Vormund

Test Plan

Team 13

Xinpi Du

Art Malinin

Wei Haow Tan

Jun Xiang Tee

Matt Walters

Lirong Yuan

Legend:

Classification of severity: S1 -- Critical to the success of the software.

S2 -- Important.

S3 -- Software could function even with defect in it.

There are several test cases that are very similar between database, GUI, and Access API. This is because a similar error or result could be caused by any of the three. For example, if the data of a NOTE is displayed incorrectly, it could be an incorrect query retrieving the wrong information, or it could be incorrectly stored in the database itself.

Requirements and test cases:

1. GUI

A. Menus 001, Drop Down Menu 1, S1

B. Select a category from drop down menu 1.

C. The list of entries that have been added to that category should be populated to drop down menu 2.

A. Menus 002, Drop Down Menu 2, S1

B. Select an entry from drop down menu 2.

C. Correct entry should appear on the textview.

A. Buttons 001, Update Button, S1

B. Open an entry by selecting values in drop down menu 1 and drop down menu 2, click on "Update" button.

C. A new window should appear with the previous entry content, and users should be able to edit it and click "Done" button.

A. Buttons 002, Update Button, S1

B. Open an entry by selecting values in drop down menu 1 and drop down menu 2, click on "Update" button, edit content, and clicks on "Done" button.

C. The textview in the center should be updated with the newly entered information.

A. Buttons 003, Add Button, S1

B. Open an entry by selecting values in drop down menu 1 and drop down menu 2, click on "add"

button.

C. Correct entry should appear for user to insert a new record.

A. Buttons 004, Remove Button, S1

B. Open an entry by selecting values in drop down menu 1 and drop down menu 2, click on “Remove” button.

C. Down menu 1 remains unchanged, but drop down menu 2 will switch back to the default value. Contents in the text view will be cleared.

