**File menu operations**

**Test plan**

**Version 2.0**

**4/19/2020**

**Document Control**

**Approval**

The Guidance Team and the customer shall approve this document.

**Document Change Control**

|  |  |
| --- | --- |
| Initial Release: | 04/11/2020 |
| Current Release: | 04/19/2020 |
| 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. Roach

Customer: Dr. Roach

Software Team Members: Brian A. Cardiel, Peter Hanson, Ricardo Sanchez

**Change Summary**

The following table details changes made between versions of this document

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Modifier | Description |
| 1.0 | 4/11/2020 | Brian Cardiel | Creation and adding of Section 1 |
| 2.0 | 4/19/2020 | Brian Cardiel | Modified Tests, Section 2, Section 3 |
|  |  |  |  |

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**](#_heading=h.30j0zll)

[**Approval ii**](#_heading=h.z337ya)

[**Document Change Control ii**](#_heading=h.3j2qqm3)

[**Distribution List ii**](#_heading=h.1y810tw)

[**Change Summary ii**](#_heading=h.4i7ojhp)

[**1.**](#_heading=h.3dy6vkm) **INTRODUCTION 1**

[**1.1.**](#_heading=h.1t3h5sf) **Purpose 1**

[**1.2.**](#_heading=h.4d34og8) **Scope 1**

[**1.3.**](#_heading=h.2s8eyo1) **System Overview 1**

[**1.4.**](#_heading=h.17dp8vu) **Suspension and Exit Criteria 1**

[**1.5.**](#_heading=h.3rdcrjn) **Document Overview 1**

[**1.6.**](#_heading=h.26in1rg) **References 1**

[**2.**](#_heading=h.2xcytpi) **TEST ITEMS AND FEATURES 2**

[**3.**](#_heading=h.lnxbz9) **TESTING APPROACH 3**

[**4.**](#_heading=h.1ci93xb) **TEST XX 4**

[**4.1.**](#_heading=h.3whwml4) **Test <<test id>> 4**

[**5.**](#_heading=h.2bn6wsx) **USER INTERFACE TESTING 5**

[**6.**](#_heading=h.1ksv4uv) **TEST SCHEDULE 6**

[**7.**](#_heading=h.44sinio) **OTHER SECTIONS 7**

[**8.**](#_heading=h.qsh70q) **APPENDIX 8**

# **Introduction**

This project focuses on the testing of a script to provide software quality assurance. The system to be tested is “dbEdit”. The testing approach is …

## **Purpose**

The purpose of this file is to provide documentation of the tests and test types performed on the “dbEdit” script. This test plan will cover black box testing and describe the application from the user’s point on view. The test plan will focus on the functionality of the application.

## **Scope**

(coming on V3 update 4/22/2020)

## **System Overview**

(coming on V3 update 4/22/2020)

## **Suspension and Exit Criteria**

(coming on V3 update 4/22/2020)

## **Document Overview**

The remainder of the document is composed of the following sections:

Section 2: Test of the functionality on the database table viewer.

Section 3: Explanation of the testing approach taken when testing the application.

## **References**

[1] Spec.docx provided by professor, Dr. Roach.

[2] GitHub provided and initialized by professor, Dr. Roach.

[3] dbEdit.jar provided by professor, Dr. Roach.

# **Test Items and Features**

The author of this test plan belongs to group 1, this section will describe the test items for group 1 only. The task for group 1 is to focus on file menu operations excluding compare file to other version, check for duplicate rows, and filter. This includes the search menu option on the database window.

* The Save an edited file function allows the user to modify a file, then to save the changes made to that file.
* The Save as option allows the user to save the current project as a new project with a different name or even different type.
* The Print to fit functionality allows the user to print the current project with no white space.
* Print Normal Fit functionality allows the user to print the current project or save as a pdf.
* Print CSV converts the current project to a CSV file so the user can print it as a CSV table.

# **Testing Approach**

These tests are designed to test file menu operations. These tests are significant types of operations, 100% of them are critical, if any of the critical tests can’t be completed, i.e. they fail, testing must continue until they all pass.

**Table 1: Test Plan**

|  |  |  |
| --- | --- | --- |
| **TEST SUITE <Identifier>** | | |
| **Description of Test Suite** | The test suite will focus on testing the functionality of the buttons located on the file menu. | |
| **Test Case Identifier** | **Objective** | **Criticality** |
| 1 | **Save an edited file** | **High** |
| 2 | **Save as** | **High** |
| 3 | **Print to Fit** | **High** |
| 4 | **Print Normal** | **High** |
| 5 | **Print CSV** | **High** |

## **Test 1**

**Objective:** The objective of this rest case is to test the Save functionality, confirm whether an edit to a file can be saved.

