release: | 1.0 |
| Indicator of Last Page in Document: | $ |
| Date of Last Review: |  |
| Date of Next Review: |  |
| Target Date for Next Update: |  |

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:

Dr. Steve Roach

Customer:

Software Team Members:

Nouri, Ali

Guajardo, Patricia S.

Sanchez, Briana

Change Summary

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Note: The template presented in this document was taken from:

Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323.

Note: The template presented in this document was taken from: Donaldson, S., and S. Siegel, *Successful Software Development*. Upper Saddle River, NJ: Prentice Hall, 2001, pp. 321-323 and modified by Humberto Mendoza and Steve Roach.

Supplementary information is from:

Pfleeger, S. *Software Engineering, Theory and Practice*. Upper Saddle River, NJ: Prentice Hall, 1998, p. 365.

Table of Contents

[Document Control ii](#_Toc22915465)

[Approval ii](#_Toc22915466)

[Document Change Control ii](#_Toc22915467)

[Distribution List ii](#_Toc22915468)

[Change Summary ii](#_Toc22915469)

[1. Introduction 1](#_Toc22915470)

[1.1. Purpose 1](#_Toc22915471)

[1.2. Scope 1](#_Toc22915472)

[1.3. System Overview 1](#_Toc22915473)

[1.4. Suspension and Exit Criteria 1](#_Toc22915474)

[1.5. Document Overview 1](#_Toc22915475)

[1.6. References 1](#_Toc22915476)

[2. Test Items and Features 2](#_Toc22915477)

[3. Testing Approach 3](#_Toc22915478)

[4. Test XX 4](#_Toc22915479)

[4.1. Test <<test id>> 4](#_Toc22915480)

[5. User Interface Testing 5](#_Toc22915481)

[6. Test Schedule 6](#_Toc22915482)

[7. Other Sections 7](#_Toc22915483)

[8. Appendix 8](#_Toc22915484)

# Introduction

In this test plan, we try to provide a comprehensive test suite for a database editor. Database editor is a software that stores data in XML files. There is no relation between the columns. This editor allows users to access the database, read, add, and modify the data. The goal of this test plan is to cover all possible bugs or glitches in the software. Here, we focus on edit the menu functionality of the software. We do not have any access to the code of the software. Thus, our approach is black-box testing. Each test cases explain with all the necessary details. The test cases are covered undo, copy row, paste row, insert row, delete row, and search functions in the editor.

## Purpose

This test plan contains all the requirements needed to test the software, a test suite that covers all editing menu functionality, and documentation of all the steps to apply each test case. The purpose of this project is to provide a comprehensive test plan document. This document covers all the steps that needed to test each test case, elaborate on the purpose of each test case, and analyze the performance of the system after each test case.

## Scope

This test plan started on April 8th and finished on April 21st. The test plan improved by the review of other team members and the writer used their feedback to provide a comprehensive test plan. GitHub was the platform to communicate between the team members.

## System Overview

The system is a database editor that uses XML to store the data. This system is an interface for users to manipulate the data in the database. User can add, remove, change, or search the data easier by using this system. When the user wants to start working with the editor, he/she must open a database edit window. This window displays the tables in a database. Initially, no database is loaded. There is a menu with File and Search. File has the options to open, close, and print tables. A user can open either a database description file or a data file. If a data file is opened, the corresponding database description is also opened. The database description window displays all the database tables. Selecting a table from this display opens the data table display in a new window. It is possible to open more than one database at a time. Opening a database table file results in the database description also being opened. It is possible to open more than one database at a time. All the windows associated with a specific database have the same color outline. Opening an already-opened table should result in the table editor window for that table being raised to the top of the screen and being visible.

## Suspension and Exit Criteria

The exit criteria for this test plan is all the high critical tests must be passed and more than 60% of the medium critical test must be passed. Otherwise, we consider the system as a failed project and the test must be repeated after debugging the problems.

