CM3019 Programming Mobile Devices
Coursework
Expense Manager

Stanislav Hamara
1207466

16/04/15

1.0 Introduction

The purpose of this documentation is to describe and evaluate and Android application that serves as an expense manager as assigned for our CM3019 Coursework. In this documentation I will describe and explain my choice of elements in the User Interface, design and relationship of the classes and the database that has been used.

2.0 Statement of compliance

All the requirements were fulfilled up to the specifications as required. There are minor bugs that will be described further in this document.

3.0 Software Design

In this section I will discuss and describe my design choices from the software development point of view. The class diagram is included (Figure 1) to show the relations between classes and each class will be described separately.

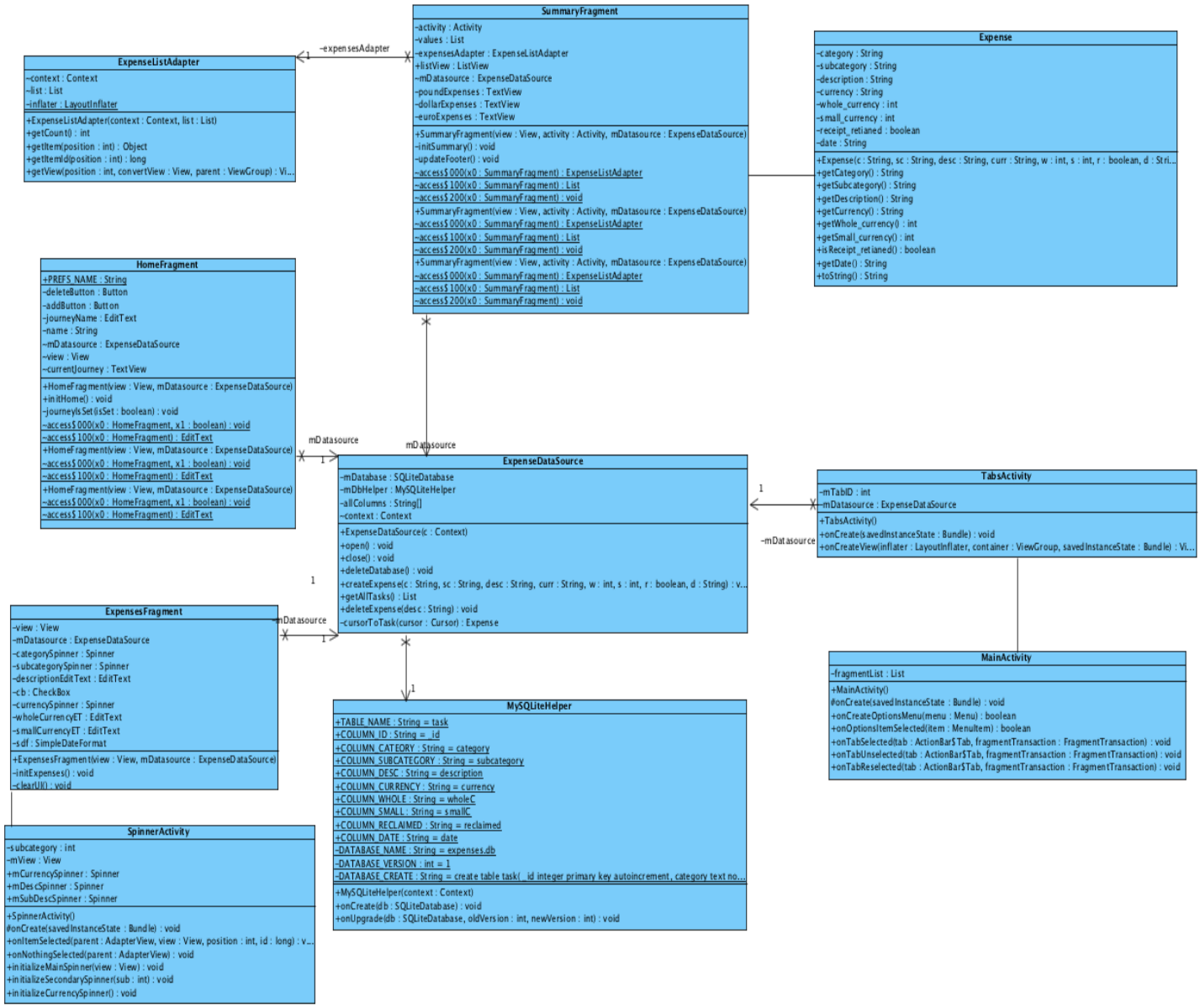

Figure 1

3.1 MainActivity

Main Activity that extends ActionBarActivty is a class that initializes and defines an Action Bar that is used to navigate between different tabs in the application. This class creates a Fragment and based on which tab is selected by the user, assigns a relevant layout to it.

