Design – MainActivity, InsertExpense & UpdateDeleteActivity

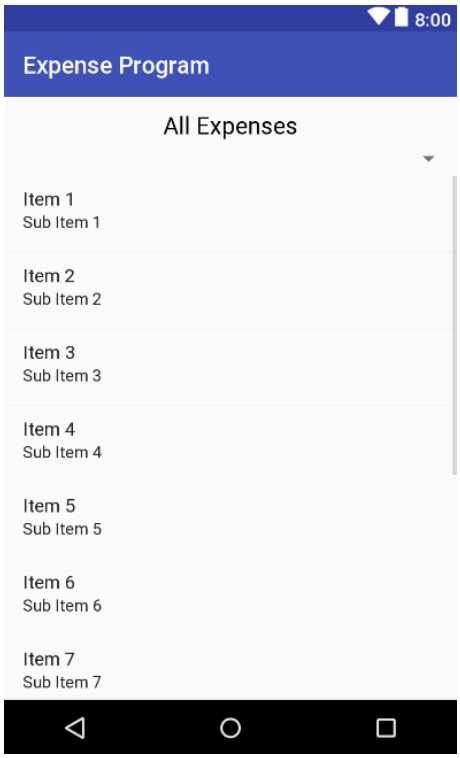
MainActivity.Java, Activity_get_all_expenses.xml. & list_item.xml

Activity_get_all_expenses.xml is the main activity shown at startup. It displays all the data obtained from the SQL database using a listview for each expense. Each expense holds multiple views for each field within the expense record. Expenses are displayed by using the DatabaseHelper.Java class which is used to retrieve the data.

The data retrieved by the DatabaseHelper is fed to ExpenseCustomAdapter.Java class which is used to manipulate the data given and modify the listview by creating a separate list_item (list_item.xml) for each expense record found in the database. It does this by using the getters and setters found in the ExpenseModel.Java class.

The data retrieved from the database can also be filtered by 'Show All', 'Expense Claimed' and 'Expense Unclaimed' by using a spinner dropdown list where each item (filter) populates the expense list depending on the filter used. It does this by using a specific SQL query for each filter.

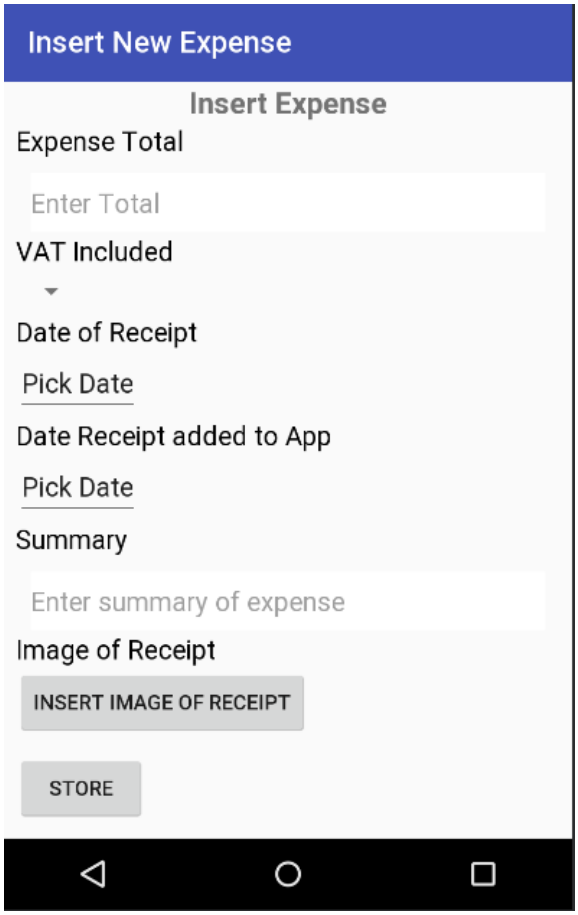
Activity_get_all_expenses.xml & list_item.xml



InsertExpense.Java & activity_insert_expenses.xml

InsertExpense.Java is used to insert new expenses to the database. All input values are validated to prevent any incorrect data from entering the database. The user is not allowed to claim an expense here as it must be claimed later via UpdateDeleteActivity.Java. Once an expense has been stored, the user will be taken back to the MainActivity (Activity_get_all_expenses.xml) where the expense inserted can then be viewed.