A. Buttons 005, Logout Button, S2

B. Login into the main page, click on “Logout” button.

C. Program should switch back to the login page. Entries in menus should be cleared.

A. User Creation 001, Registration, S1

B. Click “Create New”.

C. A window should pop up, asking for information.

A. User Creation 002, Registration, S1

B. After entering username and password information clicking “Ok”

C. View Should be updated

A. Updating/deleting a record 001, Records, S1

B. Click on username/password panel, click on a record, and click update

C. Menu should appear that will let you update the record

A. Menu 001, Menu, S1

B. Clicking on Settings on main panel

C. Should open up a settings window

A. Menu 002, Menu, S2

B. After clicking on Settings on main panel click on Delete All data button

C. Confirmation window should pop up

A. Menu 003, Menu, S2

B. After clicking on Settings on main panel, click on Delete all, click confirm

C. Another confirmation menu should pop up

A. Menu 004, Menu, S1

B. Click on Settings on main panel, click on Change Master Password

C. Menu should pop up

A. Menu 005, Menu, S1

B. After clicking on Change Master Password, enter new password and click ok

C. Should take you back to the main screen with changes made

A. Menu 006, Menu, S1

B. After entering a valid username/password hit login

C. Should successfully take you to main menu

A. Account Creation 001, Login, S1

B. Leave the username field blank, and hit register

C. Should get an error

A. Account Creation 002, Login, S1

B. Leave password field blank, and hit register

C. Should get an error

A. Account Creation 003, Login, S1

B. After entering a new username/password hit login

C. Should get a menu to pop up asking to register

2. Database

A. Creating Bank Information 001, Find Bank, S1

B. After creating a Bank, check the list for bank.

C. The bank should be added to the database.

A. Creating Bank Information 002, Find SSN, S1

B. After creating a Bank, click on the list item for the added Bank.

C. The Bank item should display the same information you added

A. Updating Bank Information 001, Update Bank, S1

B. After updating a bank, check the bank list for the bank.

C. The bank's information should be updated.

A. Deleting Bank Information 001, Remove Bank, S2

B. Check the bank list, select a bank, and click Remove.

C. The selected bank should be deleted from the database.

A. Creating Website Information 001, Find Website, S1

B. After creating a website, check the list for website.

C. The website should be added to the database.

A. Creating Website Information 002, Find SSN, S1

B. After creating a Website, click on the list item for the added Website.

C. The Website item should display the same information you added.

A. Updating Website Information 003, Update Website, S1

B. After updating a website, check the list for the website.

C. The website's information should be updated.

A. Deleting Website Information 004, Remove Website, S2

B. Check the website list, select a website, and click Remove.

C. The selected website should be deleted from the database.

A. Creating Notes Information 001, Find Notes, S1

B. After creating a notes, check the list for notes.

C. The notes should be added to the database.

A. Creating Notes Information 002, Find SSN, S1

B. After creating a Note, click on the list item for the added Note.

C. The Note should display the same information you added.

A. Updating Notes Information 003, Update Notes, S1

B. After updating an instance of notes, check the list for the notes.

C. The notes' information should be updated.

A. Deleting Notes Information 004, Remove Notes, S2

B. Check the notes list, select an instance of notes, and click Remove.

C. The selected notes should be deleted from the database.

A. Creating SSN Information 001, Find SSN, S1

B. After creating a SSN, check the list for SSN.

C. The SSN should be added to the database.

A. Creating SSN Information 002, Find SSN, S1

B. After creating a SSN, click on the list item for the added SSN.

C. The SSN shoulds display the same information you added.

A. Updating SSN Information 003, Update SSN, S1

B. After updating a SSN, check the list for the SSN.

C. The SSN's information should be updated.

A. Deleting SSN Information 004, Remove SSN, S2

B. Check the SSN list, select a SSN, and click Remove.

C. The selected SSN should be deleted from the database.

3. Encryption

A. Encrypting Schema 001, Encryption, S1

B. Execute |sqlite3 database_file| from command line.

C. The |database_file| should be encrypted and not readable by sqlite3.

A. Encrypting Schema 002, Encryption, S1

B. Sign up with valid username and password. Sign in with the same username and password.

C. Although encrypted, the database should be recoverable -- user should be able to sign in.

A. Encrypting Schema 003, Encryption, S1

B. Sign up with username and password. Sign in with the same username and an incorrect password.

C. Although encrypted, the database should be recoverable -- user should not be able to sign in.

A. Encrypting Schema 004, Encryption, S1

B. Sign up with username and password. Sign in with a different username and password.

C. Although encrypted, the database should be recoverable -- user should not be able to sign in.

A. Encrypting Schema 005, User account information, S1

B. Sign up with valid username and password that consist of all characters. Decrypt the |database_file|,

access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. Username and password should be stored and encrypted correctly. We cannot find the username or password in their original form.

A. Encrypting Schema 006, User account information, S1
B. Sign up with valid username and password that consist of all characters with special characters. Decrypt the |database_file|, access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. Username and password stored should be stored and encrypted correctly. We cannot find the username or password in their original form.

A. Encrypting Schema 007, User account information, S1
B. Sign up with valid username and password that consist of all characters with malicious characters that could trigger sql injection attack. Decrypt the |database_file|, access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. The program should not fail under the attack. Username and password should be stored and encrypted correctly. We cannot find the username or password in their original form.

A. Encrypting Schema 008, User records information, S1
B. Sign in as an existing user. Add confidential information such as bank information to the database. Decrypt the |database_file|, access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. Bank information stored should be encrypted. We cannot find the bank information in its original form in the database.

A. Encrypting Schema 009, User records information, S1
B. Sign in as an existing user. Add bank information with special characters. Decrypt the |database_file|,

access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. Bank information should be stored and encrypted. We cannot find the bank information in its original form in the database.

A. Encrypting Schema 010, User records information, S1
B. Sign in as an existing user. Add bank information with malicious code to attack the database. Decrypt the |database_file|, access the sqlite file by executing |sqlite3 decrypted_database_file| from command line, look up all tables and their contents by executing |.tables| and |select * from table_name|.
C. Program should not fail under the attack. Bank information should be stored and encrypted. We cannot find the bank information in its original form in the database.

4. File System

A. Database store file 001, File System, S1
B. Delete .DS_store file in the folder, run the program.
C. .DS_store file should be created automatically when the program start.

A. Database store file 002, File system, S1
B.Run the program, create a new account, close program , login by the created account.
C.The data should be stored into the file system. The user can login successfully.

A. Database store file 003, File system, S1
B. Open .DS_store file, and modify it directly.
C. .DS_store file should not allowed to modify it directly or it should be encrypted, or can not be open as well.

A. Database store file 004, File system, S1
B. create a new account A, add some information, close program, create a new account B, query for all information.

C. B should has no information. All data should be stored corresponding to the user account.

A. Database store file 005, File system, S1

B. Open the program twice and use same login account to login and do some update and add operation.

C. Two program of the same user should be synchronized to the file system.

5. Access API

A. Driver 001, Driver Module, S1

B. Register, login, or perform operations of creating, updating, retrieving, or deleting a record.

C. The API should be able to call and work coherently with GUI through driver to perform the aforementioned operations.

A. Stub 001, Stub Module, S1

B. Create, update, retrieve, or delete a record.

C. The API should be able to call and work coherently with Database, Encryption Class, and File System through stubs to perform the aforementioned operations.

A. User Input 001, Database, S1

B. Insert data in any text field or press any button.

C. The API should have functionalities to handle all text fields and buttons found in the application. The API should call correct system components for executing the functionalities.

