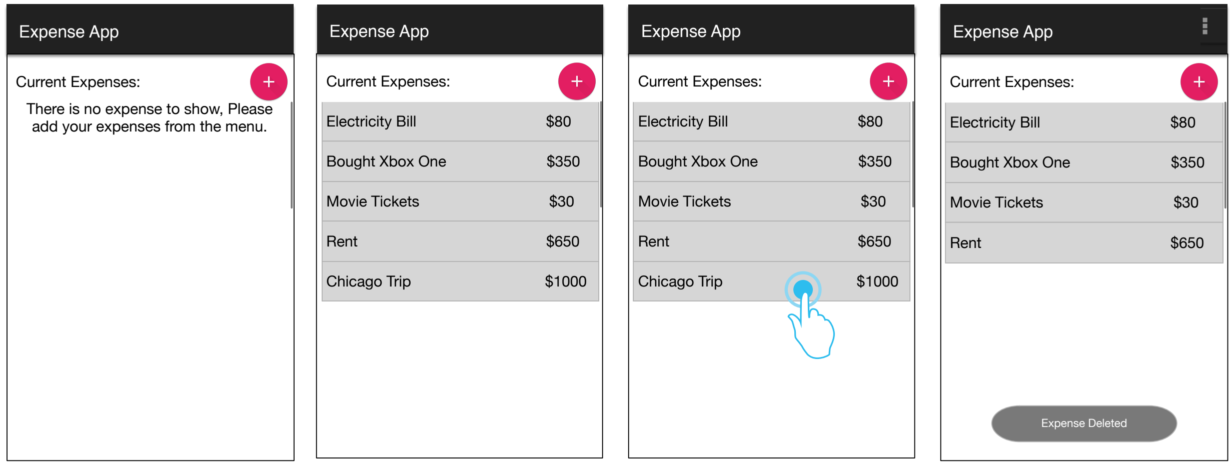
ITIS/ITCS 4180/5180 Mobile Application Development
In Class 09

Basic Instructions:

1. In every file submitted you MUST place the following comments:
   a. Assignment #.
   b. File Name.
   c. Full name of the student.
2. This is an individual assignment. Everyone must submit.
3. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will loose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
4. Export your Android project and create a zip file which includes all the project folder and any required libraries.
5. Submission details:
   a. Compress the contents of your project folder. The file name is very important and should follow the following format: **800#_InClass09.zip**
   b. Only one group member is required to submit on behalf of the whole group.
   c. You should submit the assignment through Canvas: Submit the zip file.
6. **Failure to follow the above instructions will result in point deductions.**

In Class Assignment 09 (100 points)

In this assignment you will get familiar with Firebase and implement an app to add and display expenses. You can use either Fragments or Activities to implement the application.



There is no expense to show, Please add your expenses from the menu.

Electricity Bill	$80
Bought Xbox One	$350
Movie Tickets	$30
Rent	$650
Chicago Trip	$1000

Electricity Bill	$80
Bought Xbox One	$350
Movie Tickets	$30
Rent	$650
Chicago Trip	$1000