activity_insert_expenses.xml

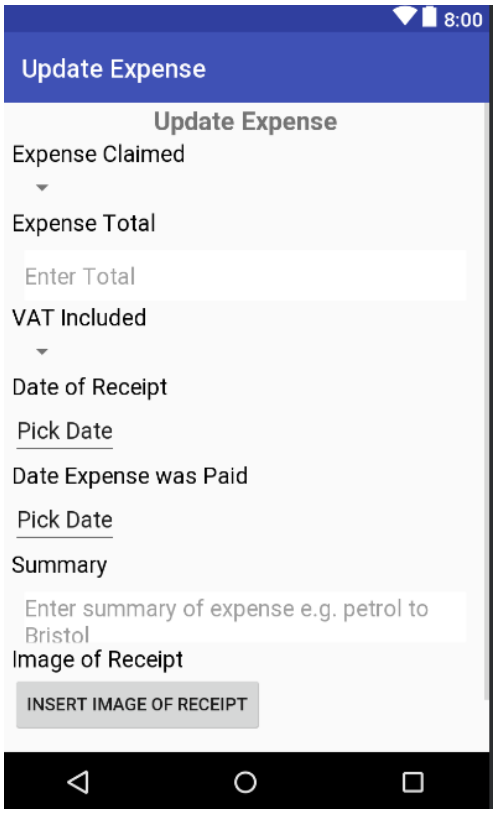


UpdateDeleteActivity.Java & activity_update_delete.xml

UpdateDeleteActivity.Java is used to update existing expenses from the database. This is the only way an expense can be set as claimed (as you can only do it later). It can also be used to update any incorrect data e.g. a mistake on the date of receipt or forgetting to check the VAT included as true. The class can also be used to delete the specified expense (record) for the database using the 'Delete' button.

All input values are validated to prevent any incorrect data from entering the database. Once an expense has been stored, the user will be taken back to the MainActivity (Activity_get_all_expenses.xml) where the expense updated can then be viewed (if not deleted).

activity_update_delete.xml

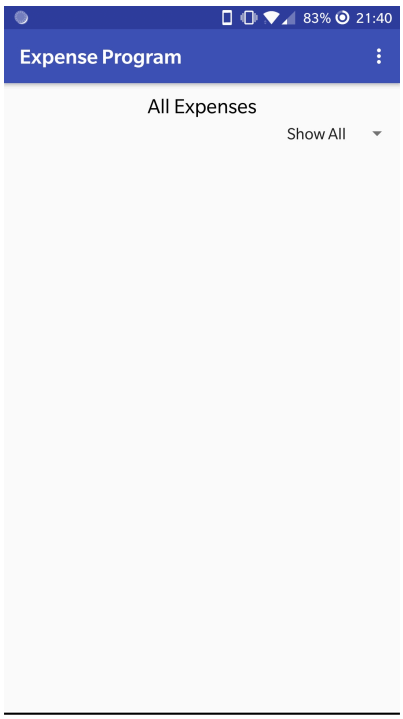


Testing

Test Plan 1 – Starting the Program/Initial State

Testing Plan	Solution
Purpose of the Test	See if the initial state of the program boots correctly and matches the design specification
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Empty State as it is the first boot of the program. Database is empty therefore no expenses will be retrieved and displayed on the activity
Expected Post-Test Conditions or Outcomes	Program should boot to the main activity showing an empty list
Verdict	Program booted without any issues and displays an empty list

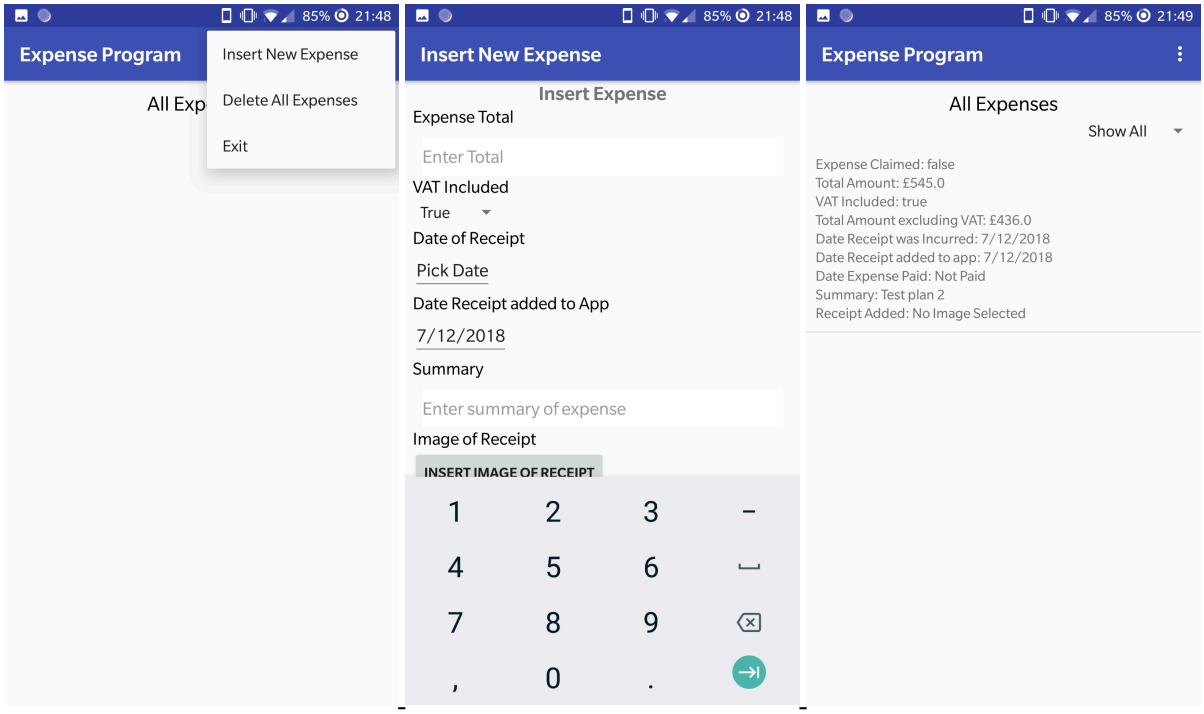
Screenshots/Evidence



Test Plan 2 – Inserting an Expense

Testing Plan	Solution
Purpose of the Test	Insert an expense into the Expense Manager via the Insert Expense spinner item using a menu
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list is empty; this will be the first expense inserted into the database and displayed to the screen
Expected Post-Test Conditions or Outcomes	A new expense should be inserted to the database and displayed on the screen
Verdict	Insert Expense activity is created successfully via the Insert Expense method found on the menu item spinner

