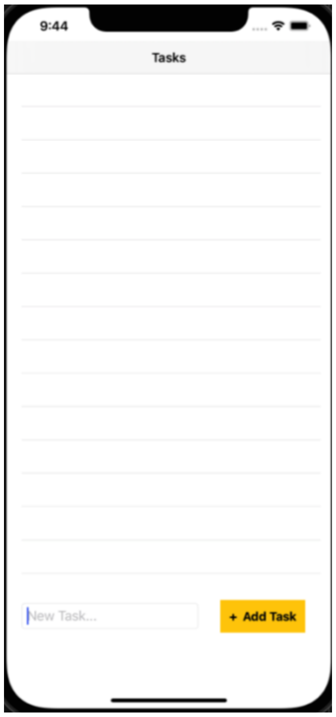
Name: Berk Cohadar
Girhub Repo: https://github.com/berkcohadar/cs441-xcode
Source Code: https://github.com/berkcohadar/cs441-xcode/blob/main/ShoppingList/ShoppingList/ViewController.swift

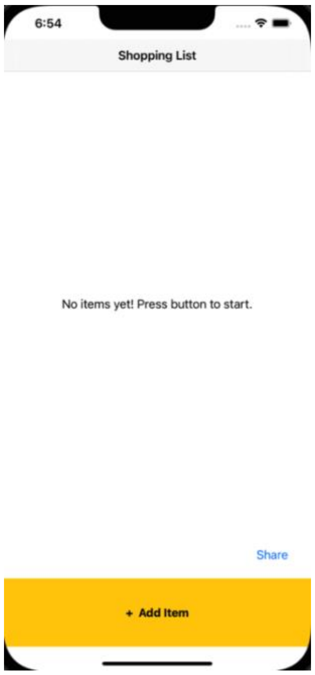
26 Sept 2021

WEEK REPORT - 3

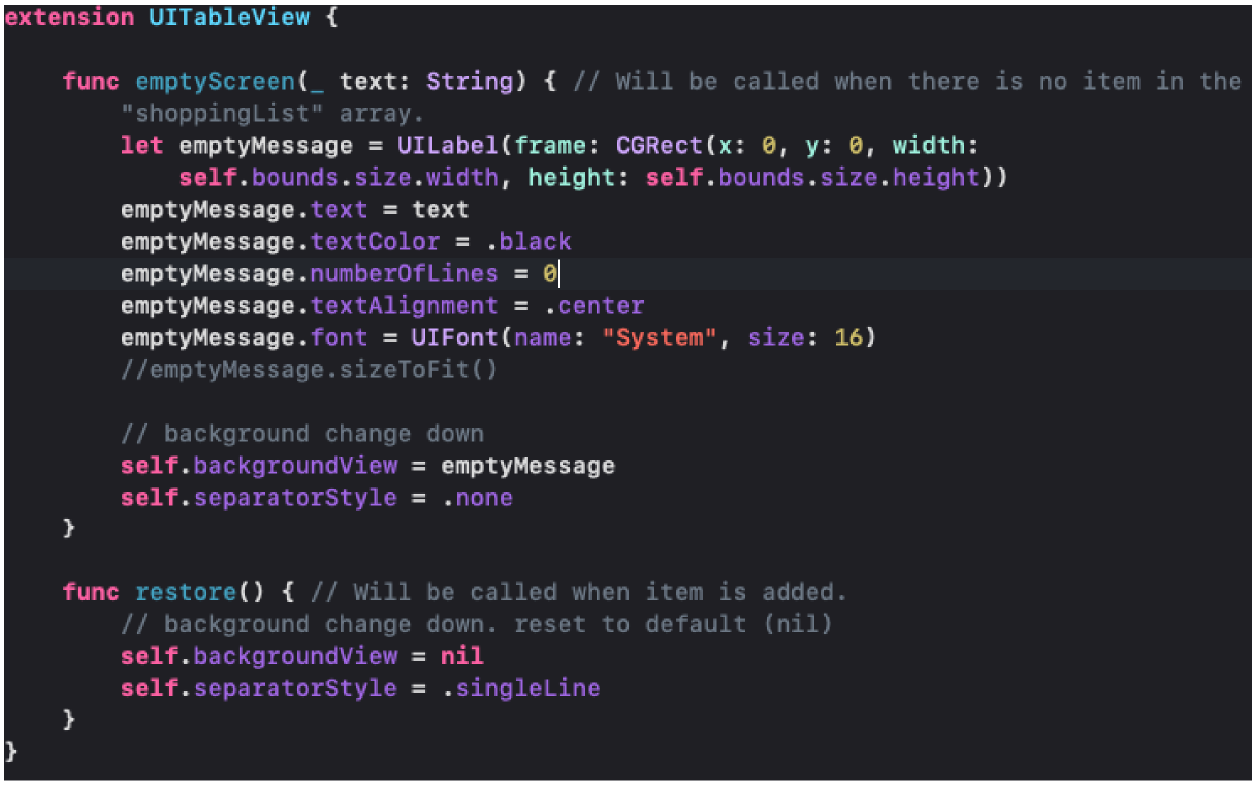
I finished the Shopping List assignment today. I've tried a lot of things this week. Most of them did not work completely. However, I implemented some useful stuff which worked so fine. Let's take a look. The left pic below is v0, and the pic on the right is v1.