3.2 TabFragment

TabFragment defines and displays the UI, based on which tab was selected by the user. Every layout has a Controller assigned to it. This controller implements all the logic and functionality for the particular layout.

TabFragment is also used for defining a data source for the application. This is where the database is created.

3.3 HomeController

HomeController takes care of deleting and adding a journey. User is introduced with a screen that allows the user to create or delete a journey. There can be only one journey created at the same time. Journey is stored in the phone storage via SharedPreferences. If the user tries to delete the journey, they are advised that there still might be some expenses unclaimed.

3.4 ExpenseController

ExpenseController defines the functionality for adding expenses. It controls the fragment that contains multiple UI elements, allowing the user to sufficiently describe the expense that needs to be added to the database. Value control is also implemented in this class.

3.5 SpinnerActivity

SpinnerActivity is linked to the ExpenseController because it controls the content of spinners in the Expense View. Based on the choice in the first spinner, second one is populated by the relevant options pulled from an array.

3.6 SummaryController

This class contains a ListView that pulls the information from the database and displays all the expenses and details that have been added by the user. User can reclaim the expense by holding their finger on the item in the list that they want to reclaim. SummaryController also calculates the total expenses in every currency separately.

3.7 ExpenseListAdapter

This class is an adapter for the ListView in the Summary View. It defines what information are taken from the database and shown in the list for each item.

3.8 Expense

Expense defines an expense object, containing all the necessary information that need to be stored in the database.

3.9 SQLiteHelper

This class is defined to help the developer to build the initial queries in a readable way. It builds the query for creating a table in the database.

3.10 ExpenseDataSource

ExpenseDataSource is used to link the application with the database; it defines insertion and deletion of the data in the database. It also extracts the data from the database and constructs Expense objects with it.

For the improved readability and the ease of use I have downloaded Sqlitebrowser(http://sqlitebrowser.org/), which allowed me to show the database at all time and made it easier to track the errors. It was especially useful during the testing. I have designed this application using only one table, as it was not necessary to store a huge amount of data and I believe it is quite efficient (Figure 2)

	_id	category	subcategory	description	currency	wholeC	smallC	reclaimed	date
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Meals	Breakfast	steak	£	43	21	0	17:11:24 20/04/2015
2	2	Transport	Taxi	to hotel	€	34	32	0	17:11:54 20/04/2015

Figure 2

4.0 User Interface

This application was designed with a very simple user interface that consists only from three different views. Everything is controlled via and Action bar on the top that allows the user to pick which tab they want to view.

Bug: When the user launches the application it starts on the home screen. When the user switches to expenses followed by summary, everything works fine.

On the other hand if the user goes directly to the summary screen and then returns back to expenses, it will still show the summary view.

Home->Expenses->Summary – **works**
Home->Summary->Expenses – **still shows Summary screen even in the Expenses tab**

I have spend few hours trying to fix this problem by testing everything I could think of, but unfortunately I was not able to come up with the solution.

The implemented views are:

- Home
- Expenses
- Summary

Home tab allows the user to add and delete the journey (Figure 3). It disables certain elements based on if the journey is created or not.

If the user tried to create a journey without a name, an alert will be displayed informing the user that the name cannot be an empty string (Figure 4).

If the user tries to delete a journey, an alert will be displayed, informing the user that some expenses might not be claimed yet (Figure 5).

