Homework 9: Stock Search IOS App with Facebook Post – A Mobile Phone Exercise

1. Objectives

- Become familiar with Xcode, IOS App development and Facebook SDK for IOS.
- Build a good-looking IOS app.
- Add social networking features using the Facebook SDK for IOS.

2. Background

2.1 Xcode

Xcode is an integrated development environment (IDE) containing a suite of software development tools developed by Apple for developing software for OS X and iOS. First released in 2003, the latest stable release is version 7.3 and is available via the Mac App Store free of charge for OS X El Capitan users.

Features:

- Swift 2 support
- Playgrounds
- Interface Builder
- Testing
- User Interface Testing
- Code Coverage

The Official homepage of the Xcode is located at:

https://developer.apple.com/xcode/

2.2 IOS

iOS (originally iPhone OS) is a mobile operating system created and developed by Apple Inc. and distributed exclusively for Apple hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod touch. It is the second most popular mobile operating system in the world by sales, after Android.

The Official IOS home page is located at:

http://www.apple.com/ios/

The Official IOS Developer homepage is located at:

https://developer.apple.com/ios/

2.3 Swift

Swift is a general-purpose, multi-paradigm, compiled programming language created for iOS, OS X, watchOS, tvOS and Linux development by Apple Inc. Swift is designed to work with Apple's Cocoa and Cocoa Touch frameworks and the large body of existing Objective-C code written for Apple products. Swift is intended to be more resilient to erroneous code ("safer") than Objective-C and also more concise. It is built with the LLVM compiler framework included in Xcode 6 and later and uses the Objective-C runtime, which allows C, Objective-C, C++ and Swift code to run within a single program.

The Official Swift homepage is located at:

https://developer.apple.com/swift/

2.4 Facebook

Facebook is a social networking service launched in February 2004, owned and operated by Facebook, Inc. Users can add friends and send them messages, and update their personal profiles to notify friends about themselves and what they are doing.

Users can additionally post news feeds to their profiles, and these feeds may include images, besides text messages.

The Facebook homepage is available at:

http://www.facebook.com

Facebook provides developers with an application-programming interface, called the Facebook Platform.

2.5 Markit on Demand

Markit on Demand API provides detailed description about the stock information of company as well as historical stock values. You can refer to the API description on the following link:

http://dev.markitondemand.com/MODApis/

2.6 Amazon Web Services (AWS)

AWS is Amazon's implementation of cloud computing. Included in AWS is Amazon Elastic Compute Cloud (EC2), which delivers scalable, pay-as-you-go compute capacity in the cloud, and AWS Elastic Beanstalk, an even easier way to quickly deploy and manage applications in the AWS cloud. You simply upload your application, and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, auto-scaling, and application health

monitoring. Elastic Beanstalk is built using familiar software stacks such as the Apache HTTP Server, PHP, and Python, Passenger for Ruby, IIS 7.5 for .NET, and Apache Tomcat for Java.

The Amazon Web Services homepage is available at: http://aws.amazon.com/

2.7 Google App Engine (GAE)

Google App Engine applications are easy to create, easy to maintain, and easy to scale as your traffic and data storage needs change. With App Engine, there are no servers to maintain. You simply upload your application and it's ready to go. App Engine applications automatically scale based on incoming traffic. Load balancing, micro services, authorization, SQL and noSQL databases, memcache, traffic splitting, logging, search, versioning, roll out and roll backs, and security scanning are all supported natively and are highly customizable.

To learn more about GAE support for PHP visit the page:

https://cloud.google.com/appengine/docs/php/

3. Prerequisites

This homework requires the use of the following components:

3.1 Download and install Xcode

To develop iOS apps using the latest technologies described in these lessons, you need a Mac computer (OS X 10.10 or later) running the latest version of Xcode. Xcode includes all the features you need to design, develop, and debug an app. Xcode also contains the iOS SDK, which extends Xcode to include the tools, compilers, and frameworks you need specifically for iOS development.