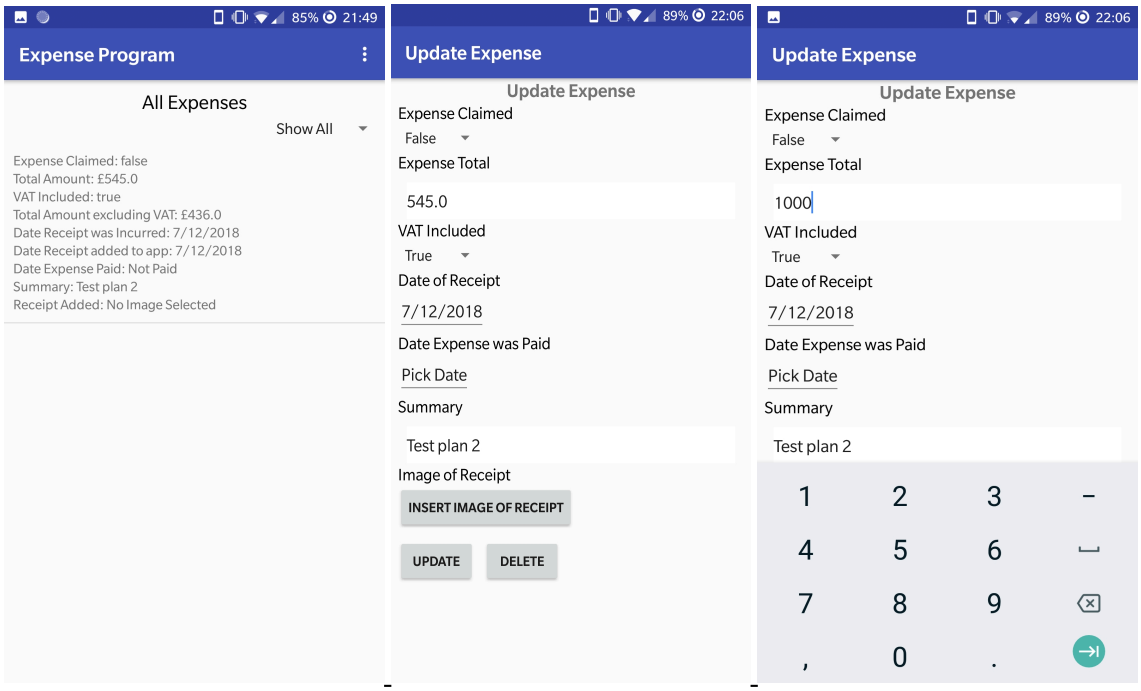
Screenshots/Evidence

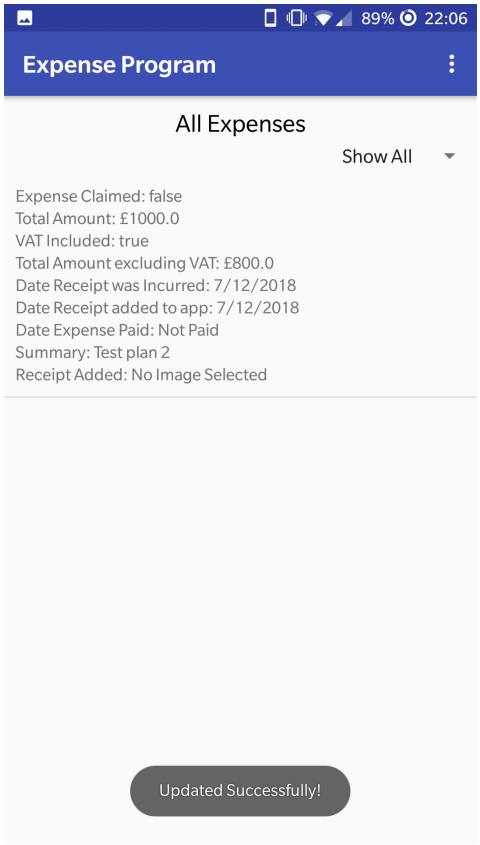


Test Plan 3 – Updating an Expense

Testing Plan	Solution
Purpose of the Test	Update an expense using an onViewClick listener to launch a new activity named Update Expense.
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 1 expense
Expected Post-Test Conditions or Outcomes	The expense should be updated sending the new data to the database and also be displayed on the screen
Verdict	Expense is updated correctly changing the 'expense total' from 545 to 1000 also displaying a toast message to notify the user it updated successfully.