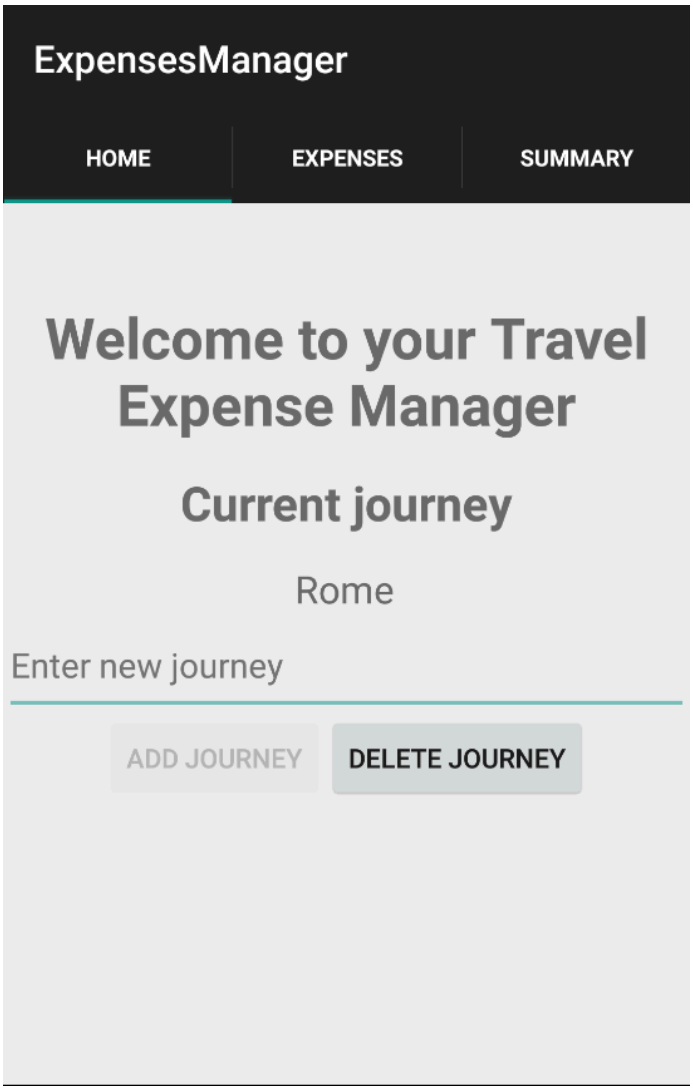

Figure 3

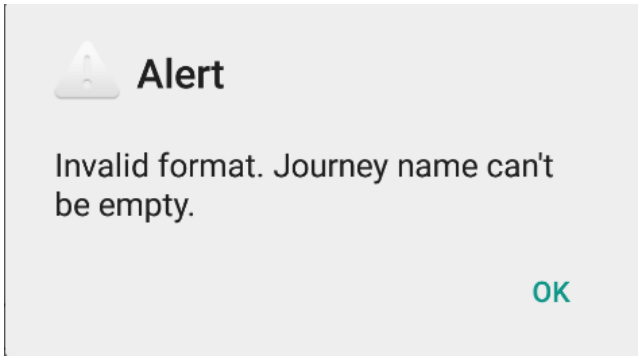

Figure 4

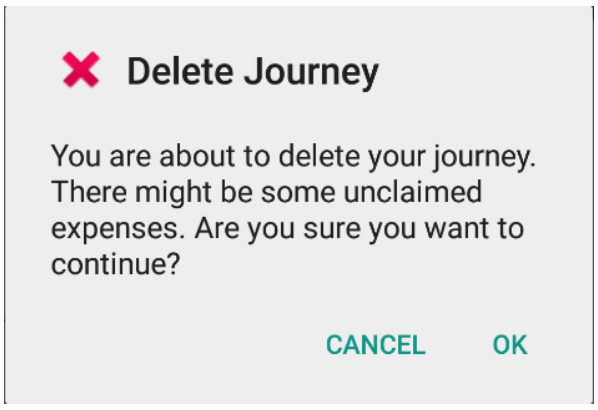

Figure 5

Expenses tab allows the user to add expenses to their list. The user is shown an UI that allows him to pick the category and subcategory of the expense from the dropdown menu. Afterwards the user inputs the description of the currency, selects the currency and inputs the amount of money spent on that expense. User can also tick a checkbox if they received a receipt from that expense (Figure 6)

The value checking has been implemented too and if the user tries to submit an invalid value, they will be prompted with an alert (Figure 7).

If the user tries to add an expense without creating a journey first, they will be prompted with an alert (Figure 8)