Download the latest version of Xcode on your Mac free from the App Store.

To download the latest version of Xcode

- Open the App Store app on your Mac (by default it's in the Dock).
- In the search field in the top-right corner, type Xcode and press the Return key.
- The Xcode app shows up as the first search result.
- Click Get and then click Install App.
- Enter your Apple ID and password when prompted.
- Xcode is downloaded into your /Applications directory.

You may use any other IDE other than Xcode, but you will be on your own if problems spring up.

3.2 Add your account to Xcode

When you add your Apple ID to the Xcode Accounts preferences, Xcode displays all the teams you belong to. Xcode also shows your role on the team and details about your signing identities and provisioning profiles that you'll create later in this document. If you don't belong to the Apple Developer Program, a personal team appears.

Here is detailed documentation:

https://developer.apple.com/library/ios/documentation/IDEs/Conceptual/AppStoreDistributionTutorial/AddingYourAccounttoXcode/AddingYourAccounttoXcode.html

3.3 Install CocoaPods

CocoaPods is a dependency manager for Swift and Objective-C Cocoa projects. It has over ten thousand libraries and can help you scale your projects elegantly. You can install dependencies using it, we will need to install many third party modules and frameworks using it.

CocoaPods is built with Ruby and is installable with the default Ruby available on OS X. We recommend you use the default ruby. Using the default Ruby install can require you to use sudo when installing gems.

Run the command below in your Mac terminal:

```
$ sudo gem install cocoapods
```

Once you have created you Xcode project, you can start to integrate cocoapods into your project.

Further guides on how to integrate cocoapods are available at: https://cocoapods.org/

3.4 Create Facebook Developer application

You also need to create a Facebook Developer application as you did for your homework 8. Follow the following steps to get started:

a. **Download SDK**: Download the latest Facebook IOS SDK:

   https://developers.facebook.com/docs/ios

b. Instructions to **import in Xcode**:

   https://developers.facebook.com/docs/ios/getting-started

c. Create a **new app on Facebook** developer:

https://developers.facebook.com/quickstarts/?platform=ios

Please follow every step of the documentation to create the Facebook app.

4. High Level Design

In this exercise, you will develop an IOS Mobile application, which will have following functionality:

An Auto-complete edit box is provided to enter the stock name or symbol. The user can then select a stock from the suggestions. This feature needs to utilize a third party module. Details on how to use the modules would be covered in Section 5.

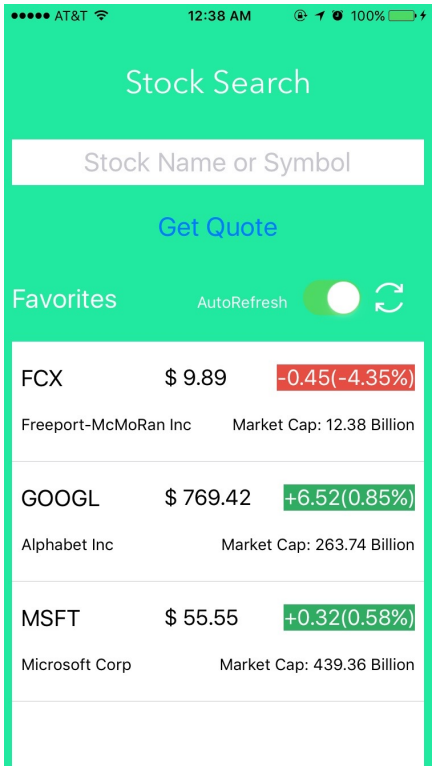

Figure 1. Search Form

- Once the user has provided data and selected a result from the Auto-complete list he would click on 'Get Quote', when validation must be done to check that the entered data is valid.
- Once the validation is successful, we would get the stock details using our PHP script hosted on Amazon Web Services/Google App Engine, which would return the result in JSON format. We would display the stock details in a 'UITableView' component in the 'Current' tab. Furthermore, our PHP script would be responsible for providing the data

to rendering the historical stock value chart in the 'Historical' tab and also rending the news articles in the 'News' tab.