Screenshots/Evidence

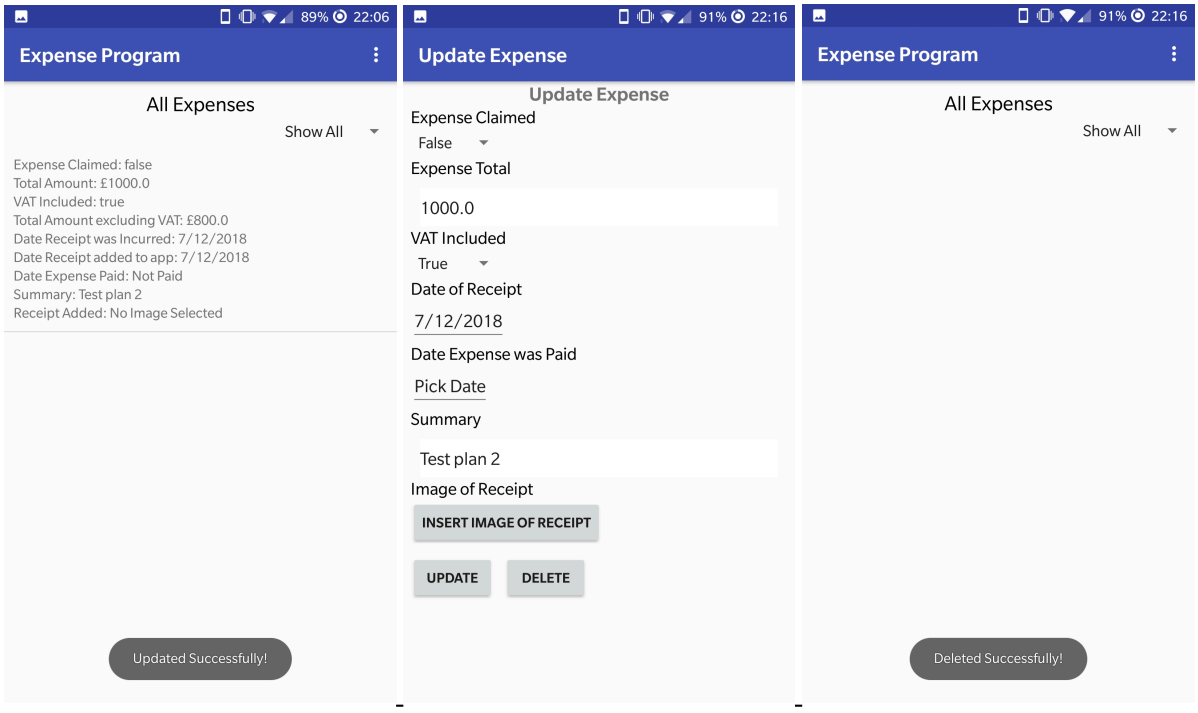



89% 22:06
Expense Program
All Expenses
Show All
Expense Claimed: false
Total Amount: £1000.0
VAT Included: true
Total Amount excluding VAT: £800.0
Date Receipt was Incurred: 7/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: Not Paid
Summary: Test plan 2
Receipt Added: No Image Selected
Updated Successfully!

Test Plan 4 – Deleting an Expense

Testing Plan	Solution
Purpose of the Test	Delete an expense using an onViewClick listener to launch a new activity named Update Expense.
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 1 expense
Expected Post-Test Conditions or Outcomes	The expense should be deleted from the database
Verdict	Expense is deleted successfully from the database and is no longer displayed on the Expense Manager list also displaying a toast message notifying the user