**Notes:** Screenshots were provided to display expected results, the expected results are the same for any operating system that can run the dbEdir.jar file but the screenshots may vary since the screenshots are the results of a Windows OS. It is recommended to zoom into the screenshots if it’s difficult to read.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No.: 1 | | | | | Current Status: Pending | | |
| Test title: **Save functionality** | | | | | | | |
| Testing approach: The following tests the save functionality by adding new given input to a table, then checks if the given input was saved on the table. The screenshots on the “Expected results” column should match after performing the corresponding operator action.  Testing approach: | | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | EXPECTED RESULTS | | | COMMENTS | |
| 1 | Run “dbEdit” executive Jar file by double clicking on the file. | Initial condition |  | | |  | |
| 2 | Click “file” tab then click the “open” option | To open file explorer so a file can be opened. |  | | | Make sure “Look in” dropdown displays the address where the project’s xml files are located on your computer. | |
| 3 | Search for the “TEST\_DB\_\_CONSTRAINT\_TABLE” XML and double click the file. | Open XML file so tables can be loaded. |  | | | There should be 2 windows open, the window that open when the program was started, and the window with the xml content that has red and yellow colors | |
| 4 | On the 2nd column “Description” go 1 row down and double click on it. Erase text inside and type “Save functionality test”. | Make a change to the XML file. |  | | |  | |
| 5 | Click on the “file” tab and click on the “save” button | Saves change to the XML file. |  | | |  | |
| 6 | Enter on the Author input box “tester”, enter on the Description input box “save test” and click the “ok” button. | The program records the author and a description record. | Notification of file being saved. | | |  | |
| 7 | Click ok on the “file Write” window. Then click the ‘X’ button on the table view (where you entered “Save functionality test”) to close it. Last close the main window by pressing the X button. | Close program completely to restart. | Program is closed. | | |  | |
| 8 | Repeat step 1 to 3. | Open XML that changes has been made previously. |  | | | Notice how after reopening the file the changes made previously were saved. | |
| Concluding Remarks: | | | | | | | |
| Testing Team:  Producer: Brian A. Cardiel  Evaluators: Peter Hanson, Ricardo Sanchez | | | | Date Completed:  TBD | | | |

## **Test 2**

**Objective:** The objective of this rest case is to test the “save as” functionality, confirm that you can save a file with a different name.

**Notes:** Repeating steps 1 to 3 from test 1 are required before beginning this test.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No.: 2 | | | | | Current Status: Pending | | |
| Test title: Using the save as functionality. | | | | | | | |
| Testing approach: Test ensures that the save as functionality is working successfully. The screenshots on the “Expected results” column should match after performing the corresponding operator action. | | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | EXPECTED RESULTS | | | COMMENTS | |
| 1 | Click on the “file” tab and click the “save as” button. | Triggers the save as functionality. |  | | |  | |
| 2 | Enter “test2File.xml” and click the “save” button. | Program is saving a new copy of the opened xml as “test2File.xml” on the address displayed on the file explorer. |  | | | Remember the computer address where the file is being saved. | |
| 3 | Enter on the Author input box “tester”, enter on the Description input box “save as test”. | The program records the author and a description record. | Notification of file being saved. | | |  | |
| 4 | Click ok on the “file Write” window. | Closes the “file write” window. | File write window disappears. XML window is still open. | | |  | |
| 5 | Close everything and restart the program again. | Reset the system | Program is closed. | | |  | |
| 6 | Click on “File” tab and click “open” button.  . | Starts file explorer |  | | |  | |
| 7 | Go to computer address where the “test2File.xml” was created and double click the xml file. | Tests if the “test2File.xml” was correctly saved and if it opens. |  | | | Notice the name of the file on the blue bar. | |
| Concluding Remarks: | | | | | | | |
| Testing Team:  Producer: Brian A. Cardiel  Evaluators: Peter Hanson, Ricardo Sanchez | | | | Date Completed:  TBD | | | |

## **Test 3**

**Objective:** The objective of this rest case is to test the “print to fit” functionality, confirm whether a file can be printed.