5. Implementation

5.1 Search Form

You must replicate the Search Form, as shown in Figure 1. The field names remain the same as in Homework 8.

The interface consists of the following:

- An 'UITextField' component allowing the user to enter the company name or symbol.
- A button 'Get Quote' to get the quote, after validation.
- A clear button in the right of 'UITextField' to clear the component.

The form has two buttons:

- GET QUOTE: Validations are first performed, when the button is clicked. If the validations are successful, then the stock details would be fetched from the server (either hosted on Google App Engine or Amazon Web Services). However, if the validations are unsuccessful, appropriate messages would be displayed and no further requests would be made to the server.
- CLEAR: This button is inside the 'UITextField' and would clear it when pressed

5.1.1 AutoComplete

The user can enter the stock name or symbol in the text view to get the stock information from our PHP script. Based on the user input, the Auto-complete would display the all the matching companies and symbols (see Figure 2) by making a HTTP request. The requests would be made only when the user enters a minimum of 3 characters to avoid unnecessary network calls.

You can use third party modules and subclasses to implement AutoComplete feature. Possible choices are MLPAutoCompleteTextField, CCAutocomplete, MJAutoComplete. You can find plenty of them at Cocoaspod website and Github. Choose the one that you think is the best. But you may not get support when you run into problems using them.

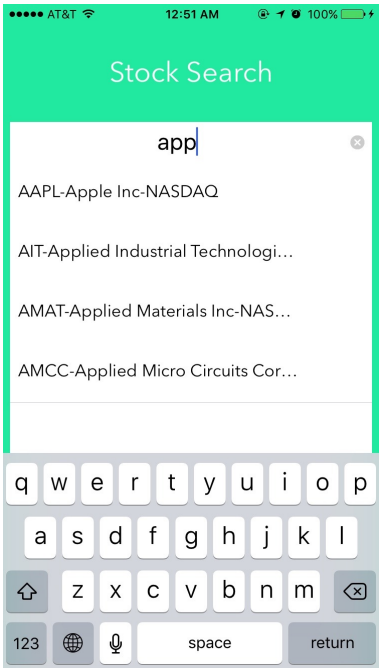

Figure 2: AutoComplete Suggestions

5.1.2 Validations

The following validations need to be implemented:

- Empty Entry: If the user does not enter anything in the 'UITextField', when he press the 'Get Quote' button you need to display an appropriate message to indicate the error.
- Invalid Selection: If the user enters an invalid entry which does not correspond to any valid stock symbol, you need to display an appropriate message to indicate the error.