A. User Record Creation 001, Database, S1

B. Create a new record of username / password, or bank account information.

C. The API should have a functionality allowing the user to create a new record. The new record should then be able to be updated, retrieved, and deleted. (*This test case is repeated as the API needs to have the functionality for creating the user record.*)

A. User Record Update 001, Database, S1

B. Update a record of username / password, or bank account information.

C. The API should have a functionality allowing the user to update a record. The record should then have the updated information when it is retrieved in the future. (*This test case is repeated as the API needs to have the functionality for updating the user record.*)

A. User Record Retrieval / Reading 001, Database, S1

B. Retrieve or read a record of username / password, or bank account information.

C. The API should have a functionality allowing the user to retrieve or read a record. The record should display the newest information correctly. (*This test case is repeated as the API needs to have the functionality for retrieving or reading the user record.*)

A. User Record Deletion 001, Database, S2

B. Delete a record of username / password, or bank account information.

C. The API should have a functionality allowing the user to delete a record. The record should be deleted from the database, and cannot be retrieved in future. All other records related to the deleted record should have the affected field set to null, or be deleted if needed. (*This test case is repeated as the API needs to have the functionality for deleting the user record.*)

A. User Authentication 001, Graphical User Interface, S1

B. Enter correct username with an incorrect password.

C. The API should prompt an error stating that the inputted username and password do not match. (*This test case is repeated as the API needs to have the functionality for verifying user credentials.*)

A. User Authentication 002, Graphical User Interface, S1

B. Enter incorrect username with a correct password.

C. The API should prompt an error stating that the inputted username and password do not match. (*This test case is repeated as the API needs to have the functionality for verifying user*

credentials.)

A. Settings Change 001, Graphical User Interface, S1

B. Change existing master password through “Settings”.

C. The API should have a functionality to change master password. The user should be able to login his or her account using only the updated master password.

A. Username / Password Management 001, Graphical User Interface, S1

B. Create, update, retrieve, or delete a record of username / password.

C. The API should have the functionality to manage a record of the username / password. The user should be able to create, update, retrieve, and delete the username / password accordingly. The application should be able to run robustly after the changes. (*This test case is repeated as the API needs to have the functionality for managing records of username/password.*)

A. Bank Account Management 001, Graphical User Interface, S1

B. Create, update, retrieve, or delete a record of bank account information.

C. The API should have the functionality to manage a record of the bank account information. The user should be able to create, update, retrieve, and delete the bank account information accordingly. The application should be able to run robustly after the changes. (*This test case is repeated as the API needs to have the functionality for managing records of bank account information.*)

6. Non-Functional Requirements

A. Reliability 001, Database connection, S2

B. Set the |database_file| to be unreadable from terminal. Run the project file and try to sign in with an existing account.

C. The program should not crash and handle it properly. The user should not be able to sign in.

A. Reliability 002, Database connection, S2

B. Set the |database_file| to be unreadable from terminal. Run the project file and try to sign in with an

existing account. Reset the |database_file| to be readable. Run the project file and try to sign in with an existing account.
C. The program should not crash. User should not be able to sign in in the first time and be able to sign in in the second time.

A. Reliability 003, Database connection, S2
B. Run two programs at the same time. Sign in as an existing user to both programs.
C. The program should not crash and be able to synchronize information. (*This test case is repeated as reliability involves information synchronization.*)

A. Usability 001, Clipboard, S3
B. Sign in as an existing user. Single click on an existing record. “Ctrl/Command-V” in a text editor.
C. The record should be pasted correctly and be the same as the record.

A. Usability 002, Clipboard, S3
B. Sign in as an existing user. Double click on an existing record. “Ctrl/Command-V” in a text editor.
C. The record should be pasted correctly and be the same as the record.

A. Usability 003, Installation, S1
B. Double click on the executable file for the software installation.
C. The software should be opened/installed successfully.

A. Usability 004, Uninstallation, S2
B. Delete the executable file.
C. The software should be deleted/uninstalled successfully.

A. Usability 005, Installation Path, S1
B. Try to open/install the software at different folders.
C. The software should be opened/installed successfully.

A. Usability 006, Main features, S1

B. Open the software and sign in to the main page.

C. The main features like Add, Update and Remove etc. should be on the GUI's main page for the user to use the software easily.

A. Usability 007, Multiple windows, S2

B. Sign in using the correct username and password.

C. The login page should disappear and only left the main GUI page.

A. Usability 008, Multiple windows, S2

B. After login, choose a category and click on the add button.

C. The add GUI pops up and the main GUI page should not allow other main features' button to be clickable to avoid multiple windows.

A. Usability 009, Multiple windows, S2

B. After login, choose a category and click on the update button.

C. The update GUI pops up and the main GUI page should not allow other main features' button to be clickable to avoid multiple windows.

A. Usability 010, Multiple windows, S2

B. After login, choose a category and click on the remove button.

C. The remove GUI pops up and the main GUI page should not allow other main features' button to be clickable to avoid multiple windows.