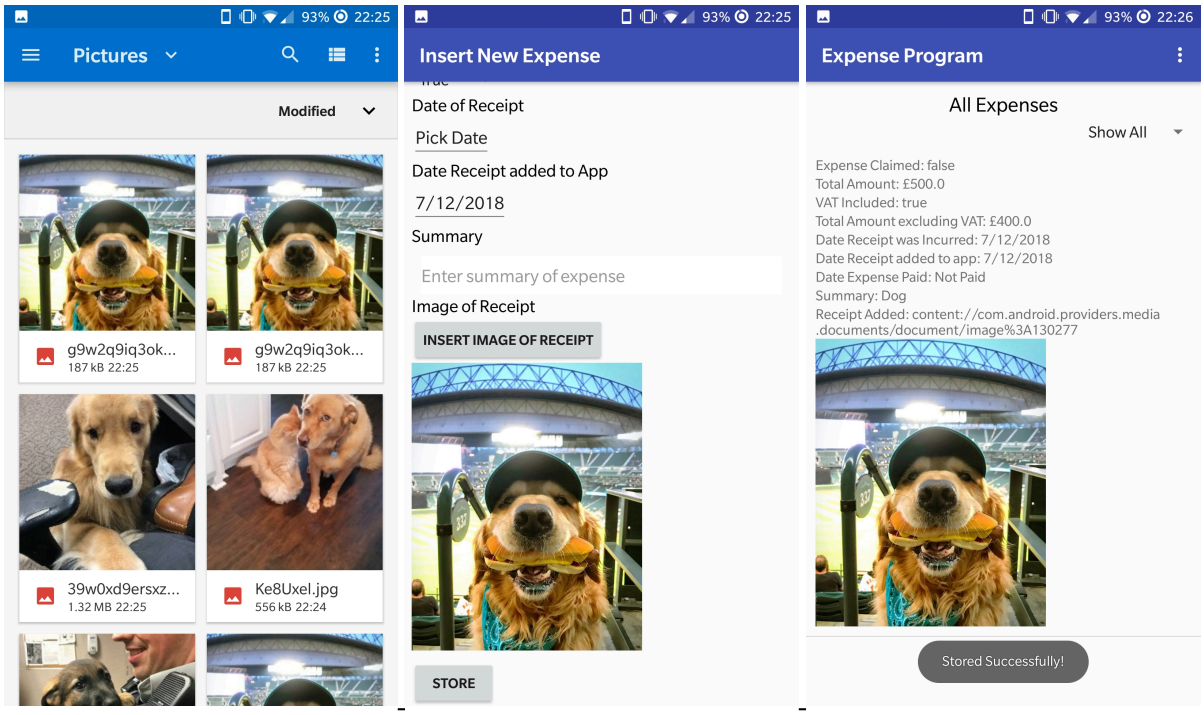
Screenshots/Evidence



Test Plan 5 – Inserting an Image

Testing Plan	Solution
Purpose of the Test	Test if image is inserted and retrieved correctly using URI data to hold the data necessary to display the image
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has no expenses
Expected Post-Test Conditions or Outcomes	The image should be inserted, displayed and retrieved correctly from the database using URI data to hold the data necessary to store and display the image
Verdict	Image is displayed and inserted to the database successfully.

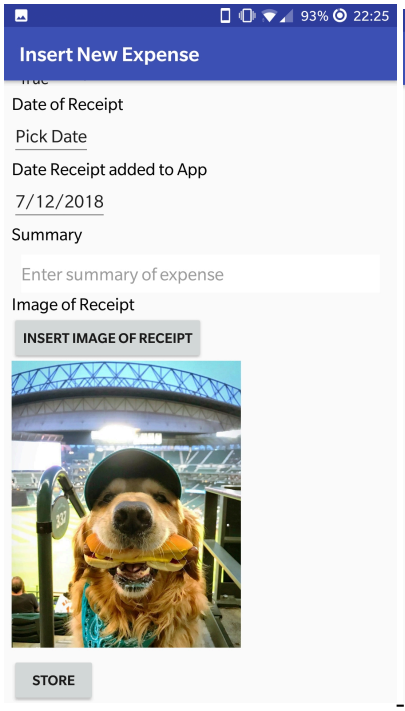
Screenshots/Evidence

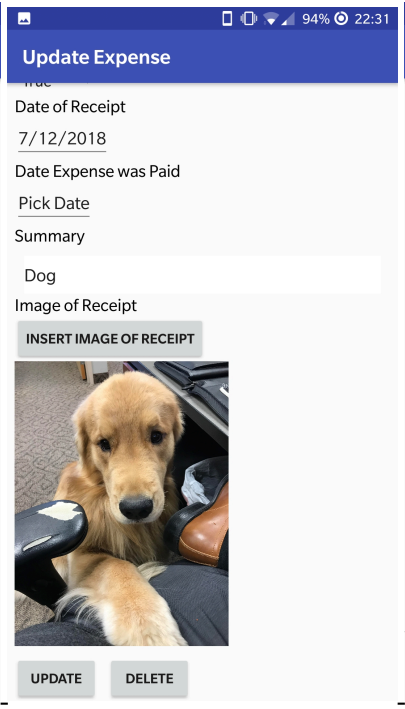


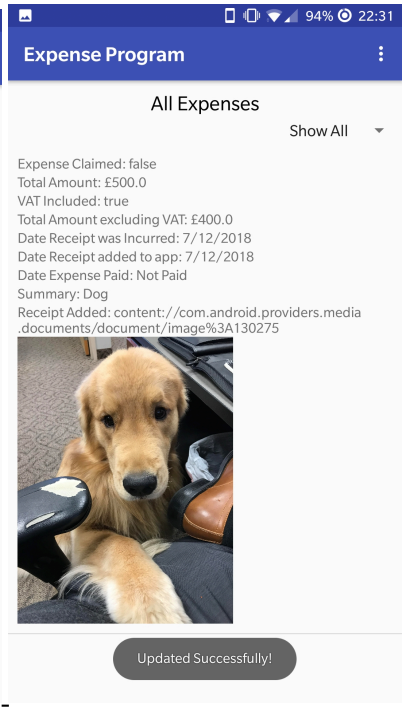
Test Plan 5 – Updating an Image

Testing Plan	Solution
Purpose of the Test	Test if image is updated and retrieved correctly using URI data to hold the data necessary to display the image
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 1 expense with an image
Expected Post-Test Conditions or Outcomes	The image should be updated, displayed and retrieved correctly from the database using URI data to hold the data necessary to store and display the image
Verdict	Image is displayed and updated to the database successfully.