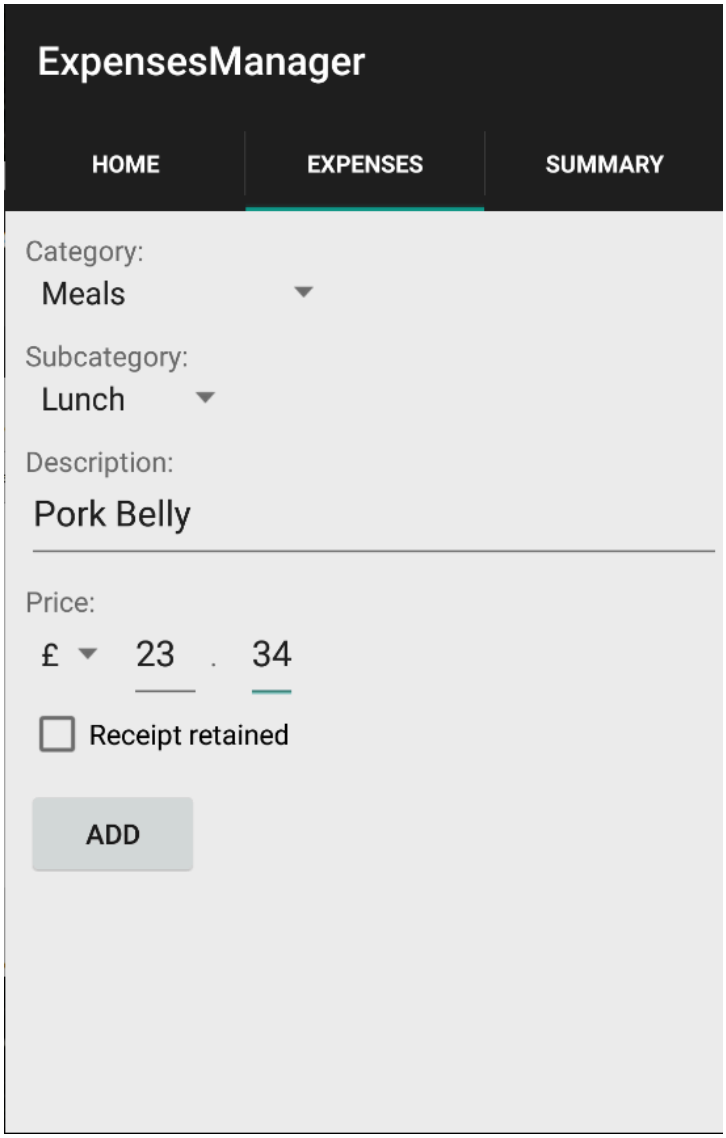

Figure 6

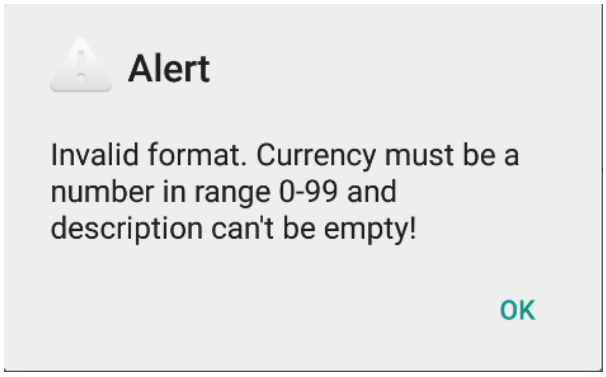

Figure 7

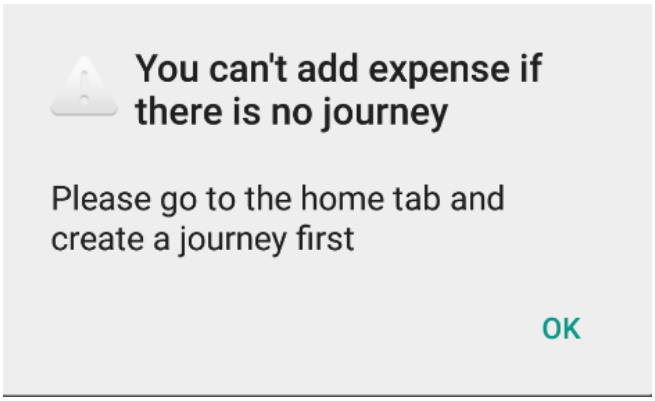

Figure 8

The summary view consists of 2 parts. List of the expenses added by the user with its detailed description and an expense summary on the bottom, which shows the user how much money, was spent in each particular currency. (Figure 9) If the user reclaimed an expense and wants to delete it from the list, the item is deleted by "long-clicking" on it. The user then needs to confirm their decision to delete the expense (Figure 10).

