CSE 361 Team 5

BudgetPal Test Plan

Kyle Crowder, Jacob Norton, Jon Glodowski, Harrison Hruby

1. Introduction

The purpose of this document is to describe the process of testing the application. To test our application, the system shall be put through a rigorous testing regime. The regime will entail multiple test cases - each of which will be designed to test a specific functionality of the application. All test cases will be traceable throughout the document in order to show which Requirements from the Requirements Specification are being tested at any one time. Each test case will be comprised of distinct steps that will apply inputs to the system, as well as an expected output based upon the aforementioned inputs.

2. Requirements

2.1. Requirements Specification 3.2.1.2.1
The user can create an account.

2.2. Requirements Specification 3.2.1.2.2
The user can login and the system can accept or deny login.

2.3. Requirements Specification 3.2.2.2.1
The user can define a budget amount and choose Daily, Weekly, Biweekly, Monthly options. The system adjusts based on the new budget.

2.4. Requirements Specification 3.2.3.2.1
The user can add an expense and the system will store it. The system will display the updated remainder of the budget on the landing page.

2.5. Requirements Specification 3.2.3.2.2
The user can select a past expense to view, edit, and delete the expense.

2.6. Requirements Specification 3.2.4.2.1
The user can create an Event. The event is stored by the system and can be accessed in list form with other events.

2.7. Requirements Specification 3.2.4.2.2
The user can edit an event and edits are saved.

2.8. Requirements Specification 3.2.5.2.1
Upon login, the user is directed to the landing page which displays the current remaining budget.

2.9. Requirements Specification 3.2.5.2.2
The landing page displays currently active events.

2.10. Requirements Specification 3.2.6.2.1
The user can access the menu and select any of the interfaces. The system will navigate to the selected interface.

2.11. Requirements Specification 3.3.1.1
The user's information can only be accessed from one terminal.

2.12. Requirements Specification 3.3.1.2
Any number of users can have an account but only one user can have access at a time.

2.13. Requirements Specification 3.3.1.3
The type of information entered by the user will be numerical, text, and selectors only.

2.14. Requirements Specification 3.5.0
The application uses touchscreen and standard input buttons buttons only.

2.15. Requirements Specification 3.6.2
The application can run at anytime. The user can restart the app by closing and reopening it.

2.16. Requirements Specification 3.6.3
Passwords are stored with the hash technique.

2.17. Requirements Specification 3.6.5
The app is only required to run on stock android.

3. Tests

3.1 Type of information

Purpose:

1. To show that only the specified input types are used in this application.

Setup:

1. Perform all other tests except 3.2
2. Take note of the types of information that are being used to complete the tests. (i.e. text, numbers, multioption selectors)

Input:

1. (Not Applicable)

Valid Output:

1. No other information was required to be input other than text, numerical, and selectors.

Tests

Requirement 2.13

3.2 Touchscreen and standard input buttons

Purpose:

1. To show that the only input methods used are the touchscreen and the standard android input buttons.

Setup:

1. Perform all other tests except 3.1
2. Take note of all input methods that are being used to complete the tests. (i.e. touch screen, physical keyboard, camera)

Input:

1. (Not Applicable)

Valid Output:

1. All tests are successful and only the touchscreen and standard input buttons are used to complete them.

Tests:

Requirement 2.14

3.3 Run Application

Purpose:

1. To show that the application can be run and restarted successfully

Setup:

Start android device

Input:

1. Tap the BudgetPal icon in the navigation drawer
2. Close the app
3. Tap the BudgetPal icon again

Valid Output:

BudgetPal opens and loads the login page

Tests:

Requirement 2.15

3.4 Create Account

Purpose:

1. The show that users able to create an account

Setup:

1. Launch BudgetPal

Inputs:

1. Tap New User Button
2. Enter “testuser” for username and “testuser” for password
3. Tap Create Button

Valid Output:

1. Application should now be at the landing page for the user

Tests:

Requirement 2.1

3.5 Accept or Deny Login

Purpose:

1. To show that users are only able to login using the correct username and password pair.

Setup:

1. Launch BudgetPal
2. Test 3.4 must have already been completed
3. Ensure another account exists on system

Case 1:

Input:

1. Enter any nonempty string that is not “testuser” for username and enter “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Fails

Case 2:

Input:

1. Enter “testuser” for username and enter any nonempty string that is not “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Fails

Case 3:

Input:

1. Enter any nonempty string that is not “testuser” for username and enter any nonempty string that is not “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Fails

Case 4:

Input:

1. Enter “testuser” for username and enter “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Successful

Case 5:

Input:

1. Enter “testuser” for username and enter any substring of “testuser” that is not “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Fails

Case 6:

Input:

1. Enter any substring of “testuser” that is not “testuser” for username and enter “testuser” for password
2. Tap Login Button

Valid Output:

1. Login Fails

Case 7:

Input:

1. Enter an empty string for username and for password
2. Tap Login Button

Valid Output

1. Login Fails

Case 8:

Input:

1. Enter a username other than testuser that already has an account and enter “testuser” for password.
2. Tap Login Button

Valid Output:

1. Login Fails

Case 9:

Input:

1. Enter “testuser” for username and enter the password of a different account for password.
2. Tap Login Button

Valid Output:

1. Login Fails

Tests:

Requirements 2.2

3.6 Define Budget

Purpose:

1. To show that the user can create a budget of a specified amount for a specified period of time.

Setup:

1. Login to testuser
2. Navigate to Change Budget page

Input:

Case 1:

1. Attempt to set the budget to “hello”

Case 2:

1. Set the value of the budget to 12.00
2. Don’t select a period

Case 3:

1. Set the value of the budget to 12.00
2. Select the Daily period
3. Tap the Save button

Valid Output:

Case 1:

1. The input should be rejected.

Case 2:

1. The Save button should not be enabled

Case 3:

1. The Landing page should be displayed with the new budget displayed.

Tests:

Requirements 2.3

3.7 Add expense

Purpose:

1. To show that a user can create an expense and the system will update the remain budget to reflect the additional expense.

Setup:

1. Launch BudgetPal

Inputs:

1. Navigate to Add Expense
3. Enter 1.00
4. Choose Category “Payment”
5. Add Description “Test”
6. Tap Add button

Valid Output:

1. The remaining budget should be updated to reflect the new expense.

Tests:

Requirement 2.4

3.8 View, Edit, Delete expense

Purpose:

1. To show that users are able to view previously added expenses, edit them, and delete them if desired.

Setup:

1. Launch BudgetPal
2. Login to testuser
3. Create an Expense with the amount 6.00, category Grocery, description "My Expense"

Inputs:

Case 1:

1. Navigate to View Expenses
2. Tap the "My Expense" expense.

Case 2:

1. Navigate to View Expenses
2. Tap the "My Expenses" expense.
3. Change the amount for the expense to 7.00.
4. Press "Update Expense" button.

Case 3:

1. Navigate to View Expenses
2. Tap the "My Expenses" expense.
3. Press "Delete" Button.

Valid Output:

Case 1:

1. A list of expenses for the current budget period is displayed

Case 2:

1. The list of expenses should show the expense with an updated amount.

Case 3:

1. The list of expense should no longer dispalay the "My Expense" expense.

Tests:

Requirement 2.5

3.9 Create an Event

Purpose:

1. To show that users are able to added events to their account

Setup:

1. Launch BudgetPal
2. Login “testuser”

Inputs:

1. Navigate to Add Event
2. Enter Name - “Pay Bill”
3. Enter Date - “2030-04-04”
4. Tap Add

Valid Output:

1. New event should now appear under Upcoming Events

Tests:

Requirement 2.6

3.10 Edit an Event

Purpose:

1. To show that users are able to edit events that have been previously added to their account

Setup:

1. Launch BudgetPal
2. Login “testuser”

Inputs:

1. Navigate to Edit Event
2. Enter new Name: “bankruptcy”
3. Enter new Date: “2030-05-04”
4. Tap Add

Valid Output:

1. Edited event should now appear under View Events

Tests:

Requirement 2.7

3.11 Landing page displays current remaining budget and reset date

Purpose:

1. To show that the landing page displays the current remaining budget and reset date

Setup:

1. Launch BudgetPal
2. Login “testuser”

Inputs:

1. Change budget to “$11” per week
2. Logout
3. Login with the same account

Valid Output:

1. Verify that the budget is listed as being $11 per week on the landing page.
2. Verify that you are listed as have $11 until next week's date
3. If 5 and 6 are true then the test is passed

Tests:

Requirement 2.8

3.12 Landing Page displays currently active events.

Purpose:

1. To show that the events displayed under Upcoming events are only events that have yet to happen.

Setup:

1. Launch BudgetPal
2. Login
3. Add an Event with a date before the current date.
4. Add an Event with a date after the current date.