Screenshots/Evidence



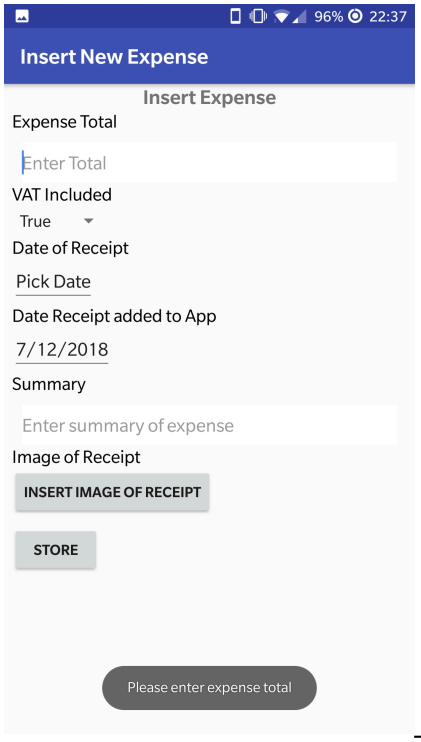


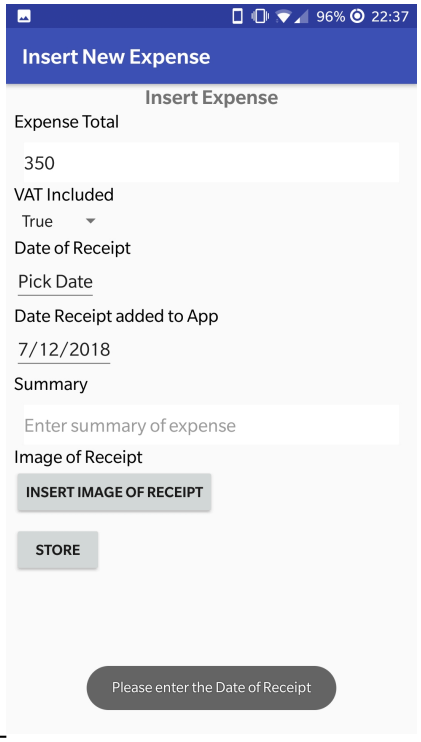


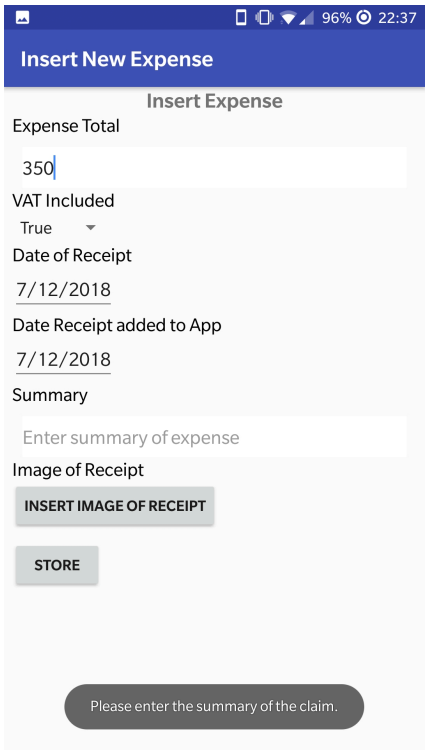
Test Plan 6 – Testing Verification of Input Validation on Insert Expense

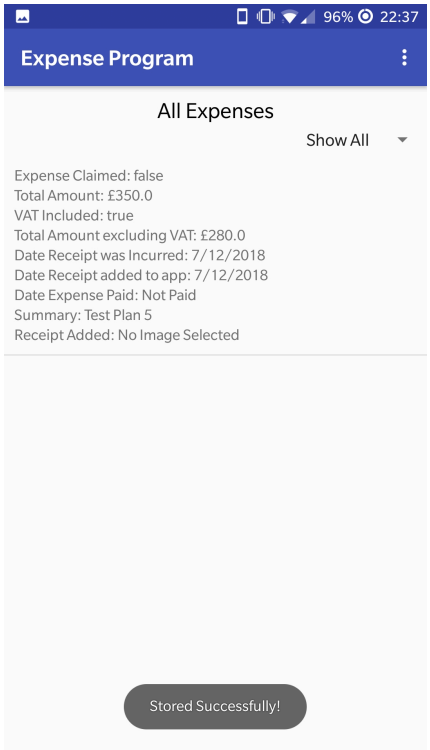
Testing Plan	Solution
Purpose of the Test	Test if all Views on the Insert Expense Activity with the ability to input values are correctly validated. The date of the expense added to the program should also not be clickable and only insert the current date.
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list is empty
Expected Post-Test Conditions or Outcomes	The user should be notified for each input validation that is required via a toast message
Verdict	Input validation is successful. All toasts displayed correctly when required. Date of expense added is not clickable and inserts the current date as its value.

Screenshots/Evidence


96% 22:37
Insert New Expense
Insert Expense
Expense Total
Enter Total
VAT Included
True
Date of Receipt
Pick Date
Date Receipt added to App
7/12/2018
Summary
Enter summary of expense
Image of Receipt
INSERT IMAGE OF RECEIPT
STORE
Please enter expense total


96% 22:37
Insert New Expense
Insert Expense
Expense Total
350
VAT Included
True
Date of Receipt
Pick Date
Date Receipt added to App
7/12/2018
Summary
Enter summary of expense
Image of Receipt
INSERT IMAGE OF RECEIPT
STORE
Please enter the Date of Receipt