## Document Overview

In chapter one, the system introduced, the scope of the project explained, the suspension and exit criteria specified, and an overview of the system provided. In chapter two, the test items and features explained. Chapter three is about the test approach and listed the test cases, the critical status of each test case, and demonstrated each test case objective. Chapter four each test case explained, the operation actions and the purpose of them provided. The expected results and the actual result can be found for each test case in this chapter. Finally, in Chapter five the test schedule is listed. The appendix is available in chapter six of this document.

## References

[1] Spec.docx

# Test Items and Features

In this test, our focus will be on editing menu operations. All the possible functions and operations must be tested. There are some common user actions that need to be tested. The functions that are provided on the menu are inserting, deleting, copying, pasting, Search, and undo. We will go through each of them and provide appropriate tests for each of these functions.

## Undo

This function reverses the last command or deletes the last entry the user typed. This function can be active by pointer or just press its shortcut (^z). This function works by storing the last changes made. By calling this function, the last changes will be overwritten on what we have.

## Copy Row

This function copies the selection from the table and stores it in the clipboard of the O.S. It can be paste in another cell or can be paste in space in another software or document. The shortcut to copy a row is (^c). The clipboard of the O.S. just can store one content. That means by copy two different objects at a different time, the second one will overwrite on the first one.

## Paste Row

This function pastes the content in the clipboard of the O.S. into the selected cell. This function can apply for multiple times. So, the users can paste the content into multiple cells without losing the content. This function can be called by using the pointer or using the (^p) shortcut.

## Insert Row

This function helps the users to add a new entity to the dataset. Technically it creates a new row in the table. This function stores the new entity by adding a new node to the XML file too. This function can be active by using a pointer or press the shortcut (^i).

## Delete Row

It deletes selected row from the table. This function removes an entity from the database. It can be called by pointer or pressing the shortcut (^d)

## Search

This function helps users to find a specific string they are looking for. This function doesn’t have any shortcut and must be called by the pointer. The GUI could provide a shortcut for this action, but it didn’t.

### Search and Replace Elements

Users can use the find function or they can use find and replace action. It helps the users to find specific data and replace the cell by another value. The ReplaceAll function is already implemented in the GUI. Thus, the users don’t need to replace the previous value by a new value one by one.

### Find Match Cases

Find Match Case option helps users to find the exact match they are looking for. This option helps them to find the result they want more accurate. It shows the cells that have the word inside them.

### Find Match Whole Words

This option helps the user to find the exact word they want on the table. This option just returns the cells that have the exact word the users searched.

# Testing Approach

In this test, we try to test the functionalities of the editing menu in the database editor software. The purpose of this testing is to check the functionality of the edit menu and find any possible scenarios that cause any error. We will use black-box testing as our approach. We do not have any access to the software codes, and this is the reason we chose black box testing. We will try different scenarios, inputs, and check the boundaries to be sure the edit menu works and all the functions inside the edit menu are working without any problem. We will categories the test cases by low, medium, and high critical condition. These labels help debuggers to address the problems based on the priority of fixing the issues.

Table 1: Test Plan

|  |  |  |
| --- | --- | --- |
| **TEST SUITE <Identifier>** | | |
| **Description of Test Suite** | **In this test suite, the focus is to test edit menu operations on the software. This test suite is included testing for inserting, deleting, copying, pasting, Search, and undo** | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| Test Case 1 | **Test the functionality of UNDO** | **High** |
| Test Case 2 | **Test the functionality of COPY ROW** | **Medium** |
| Test Case 3 | **Test the functionality of PASTE ROW** | **Medium** |
| Test Case 4 | **Test the functionality of INSERT ROW** | **High** |
| Test Case 5 | **Test the functionality of DELETE ROW** | **High** |
| Test Case 6 | **Test the functionality of SEARCH** | **High** |
| Test Case 7 | **Test the functionality of REPLACE** | **Medium** |
| Test Case 8 | **Test the functionality of EXACT MATCH** | **Medium** |
| Test Case 9 | **Test the functionality of MATCH WHOLE WORDS** | **Medium** |
| Test Case 10 | **Test the functionality of GUI RESPONSIBILITY** | **High** |

We provide 10 different test cases in our proposed Test Suite. We labeled them based on the critical status they have. We labeled high critical those functions which are the main editing functions of a table. The medium labels belong to those that help users to work faster or improving the user-friendly interface functions.