Inputs:

1. (Not Applicable)

Valid Output:

1. The before Event should not be displayed and the after Event should be displayed.

Tests:

Requirement 2.9

3.13 Navigation

Purpose:

1. To show that users are able to navigate between screens using the provided inteface.

Setup:

1. Launch BudgetPal
2. Login “testuser”

Input:

1. Open navigation drawer
2. Select a menu item
3. Verify it took you to the correct page
4. Tap the back button
5. Repeat 1 - 4 for all menu items
6. Repeat 1 -5 for all pages in the menu

Expected Output

1. All navigation actions will navigate to the appropriate page.

Tests:

Requirement 2.10

3.14 Single-Terminal Access

Purpose:

1. To show that the current implementation only allows for an account to be accessed on one device.

Setup:

1. Open BudgetPal on two devices with the app installed
2. Perform Test 3.4 on Device A.

Inputs:

1. On Device A, enter “testuser” for username and “testuser” for password
2. On Device A, tap “Login” button
3. On Device B, enter “testuser” for username and “testuser” for password
4. On Device B, tap “Login” button

Valid Output:

1. A successful login should occur on device A.
2. An unsuccessful login should occur on device B.

Tests:

Requirement 2.11

3.15 Multi-user capable / Single-user access

Purpose:

1. To show that multiple users can have an account simultaneously but only one user can be logged in at a time.

Setup:

1. Launch BudgetPal
2. Test 3.4 must be performed first
3. An account “testusertwo” must also be created

Input:

1. Login to “testuser”
2. Navigate to View Events and note the contents
3. Logout
4. Login to “tesusertwo”
5. Navigate to View Events and note the contents

Valid Output:

1. If neither events view contains the other’s events (unless the event has been added to both accounts) the test is passed.

Tests:

Requirement 2.12

3.16 Passwords are stored with hash technique

Purpose:

1. To show that passwords are not being stored in plaintext and being hashed properly.

Setup:

1. Test 3.4 must be performed first.
2. Open a command line terminal
3. Use the command `adb devices` to find the name of the device
4. Use the command `adb -s <device name> shell` to gain access the the devices shell
5. Use the command `cd /data/data/com.example.aegis.budgpal/databases` to navigate to the database file
6. Open the sqlite database with the command `sqlite3 userDB.db`
7. Use the command `SELECT * From User WHERE Username = ‘testuser’;`

Valid Output:

1. The contents of the HashedPassword column should not contain the text “testuser” and instead contain the output of a hash function.

Tests:

Requirement 2.16

3.17 Stock Android Requirement

Purpose:

1. To show that the application will run on a stock android device.

Setup:

1. Install apk onto a stock android device

Input:

1. Tap BudgetPal icon

Valid Output:

1. BudgetPal starts

Test

Requirements 2.17

4. Cleanup

1. testuser and testusertwo must be deleted in order to allow the next round of tests to go correctly.

5. Report

1. Test 3.1

    a. Result: Passed

2. Test 3.2

    a. Result: Passed

3. Test 3.3

    a. Result: Passed

4. Test 3.4

    a. Result: Passed

5. Test 3.5

    Case 1: Passed
    Case 2: Passed
    Case 3: Passed
    Case 4: Passed
    Case 5: Passed
    Case 6: Passed
    Case 7: Passed
    Case 8: Passed
    Case 9: Passed

6. Test 3.6

    Case 1: Passed
    Case 2: Passed
    Case 3: Passed

7. Test 3.7

    Result: Passed

8. Test 3.8

    Case 1: Passed
    Case 2: Passed
    Case 3: Failed (Expense is still listed)

9. Test 3.9

    a. Result: Passed

10. Test 3.10
    a. Result: Passed

11. Test 3.11
    a. Result: Passed

12. Test 3.12
    a. Result: Passed

13. Test 3.13
    a. Result: Passed

14. Test 3.14
    a. Result: Passed

15. Test 3.15
    a. Result: Passed

16. Test 3.16
    a. Result: Passed

17. Test 3.17
    a. Result: Passed