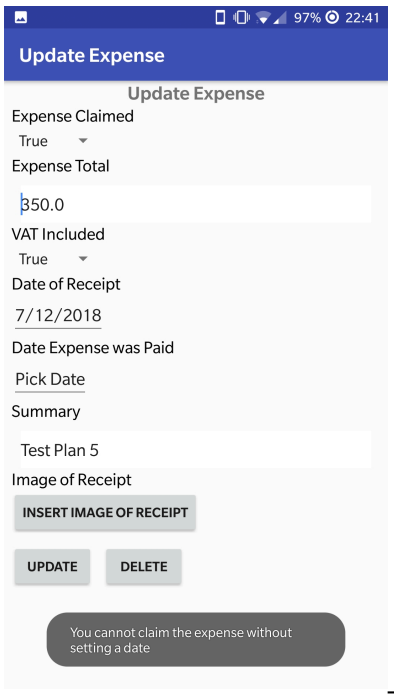

96% 22:37
Insert New Expense
Insert Expense
Expense Total
350
VAT Included
True
Date of Receipt
7/12/2018
Date Receipt added to App
7/12/2018
Summary
Enter summary of expense
Image of Receipt
INSERT IMAGE OF RECEIPT
STORE
Please enter the summary of the claim.

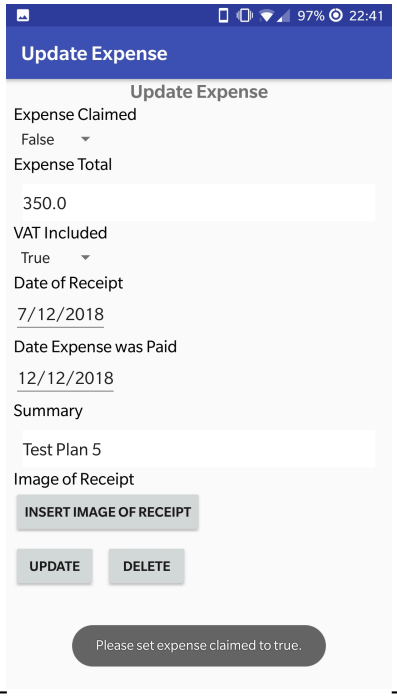

96% 22:37
Expense Program
All Expenses
Show All
Expense Claimed: false
Total Amount: £350.0
VAT Included: true
Total Amount excluding VAT: £280.0
Date Receipt was Incurred: 7/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: Not Paid
Summary: Test Plan 5
Receipt Added: No Image Selected
Stored Successfully!

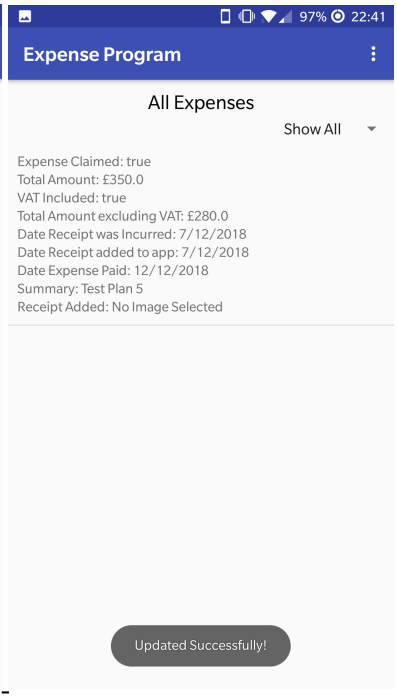
Test Plan 7 – Testing Verification of Input Validation on Update Expense

Testing Plan	Solution
Purpose of the Test	Test if all Views on the Update Expense Activity with the ability to input values are correctly validated
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 1 expense
Expected Post-Test Conditions or Outcomes	The user should be notified for each input validation that is required via a toast message
Verdict	Input validation is successful. All toasts displayed correctly when required.