⇨



I decided to add a blank page in order to increase the user experience. Now, user knows what to do. I've implemented another extension to the UITableView in order to have this feature. Let's see the code below.

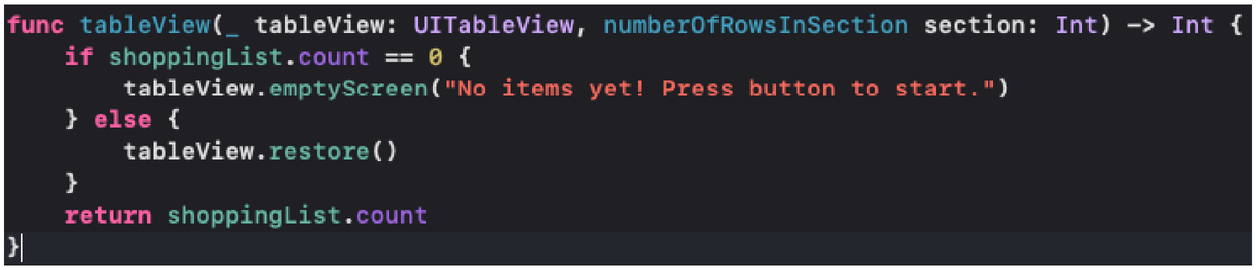


```swift
extension UITableView {

    func emptyScreen(_ text: String) { // Will be called when there is no item in the
        "shoppingList" array.
        let emptyMessage = UILabel(frame: CGRect(x: 0, y: 0, width:
            self.bounds.size.width, height: self.bounds.size.height))
        emptyMessage.text = text
        emptyMessage.textColor = .black
        emptyMessage.numberOfLines = 0
        emptyMessage.textAlignment = .center
        emptyMessage.font = UIFont(name: "System", size: 16)
        //emptyMessage.sizeToFit()

        // background change down
        self.backgroundView = emptyMessage
        self.separatorStyle = .none
    }

    func restore() { // Will be called when item is added.
        // background change down. reset to default (nil)
        self.backgroundView = nil
        self.separatorStyle = .singleLine
    }
}
```

In the first function "emptyScreen", I create a new UILabel, which will be shown in the middle of the app's screen. The text comes from caller, as an input. I generate "No items yet! Press button to start." as an empty screen warning. The styles can be changed inside the function, cool. At the end, the backgroundView of UITableView is changed to our UILabel. This was the trick.

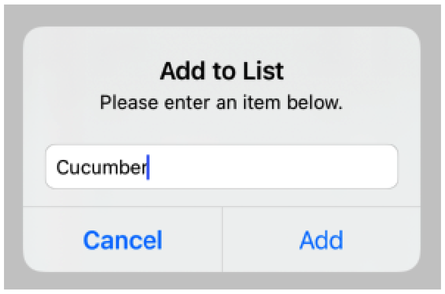
In the second function "restore", the backgroundView is set to default, which is none. Also, I learned we should change the separatorStyle to singleLine since we use TableView.