Electricity Bill	$80
Bought Xbox One	$350
Movie Tickets	$30
Rent	$650

Expense Deleted

(a) No expenses in the list

(b) ListView with expenses

(c) Long Click to delete expense

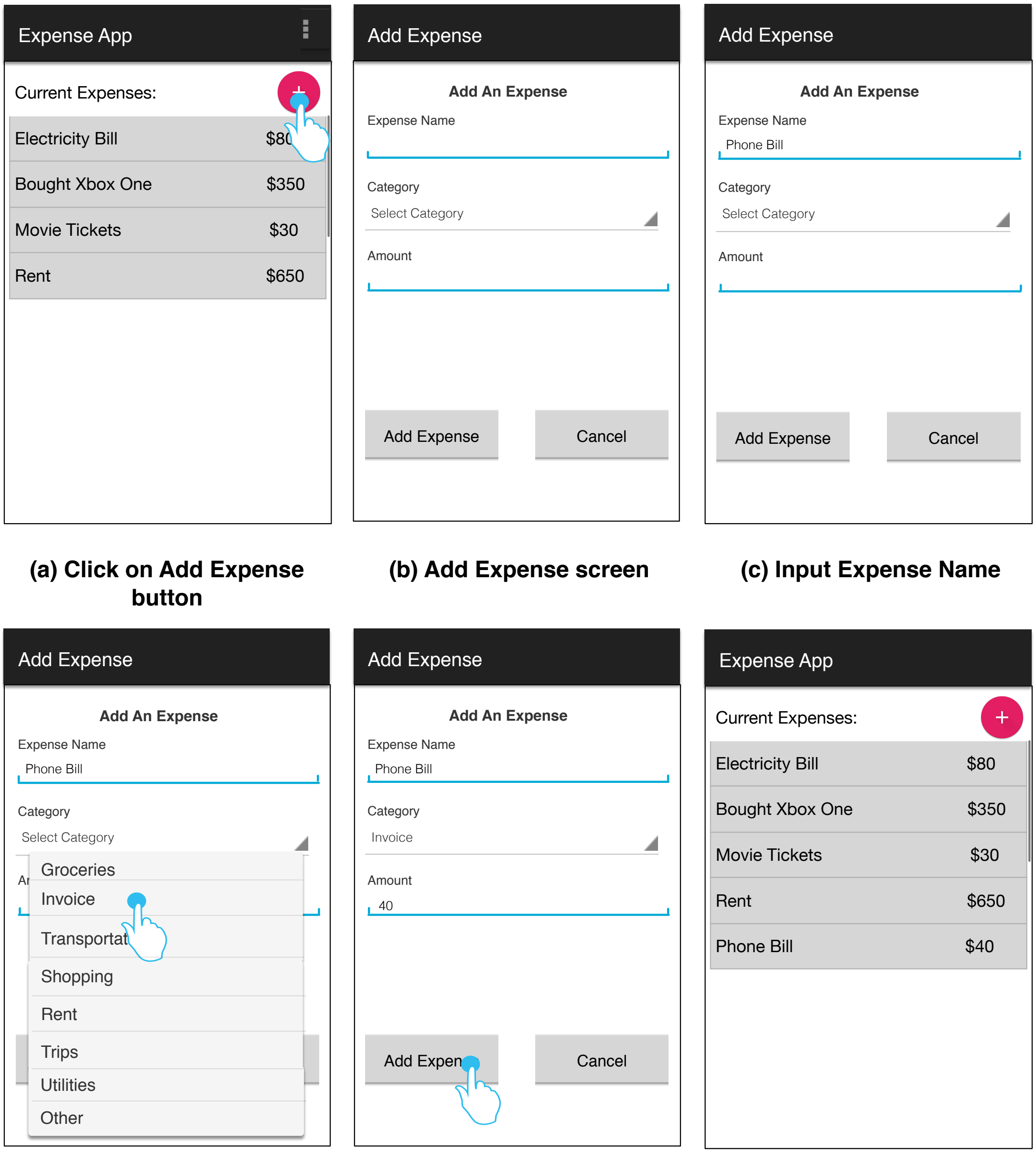
(d) Expense is deleted

Figure 1: ExpenseApp screen

Part A: ExpenseApp screen(30 points)

The main activity should start by displaying the ExpenseApp screen, with the following requirements:

1. When the app first starts, there should be no expenses added in the list. So it should display the ExpenseApp screen with a message, "There is no expense to show, Please add your expenses from the menu." (Figure 1(a)).
2. The list of expenses should be stored in Firebase realtime database.
3. The ExpenseApp screen should use a ListView/RecyclerView to display the list of expenses as shown in (Figure 1(b)).
   a) Long press on an item should delete the expense from the list. It should update Firebase, and refresh the ListView to indicate this change. A Toast should be displayed having the message, "Expense Deleted" (Figure 1(c, d)).
   b) Clicking on an expense item should display the ShowExpense screen, you should push the ExpenseApp screen on the screen stack.
4. Clicking on the add (+) icon should start the AddExpense screen.