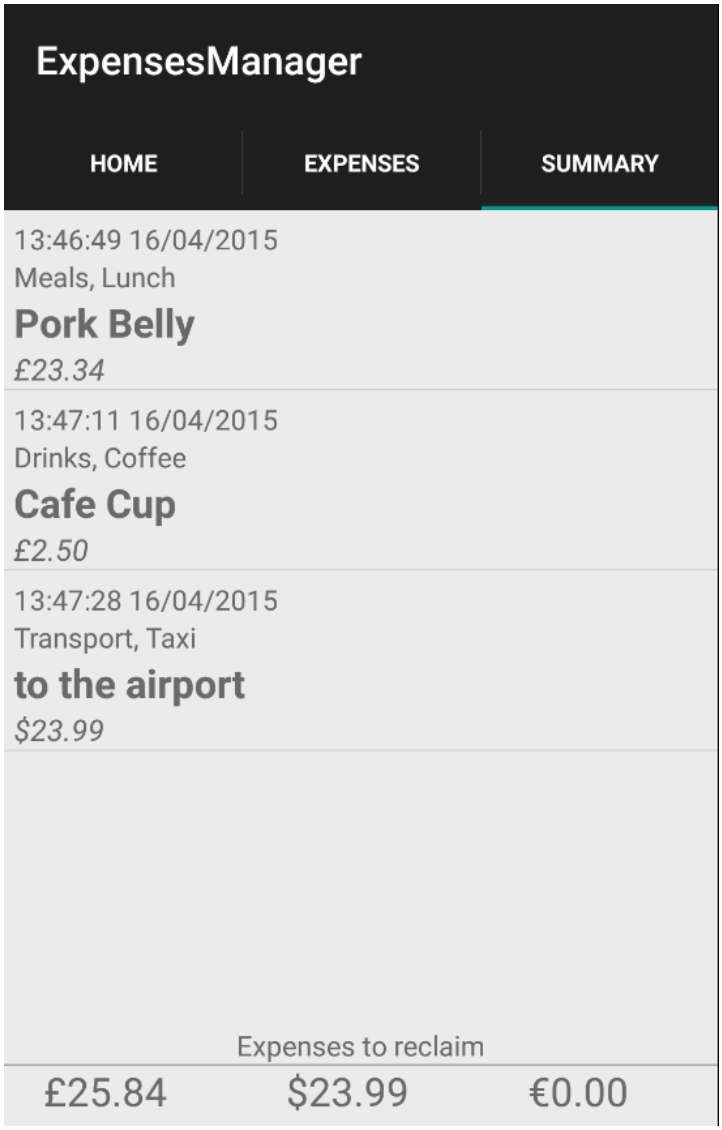

Figure 9

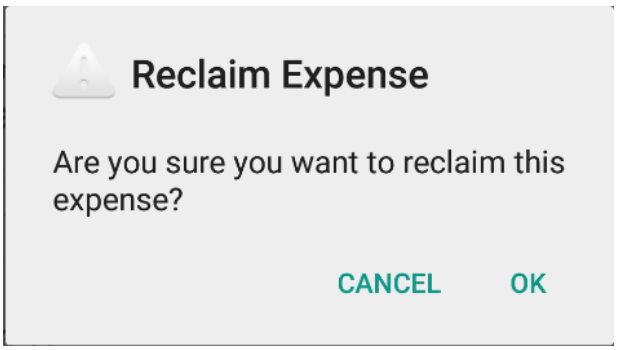

Figure 10

5.0 Testing

Test#	Description	Result expected	Result
AC01	Launch	Application launches and presents the user with the home screen	Pass
AC02	Creation of a journey	Journey is created after user inputs the name of the journey and presses the "ADD JOURNEY" button	Pass
AC03	Deletion of the journey	Journey is deleted after user presses the "DELETE JOURNEY" button and confirms the deletion. The database content is deleted too	Pass
AC04	Journey name check	The user is prompt with an alert if they try to add a journey without giving it a name	Pass
AC05	Navigation between tabs	Based on user's choice of the tab, a relevant layout is loaded	Pass
AC06	Expense subcategory spinner	The content of the subcategory spinner changes based on what was selected in the first spinner	Pass
AC07	Expense value check	The user is prompted with an alert if they try to submit an expense with invalid values	Pass
AC08	Expense addition	The expense is added to the database after user presses the "ADD" button	Pass
AC09	Summary list view	All the expenses are pulled from the database and displayed in the list in a form designed in the ListAdapter	Pass
AC10	Sum of expenses	Sum of expenses for each currency is calculated on the bottom of Summary screen	Pass
AC11	Long click reclaiming	The expense is deleted from the database and the list after user long clicks on the particular expense and confirms the deletion	Pass
AC12	Sum of expenses update	Sum of expenses for each currency is updated every time an expense is either added or deleted	Pass
AC13	Persistence of the data	All the data remains unchanged after the application is closed or restarted	Pass