Below shows how I call this function above.

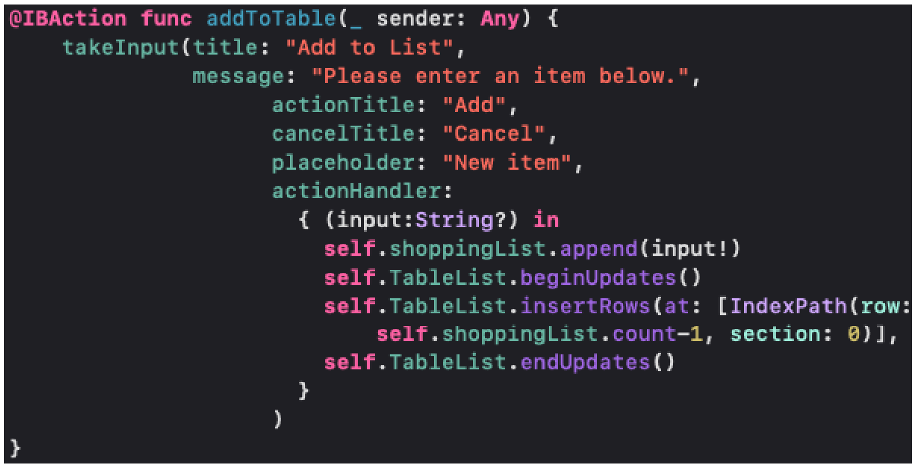


```swift
func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    if shoppingList.count == 0 {
        tableView.emptyScreen("No items yet! Press button to start.")
    } else {
        tableView.restore()
    }
    return shoppingList.count
}
```

Furthermore, I learned a cooler way to take inputs. I feel more relaxed when using that kind of boxes. You can enter what you want to add, then either press the 'enter' key or 'Add' button.

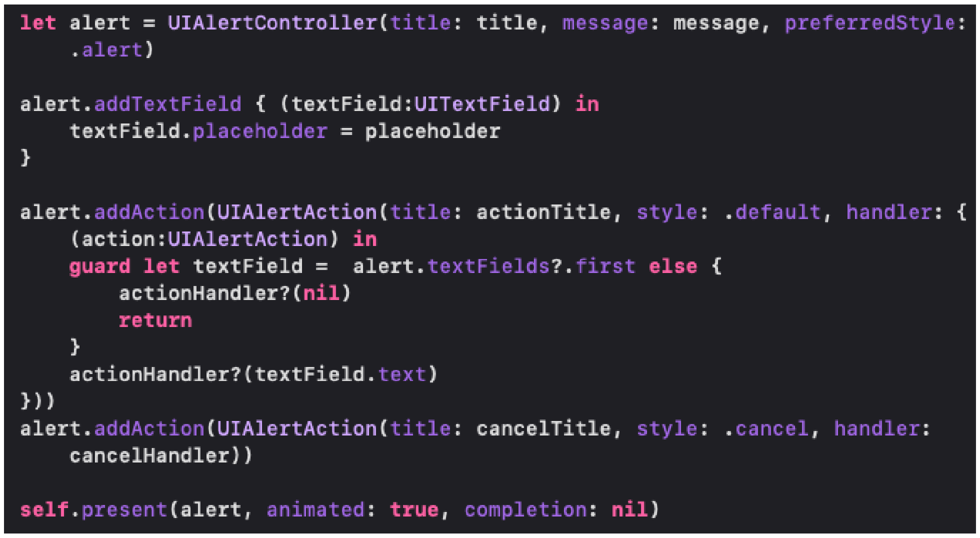


In order to open the box, I assigned a function to our “Add Item” button. The function calls another function which I constructed before. The takeInput function prepares the variables, names, action types, etc. and then completes the action. The action here is to add the input string to our pre-declared shoppingList array. Then, it starts the tableView update and insert a new row, then finishes the update.