**Notes:** Repeating steps 1 to 3 from test 1 are required before beginning this test.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No.: 3 | | | | | Current Status: Pending | | |
| Test title: Check the “Print to fit” functionality | | | | | | | |
| Testing approach: Test ensures that when the “print to fit” functionality successfully prints a table on a given xml. . The screenshots on the “Expected results” column should match after performing the corresponding operator action. | | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | EXPECTED RESULTS | | | COMMENTS | |
| 1 | Click on the “file” tab and click the “Print to fit” button | Trigger the “print to fit” function. |  | | |  | |
| 2 | Click on the drop down button and select “Microsoft Print to PDF” and click on it. | Instead of physical printing it turns the print to a digital pdf file. |  | | | The “Microsoft print to PDF” may have already been preselected. This step makes sure that option is selected. | |
| 3 | Click “print” button. Window explorer opens, select destination of pdf file. Enter “test3” on file name and click “save” button. | Saves a pdf of the xml tables on selected computer directory. |  | | | 1) Alert message will appear as seen on the 1st picture.  2) File explorer appears as seen on 2nd picture.  3) Once “save” button is click the user will be back to the program and the file explorer will close. | |
| 4 | Go to the directory of the “test3” pdf and double click it | Open the digital print of test 3 table. |  | | | Notice that it’s a pdf format. This Simulates what it would have been physically printed as a print to fit. | |
| Concluding Remarks: | | | | | | | |
| Testing Team:  Producer: Brian A. Cardiel  Evaluators: Peter Hanson, Ricardo Sanchez | | | | Date Completed:  TBD | | | |

## **Test 4**

**Objective:** The objective of this rest case is to test the “print normal” functionality, confirm whether a file can be printed.

**Notes:** Repeating steps 1 to 3 from test 1 are required before beginning this test.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No.: 4 | | | | | Current Status: Pending | | |
| Test title: Check the “Print to fit” functionality | | | | | | | |
| Testing approach: Test ensures that when the “print to fit” functionality successfully prints a table on a given xml. . The screenshots on the “Expected results” column should match after performing the corresponding operator action. | | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | EXPECTED RESULTS | | | COMMENTS | |
| 1 | Click on the “file” tab and click the “Print Normal” button | Trigger the “print normal” function. |  | | |  | |
| 2 | Click on the drop down button and select “Microsoft Print to PDF” and click on it. | Instead of physical printing it turns the print to a digital pdf file. |  | | | The “Microsoft print to PDF” may have already been preselected. This step makes sure that option is selected. | |
| 3 | Click “print” button. Window explorer opens, select destination of pdf file. Enter “test4” on file name and click “save” button. | Saves a pdf of the xml tables on selected computer directory. |  | | | 1) Alert message will appear as seen on the 1st picture.  2) File explorer appears as seen on 2nd picture.  3) Once “save” button is click the user will be back to the program and the file explorer will close. | |
| 4 | Go to the directory of the “test4” pdf and double click it | Open the digital print of test 3 table. |  | | | Notice that it’s a pdf format. This Simulates what it would have been physically printed as a print to fit. | |
| Concluding Remarks: | | | | | | | |
| Testing Team:  Producer: Brian A. Cardiel  Evaluators: Peter Hanson, Ricardo Sanchez | | | | Date Completed:  TBD | | | |

## **Test 5**

**Objective:** The objective of this rest case is to test the “Print CSV” functionality, confirm that a file is converted to CSV format so it can be printed.

**Notes:** Repeating steps 1 to 3 from test 1 are required before beginning this test.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No.: 4 | | | | | Current Status: Pending | | |
| Test title: Check the “Print CSV” functionality | | | | | | | |
| Testing approach: Test ensures that when the “print to fit” functionality successfully prints a table on a given xml. | | | | | | | |
| STEP | OPERATOR ACTION | PURPOSE | EXPECTED RESULTS | | | COMMENTS | |
| 1 | Click on the “file” tab and click the “Print csv” button. | Trigger the “print normal” function, a CSV file will be generated. |  | | | Remember the directory that the message displays. | |
| 2 | Click the “ok” button on the message. The message will disappear, go to the directory that the message. | Got to directory were CSV was generated. |  | | |  | |
| 3 | Find the csv file generated, the file should be named “TEST\_DB\_CONSTRANT\_TABLE” and double click it to open. | Open CSV |  | | | Notice that the XML was converted to a CSV file so the user can print it as a table. | |
| Concluding Remarks: | | | | | | | |
| Testing Team:  Producer: Brian A. Cardiel  Evaluators: Peter Hanson, Ricardo Sanchez | | | | Date Completed:  TBD | | | |

# **Test Schedule**

<<>>

|  |  |  |
| --- | --- | --- |
| **Task and date** | **People** | **Description** |
| Test 1 TBA | TBA | .. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# **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. >>