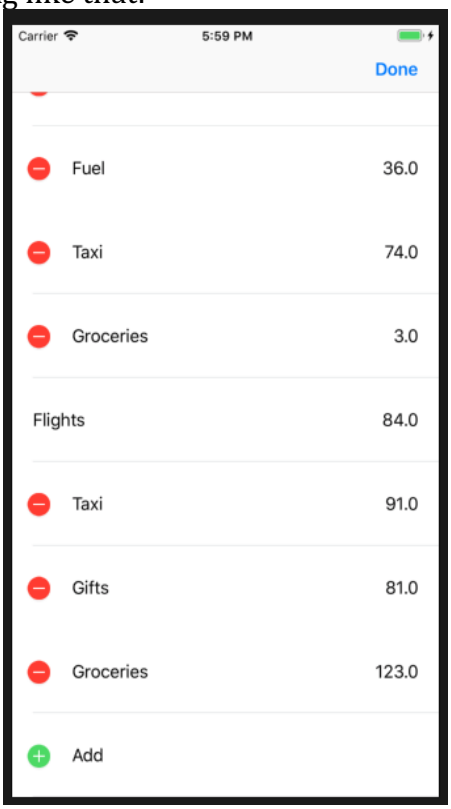
UITableView Editing Exercise

For this task we would like to create a simple app in which we could add and delete expenses using methods of **UITableViewDelegate** and **UITableViewDataSource** protocols.

Our final app should look something like that:



By following the tasks below you will learn a basic functionality that UITableView provides us.

Tasks:

1. First thing first, we will add the table view to our ViewController. Go to the main storyboard, find TableView in *object library* and drag and drop it to the view controller (don't forget to add the constraints and all necessary connections to related ViewController class)!

2. Let's now confirm our ViewController to the **UITableViewDelegate** and **UITableViewDataSource** protocols. The data for the expenses we would like to present is stored inside **var expenses: [ExpenseData]**. Define all required methods for protocols, but yet not fill, we will do it in subtasks below.

   Tip: You can find which methods are required by protocols by clicking on **UITableViewDataSource** with command key pressed, those without word *optional* are required ones.

   2.a. Now let's create a simple prototype cell with reuse identifier "ExpenseCell". Go back to main storyboard and do it inside our freshly created TableView. The cell has to be of the style *Right Detail,* you can set it in *Attribute Inspector.*

2.b Now go ahead and fill in the required methods for **UITableViewDataSource**, we have to dequeue the reusable cell we just created. Fill in the cells with expense data by setting the type as the title text label and amount as the detail text label of our cell.

3. So far so good! But if we tap on our cells nothing will happen, let's fix it. There is a separate view controller on our storyboard, can you connect it to so that upon selection of our ExpenseCell we will segue to it?

   Tip: Take a look at *didSelectRowAt* method

   Optional: If you noticed once the row is selected it stays like that, can you deselect it?

   Optional: The designer came to you and said that the cell height should be 80.0, can you do that?

4. Now lets go for the more interesting part, let's add and delete items in our list. There is predefined edit button we can use, take a look at *editButtonItem*. Lets place it as our *rightBarButtonItem.* Now for this task we will need new cell, so that when user taps on it while in editing mode we will create new data entry and insert the row to the tableView.

   4.a Go back to storyboard and create new prototype cell with reuse identifier "AddCell", the cell should have Basic style. This cell should always be at the end of the list, go ahead and deque it as you did for ExpenseCell before.

   4.b Let's specify the editing style for our cells, we would like our last cell to be of the style .insert and the rest of them of style .delete

   Tip: take a look at editingStyleForRowAt

   4.b Override the *setEditing* method, this method will be called once user presses editButtonItem. Once we are in editing mode we should insert our AddCell and otherwise we should remove our AddCell.

   Tip: Can you modify numberOfRowsInSection specifically for editing mode so that we show extra cell?

5. Right now our editing mode will do nothing, lets fix it by implementing the delete and insert actions itself. Implement **commitEditingStyle** method, if editingStyle is .delete then delete the entry and the row, if it is .insert then insert the new random entry to the expense data and the row.

   Tip: You can create new random data by using ExpenseData(type: .randomType(), amount: Double(arc4random_uniform(130))

6. Bonus: We require that expenses of type Flights cannot be deleted in editing mode, can you do that? Take a look at **canEditRowAt**

7. Bonus: We would like to add one more edit action for our cell once cell is swiped or is in editing mode, lets call it “Move To”, can you add it? Take a look at **editActionsForRowAt**