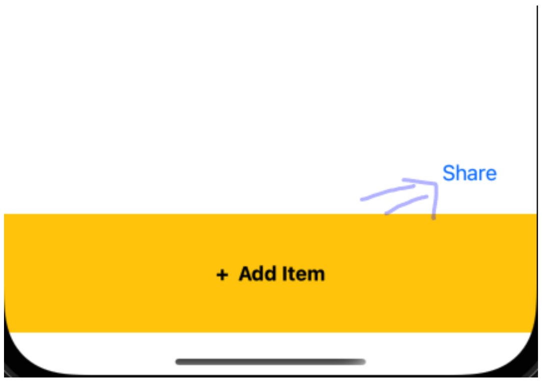
```
@IBAction func addToTable(_ sender: Any) {
    takeInput(title: "Add to List",
              message: "Please enter an item below.",
                    actionTitle: "Add",
                    cancelTitle: "Cancel",
                    placeholder: "New item",
                    actionHandler:
                      { (input:String?) in
                        self.shoppingList.append(input!)
                        self.TableList.beginUpdates()
                        self.TableList.insertRows(at: [IndexPath(row:
                            self.shoppingList.count-1, section: 0)],
                        self.TableList.endUpdates()
                      }
                    )
}
```

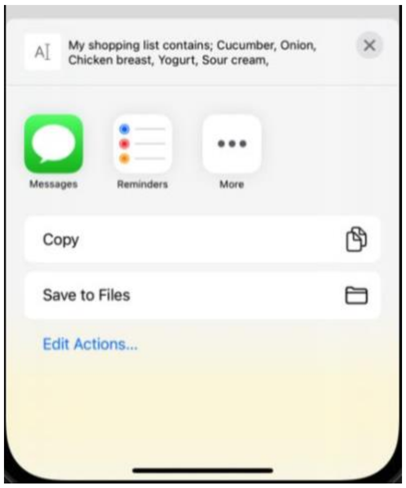
After having the parameters for alert box, the below code is triggered. This determines the title, message (subtitle), placeholder, and adds action to both ‘add’ and ‘cancel’ buttons.



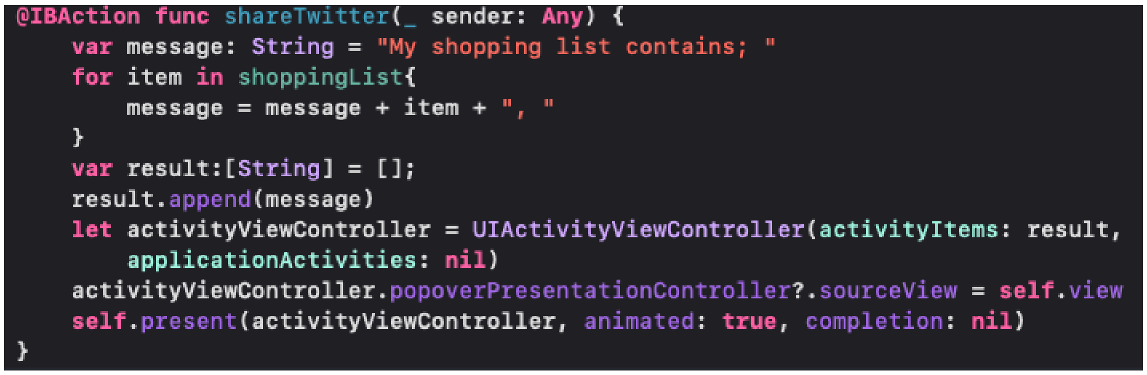
```
let alert = UIAlertController(title: title, message: message, preferredStyle:
    .alert)

alert.addTextField { (textField:UITextField) in
    textField.placeholder = placeholder
}

alert.addAction(UIAlertAction(title: actionTitle, style: .default, handler: {
    (action:UIAlertAction) in
    guard let textField =  alert.textFields?.first else {
        actionHandler?(nil)
        return
    }
    actionHandler?(textField.text)
}))
alert.addAction(UIAlertAction(title: cancelTitle, style: .cancel, handler:
    cancelHandler))

self.present(alert, animated: true, completion: nil)
```

I also learned how to add share link. I first tried a twitter share, however I realized that specific app share feature was deprecated in ios 11, sad. So, I added a share button which works with everything, cooler. It iterates the list items and generates a new, clear string. Then it becomes sharable as a plain text. The generated text: “My shopping list contains; 55” TV, HDMI Cable, TV Stand,”.