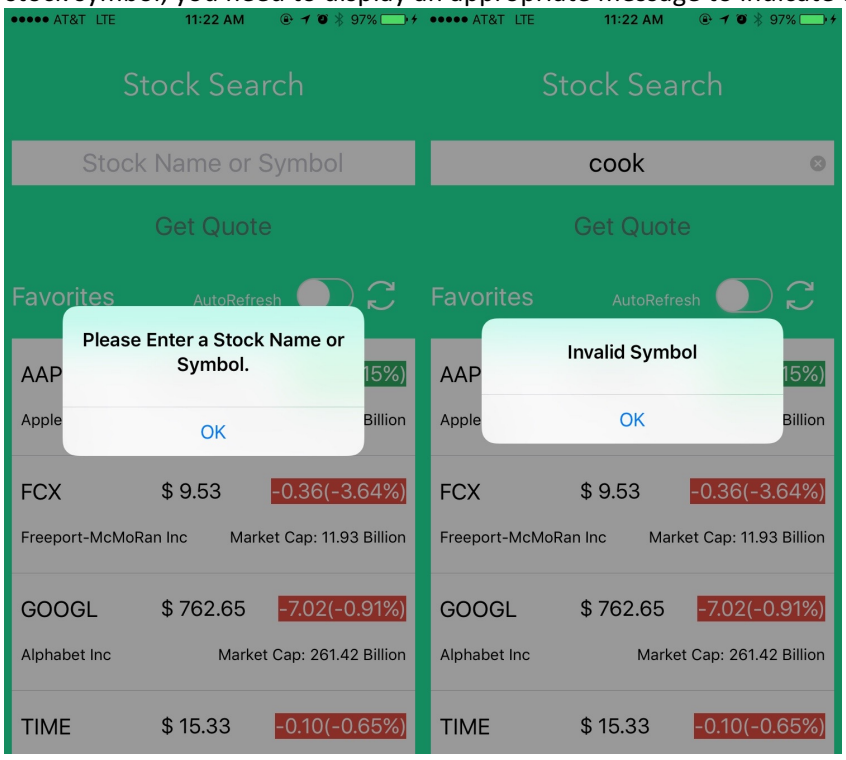

Figure 3: Validations

5.1.3 Get Quote Execution

Once the validation is successful, you should execute an HTTP request transaction to the PHP script which is located in the Google App Engine/Amazon Web Services.

The PHP script on Google App Engine/Amazon Web Services, modified after Homework 6, is used to retrieve data from Markit on Demand. You should pass the company symbol as a parameter of the transaction when calling the PHP script.

For example, if your Google App Engine/Amazon Web Services service is located at

http://example.appspot.com

and the user enters 'AAPL' as the company symbol, then a query using your lookup API of the following type needs to be generated:

http://example.appspot.com/?api=lookup&symbol=AAPL

The PHP script running on the Google App Engine/Amazon Web Services would extract the stock details of the company symbol, perform an API request to Markit on Demand, and returns the data in JSON data.

After obtaining the query results from the callback of the AJAX request, we would be displaying the results in a drop-down suggestion list.

Notice: In Homework 6 your PHP script produced HTML. In this exercise, the output must be changed to JSON and the PHP code must run on Google App Engine/Amazon Web Services.

5.2 Favorite List

The interface consists of the following:

- An Automatic Refresh switch
- Refresh button
- The Favorite 'UITableView' showing the list of favorite stocks.
- The Favorite 'UITableView' starts with an empty favorite list.

The Favorite stocks would be displayed in a list as per Figure 4. The data for the favorite list should be loaded from 'Core Data'. For more about 'Core Data' please refer to the appendix.

Figure 4: Favorite Stocks

Every entry in the favorite list should contains the following:

Field	Description
Symbol	Displays the symbol of the company
Company Name	Displays the name of the company
Stock Price	Displays the current stock price of the company
Change Percentage	Displays the percentage change of the price of the company. It should be rounded to 2 decimal places and the background should indicate an increase or decrease in the change. For example, the background should be green if the change is positive and red if it is negative. E.g. +1.50% in green background.
Market Cap	Displays the market cap of the current stock. Possible suffixes are {Billion, Million}. Each value should be calculated and appended with appropriate suffix and rounded to 2 decimals. E.g. 5.71 Billion or 5.71 Million or 571000.

Additionally, there needs to be a few important features:

- Automatic Refresh – A switch (Refer to Section 7): when it is on it should refresh the Favorite table every 10 seconds.
- Refresh button – Should refresh the Favorite table when pressed
- Stock Details – The stock details page should be loaded when the user clicks on any entry of the 'UITableView'.

5.3 Stock Details

The Stock Details section should be designed as per Figure 5.

The stock detail section should have 3 pages:

- Current Stock
- Historical Charts
- News Feeds

You can use three buttons to navigate between 3 pages above.

The back button in the header should navigate back to the Search Form.