(a) Click on Add Expense button

(b) Add Expense screen

(c) Input Expense Name

(d) Select Category

(e) Input amount

(f) Added to Expense List

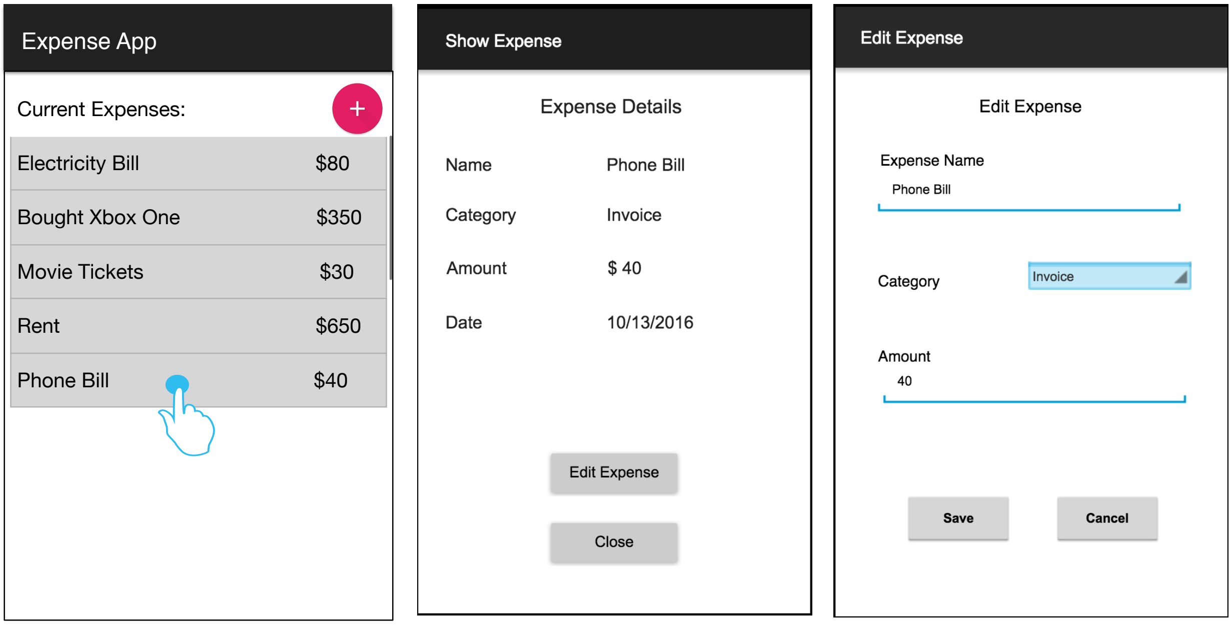
Figure 2: AddExpense screen

Part B: AddExpense screen (30 points)

This screen should enable user to add a new expense. You should complete the following tasks:

1. The user should be able to enter the expense name, category and amount. The app should take the current date as the expense date. This information should be stored in an Expense Object.
2. The categories should be in a selection pane as in the Figure 2(d). The categories you should include are: Groceries, Invoice, Transportation, Shopping, Rent, Trips, Utilities and Other.

3. Clicking on “Add Expense” button should validate the user’s input and ensure that all the fields are provided. If any field is missing, display a Toast to indicate the missing field. If all the fields are provided correctly, save the fields as an Expense object, and add the new expense to Firebase. Then display the main activity with the added expense, as shown in Figure 2(f).



Current Expenses:

Electricity Bill	$80
Bought Xbox One	$350
Movie Tickets	$30
Rent	$650
Phone Bill	$40

Name	Phone Bill
Category	Invoice
Amount	$ 40
Date	10/13/2016

Expense Name
Phone Bill

Category Invoice

Amount
40

(a) Click on an Expense item (b) Expense Details (c) Edit Expense

Figure 3: ShowExpenses and EditExpense screens

Part C: ShowExpenses screen (15 Points):

Implement the following requirements:

1. When the user clicks on an expense item in the ExpenseApp screen, the ShowExpenses screen should be started to show the details of selected expense item. as shown in Figure 3(b).
2. If the user clicks Edit Expense button, it should start EditExpense screen with preloaded values.
3. Upon clicking Close button, the screen should be closed and should navigate back to the ExpenseApp screen.

Part D: EditExpense screen (25 points)

Implement the following requirements:

1. It is identical to the Add Expense screen with preloaded values for the particular expense.
2. If the user makes changes to the values, and clicks on Save button, it should update the corresponding values to Firebase, and get back to the main screen with updated values.

   HINT: If you need an Unique ID you can use UUID.randomUUID(); or, any unique ID you want.