Screenshots/Evidence



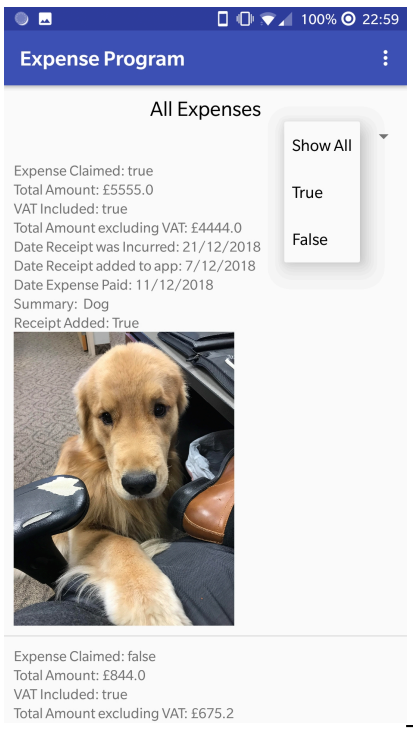


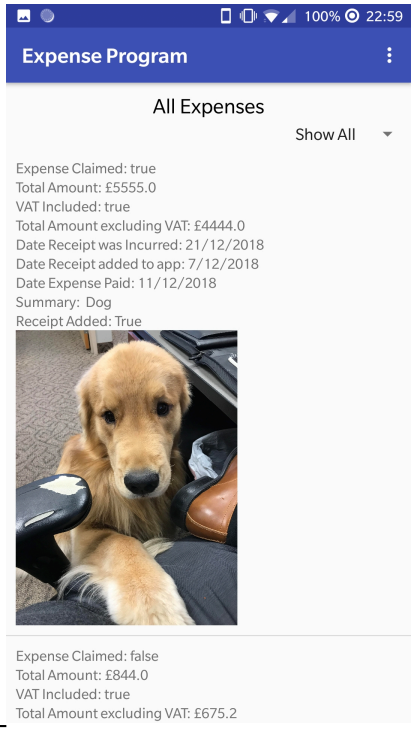


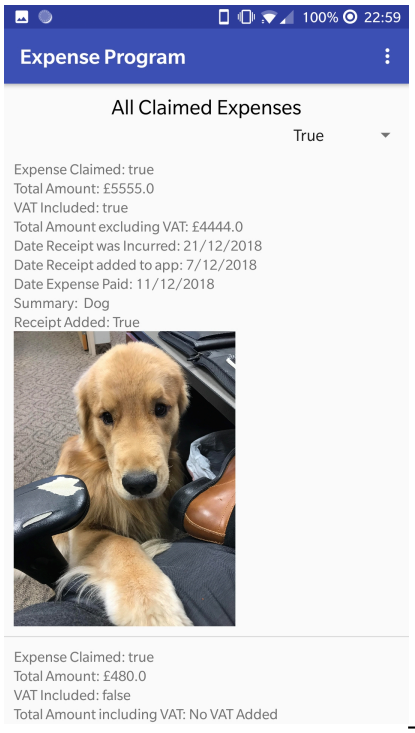
Test Plan 8 – Testing Filter for Expenses ('Show All', 'Claimed' & 'Not Claimed')

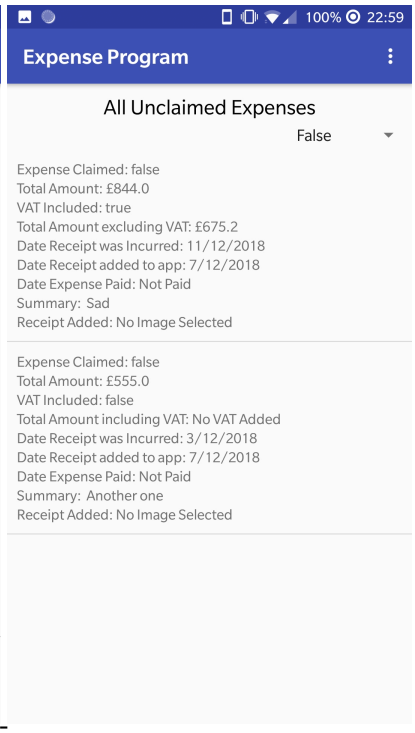
Testing Plan	Solution
Purpose of the Test	Test if the spinner dropdown item list ('Show All', 'Claimed' & 'Not Claimed') each display the specified expenses from the database for each filter
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 4 expenses, 2 claimed, 2 not claimed.
Expected Post-Test Conditions or Outcomes	The user should be able to switch between each filter using the spinner dropdown to display the specified expenses depending on selection
Verdict	Spinner selection works successfully. The specified expenses are shown when changing filters.

Screenshots/Evidence


100% 22:59
Expense Program
All Expenses
Show All
True
False
Expense Claimed: true
Total Amount: £5555.0
VAT Included: true
Total Amount excluding VAT: £4444.0
Date Receipt was Incurred: 21/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: 11/12/2018
Summary: Dog
Receipt Added: True
Expense Claimed: false
Total Amount: £844.0
VAT Included: true
Total Amount excluding VAT: £675.2


100% 22:59
Expense Program
All Expenses
Show All
Expense Claimed: true
Total Amount: £5555.0
VAT Included: true
Total Amount excluding VAT: £4444.0
Date Receipt was Incurred: 21/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: 11/12/2018
Summary: Dog
Receipt Added: True
Expense Claimed: false
Total Amount: £844.0
VAT Included: true
Total Amount excluding VAT: £675.2


100% 22:59
Expense Program
All Claimed Expenses
True
Expense Claimed: true
Total Amount: £5555.0
VAT Included: true
Total Amount excluding VAT: £4444.0
Date Receipt was Incurred: 21/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: 11/12/2018
Summary: Dog
Receipt Added: True
Expense Claimed: true
Total Amount: £480.0
VAT Included: false
Total Amount including VAT: No VAT Added


100% 22:59
Expense Program
All Unclaimed Expenses
False
Expense Claimed: false
Total Amount: £844.0
VAT Included: true
Total Amount excluding VAT: £675.2
Date Receipt was Incurred: 11/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: Not Paid
Summary: Sad
Receipt Added: No Image Selected
Expense Claimed: false
Total Amount: £555.0
VAT Included: false
Total Amount including VAT: No VAT Added
Date Receipt was Incurred: 3/12/2018
Date Receipt added to app: 7/12/2018
Date Expense Paid: Not Paid
Summary: Another one
Receipt Added: No Image Selected

Test Plan 9 – Deleting all Expenses from the Database

Testing Plan	Solution
Purpose of the Test	Test if the menu option 'Delete All Expenses' deletes all records from the database
Existing pre-test conditions prevailing prior to the test i.e. user interface and data	Expense list has 4 expenses
Expected Post-Test Conditions or Outcomes	The user should be able to select the 'Delete All Expenses' option from the dropdown menu and delete all records from the database leaving an empty list.
Verdict	Deletion of records from the database works successfully leaving the user with an empty list.

Screenshots/Evidence