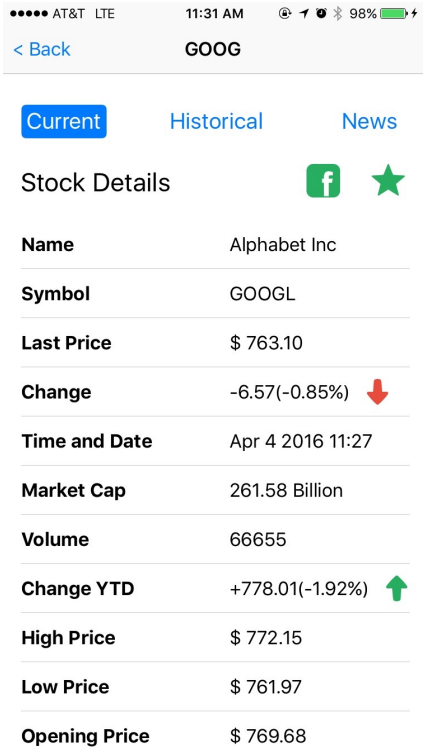

Figure 5: Stock Details

The Stock Details would be starting with the 'Current' page as loaded by default. Furthermore, the stock details would have a table showing all the stock values. The list of the stock details would be implemented using a 'UITableView'. The following stock values would be displayed:

Field	Description
Company Name	Displays the name of the company
Symbol	Displays the symbol of the company
Stock Price	Displays the current stock price of the company
Change (Change Percentage)	Displays the change and percentage change of the price of the company. It should be rounded to 2 decimal places, followed by an indicator(image) which indicates whether the change is positive or negative. For example, the image should be a green upward arrow if the change is positive and red download arrow if it is negative.
TimeStamp	Display the timestamp of the last updated price.

Market Cap	Displays the market cap of the current stock. Possible suffixes are {Billions, Millions, None}. Each value should be calculated and appended with appropriate suffix and rounded to 2 decimals. E.g. 5.71 Billion or 5.71 Million or 571000.
Volume	Displays the volume of the current stock
Change YTD (Change YTD Percentage)	Displays the Change of the stock from the beginning of the current year till date. It should be rounded to 2 decimal places, followed by an indicator(image) which indicates whether the change is positive or negative. For example, the image should be a green upward arrow if the change is positive and red download arrow if it is negative.
High	Displays the highest value of the stock's price for the current date.
Low	Displays the lowest value of the stock's price for the current date.
Open	Displays the opening value of the stock's price for the current date.

Below the list of stock details, you need to show the today's stock activity in the form of an image. This would be implemented using an 'UIImageView'. The image of the current daily chart of the stock of the company should be retrieved via Yahoo charts API, as was done in HW8.

5.4 Historical Charts

This tab represents the historical charts showing the value of the stock in the past. It has to be rendered as per Figure 6.

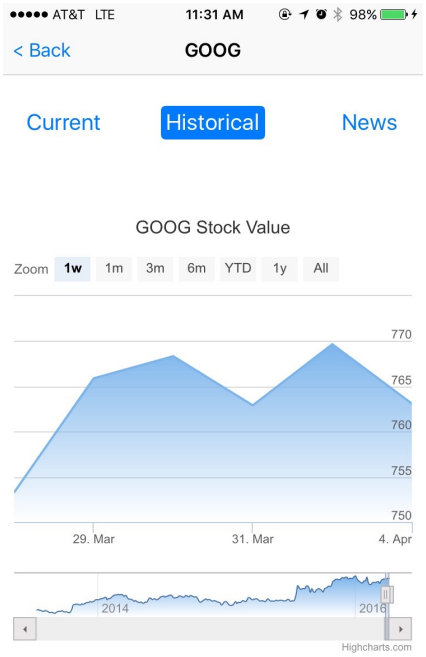

Figure 6: Historical Charts

This has to be implemented using a 'UIWebView' using HighCharts library. This is similar to implementation in HW8.