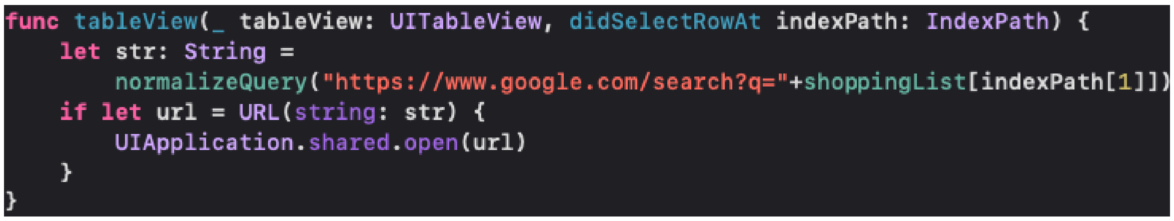



My shopping list contains; Cucumber, Onion, Chicken breast, Yogurt, Sour cream,

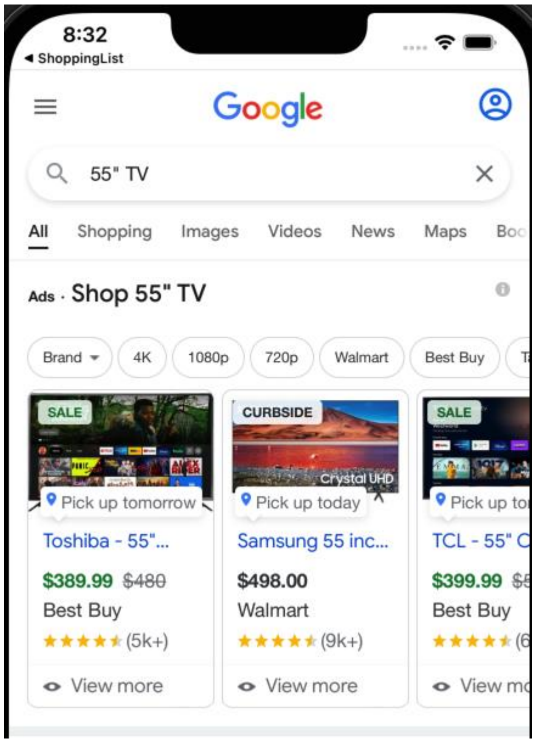
The implementation is shown below. We construct the sharable text, then send it to activity view controller. Then a popup comes out and shows the sharing options.


```
@IBAction func shareTwitter(_ sender: Any) {
    var message: String = "My shopping list contains; "
    for item in shoppingList{
        message = message + item + ", "
    }
    var result:[String] = [];
    result.append(message)
    let activityViewController = UIActivityViewController(activityItems: result,
        applicationActivities: nil)
    activityViewController.popoverPresentationController?.sourceView = self.view
    self.present(activityViewController, animated: true, completion: nil)
}
```

Last but not least, I added a look-up option to the items in our shopping list. While I am doing that, I learned how to redirect the user to another link, or application. The below code is to redirect the user to google search.


```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    let str: String =
        normalizeQuery("https://www.google.com/search?q="+shoppingList[indexPath[1]])
    if let url = URL(string: str) {
        UIApplication.shared.open(url)
    }
}
```

The result



8:32
◀ ShoppingList

Google

55" TV

All Shopping Images Videos News Maps Boo

Ads · Shop 55" TV

Brand 4K 1080p 720p Walmart Best Buy T

SALE	CURBSIDE	SALE
Pick up tomorrow	Pick up today	Pick up to
Toshiba - 55"...	Samsung 55 inc...	TCL - 55" C
$389.99 ~~$480~~	$498.00	$399.99 $5
Best Buy	Walmart	Best Buy
★★★★★ (5k+)	★★★★★ (9k+)	★★★★★ (6
View more	View more	View mo

See you next week!