# Test Edit Menu Functions

In this section, we provide the requirements of each test case, the steps to apply those test cases, the inputs for each test case, and the expected output. We try to cover all the possible scenarios that cause any bugs or glitch in the software. In this section, each test case documented well, and we provided details and information to repeat the test later if it is necessary or analyze the results of any cases.

## Requirements

Before starting the test, please be sure that you have these following files in a same directory.

DbEdit.jar

TEST\_DB.xml

TEST\_DB\_\_BIG\_TABLE.XML

TEST\_DB\_\_CONSTRAINT\_TABLE.XML

TEST\_DB\_\_TYPE\_TABLE.XML

DB/ TEST\_DB\_\_TYPE\_TABLE.XML

You need to be sure that Java Compiler already is installed on your machine.

## Test <<test id>>

**Objective: <**< Define the objective of Test XX.Y. >>

**Notes:** <<This area provides general notes concerning the test procedure. Such notes might include comments on how to execute the test procedure, an estimate of the test duration, the requirements of the procedure tests, or a statement of resources needed for this test.>>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No.: << Unique test ID >> | | | | Current Status: << Passed / Failed / Pending >> | | |
| Test title: <<This line contains the long title of the test procedure.>> | | | | | | |
| Testing approach: <<Included in this section is a description of test harnesses, testing frameworks, environmental requirements, test tools and test automation that will be employed to achieve testing. Include naming conventions for tests and test scripts if appropriate. Provide requirements traceability and test priority.  >> | | | | | | |
| STEP  <<N>> | OPERATOR ACTION  Describe the actions taken by the person executing the test procedure. Include the test suite, or the name of the test file (in this case, the contents of the file should be given in the appendix). | PURPOSE  Describe the reason for the step. | | | EXEPCTED RESULTS  Describe the expected response of the system being tested to the action specified under OPERATOR ACTION. This should be derived from the SRS and SDD. Clearly indicate how we determine whether the step passes. | COMMENTS |
| Concluding Remarks: | | | | | | |
| Testing Team:  << List members of testing team and lead >> | | | Date Completed: | | | |

# User Interface Testing

<<This section focuses on the interaction between the user and the system. For testing the user interface, consider the following traits:

* Consistent terminology, shortcut keys, menu selections, and presentation
* Correct language, spelling, and grammar.
* Flexibility in navigation between windows and interface elements.
* Error handling that will inform user of critical operations.
* Follows standards and guidelines such as placement of scroll bars, windows, and menu items.

This section could be integrated into Section 4.

>>

# Test Schedule

<< Specify the schedule for testing activities. A table with the order and completion dates of the tests is useful. The table below might be useful.>>

|  |  |  |
| --- | --- | --- |
| **Task and date** | **People** | **Description** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Other Sections

<< Other sections that may appear in a test plan (but not required for this course) are:

* Test Management Requirements: how testing is to be managed; a delineation of responsibilities of each project organization involved with testing
* Staffing and training needs: delineate the responsibilities of those individuals who are to perform the testing, level of skill required, and training to be provided
* Environmental Requirements: describe the hardware (including communication and network equipment) needed to support testing; describe configuration of hardware components on which software and database to be tested are to operate.
* Software Requirements: describe the software needed to support testing; include the software code and databases that are object of the testing. Also include software tools such as compilers, CASE instruments and simulators that are needed to model the user’s operational environment.
* Risk and contingencies
* Cost: include an estimate of costs.
* Approvals
* Test Deliverables

>>

# Appendix

<< possibly more readable to put the expected output here and refer to it in the previous sections. Might also provide explicit directions for analysis of output, if it’s easier to read as an appendix or if analysis is post execution. >>

$