It should have the following details:

- It should have the following zoom levels: 1 week, 1 month, 3 months, 6 months, 1 year, YTD and All.
- It should default to showing 1 week worth of stock data.
- The title of the chart should be company symbol stock value.
- The X Axis should be date time.
- The Y Axis should be Stock Value

5.5 News Feed

This page represents 4 news articles related to the current stock. This would be rendered as per Figure 7.

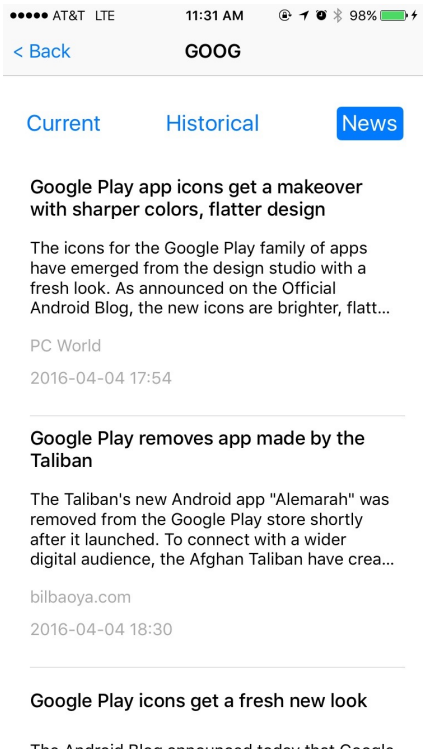

Figure 7: News Feed

The news articles need to be implemented in a 'UITableView'. Each of the entry in the list should display the following details:

Field	Value
Article Title	Represents the title of the news article
Article Content	Represents the content of the news article
Publisher	Represents the publisher of the news article
Date	Represents the date of the news article

You will be using the Google News Feed API or Bing Search API.

5.6 Facebook Share

The "Facebook" icon should be provided to post stock information on Facebook (similar to HW8). The Facebook post from the IOS app will contain exactly the same contents except for the caption as from HW8. See Figure 7 below.

When the user posts information on Facebook, a success message should be displayed on the screen. When the user cancels the Facebook dialog, a corresponding message should be displayed on the screen.

While posting information on Facebook, the app will authorize (login) the user to Facebook; requests the user permission to access the user's data; and then post the message to Facebook just like what was done in HW8.

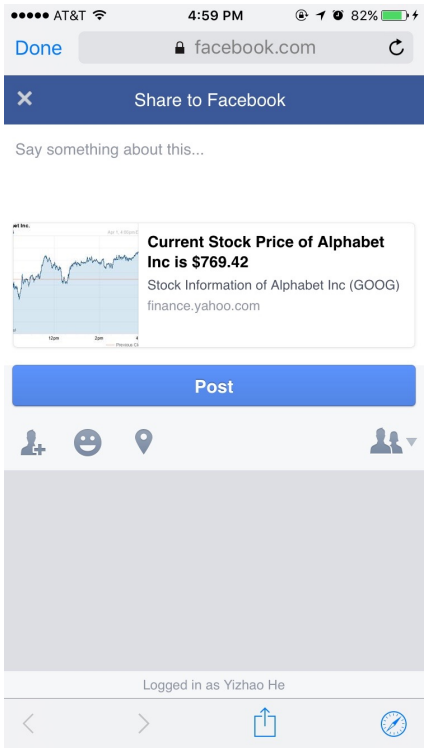

Figure 8. Facebook Share (on device)

When the user clicks on the title of the Facebook post (which is the same as Stock Name), he should be redirected to the Yahoo Finance page for the corresponding stock. See Figure 9 below.

Figure 9. Facebook share (on facebook.com)

6. Implementation Hints

See the HW9 IOS Clues file.

7. Material You Need to Submit

Unlike other exercises, you will have to "demo" your submission "in person" during a special grading session. Details and logistics for the demo will be provided in class, in the Announcement page and in Piazza.

You should also ZIP your project source directory and SUBMIT the resulting ZIP file. Make sure that the source path does not include the .app file in the product folder.