**Part I - Refactor**

Refactor your project to use the following libraries:

~~ChameleonFramework - all colors used in the project should be specified using methods  from this library (such as the constructor for UIColor that takes a hex).~~

~~SwiftyJSON - it is up to you how much you want to use this. However, we should see at  least a few places where you’ve utilized some of the methods from this library. Remember, this library helps you work with JSON. So it would be a good idea to use its methods to cleanup your code in your Firebase methods where you retrieve a JSON response from the database.~~

~~HanekeSwift - all your images should be loaded using this. The library loads an image  from any URL you specify, and also makes caching easy.~~

~~PromiseKit - this is the bulk of refactoring work. Using our newfound knowledge of promises and PromiseKit, integrate this library and make it so all async functions in your project now utilize Promises instead of just closures.~~

~~ObjectMapper - this should be an easy refactor that cleans up your code a lot. Refactor your model classes to make use of this library. Also find the places in your code where you are working with model objects.~~

Change your usage of model objects so that now you use the default constructor that ObjectMapper provides (ex: MenuItem(JSON: dict)).

* Also, whenever you need to write data to the database or persistent storage, please use the object.toJSON() method provided by ObjectMapper.
* ●  Style - update your codebase to follow the style guidelines outlined here: https://github.com/raywenderlich/swift-style-guide . The more you do the better. At the very least, please organize your code using extensions and do not use self unless required.
* **Part II - New Features**  Implement the following new features in your project:

~~●  Profile Picture - have the user upload a profile picture during signup. This should be  stored in Firebase storage. An imageUrl should be stored in the Firebase Database in the  User object so that you can easily reference the image.~~

~~Who’s Interested Modal - In case you don’t have this already, make it easy for the user to  view who’s interested in the event. Upon tapping a button, a popup should open that shows a list of everyone who is going. Feel free to use the AKModalView library from the MDBMemes demo project to make this job easier for you.~~

~~My Events - make it so the app is tab-based. The main feed is the 1st tab in the TabBarController. The 2nd tab is my events. This should only include events that the user has either created or said interested to. This means you should store the eventIds for events the user is interested in inside the User object. You should then use DispatchGroups from GCD to query these specific events.~~

~~Location - add a map to your Event Detail Screen that shows the location of the event. You will utilize MapKit & CoreLocation to do this. Please refer to this week’s resources to learn the concept.~~

~~Also add a button that automatically opens Apple Maps and provides directions to the location of the event. Hint: this is only a few lines of code.~~

~~Sort Feed - make sure your main feed only shows upcoming events, and sort them so that the event coming up the soonest is at the top.~~

**Part III - Open Ended Feature**

~~Go find a RESTful API online that you think would be useful to this project. Design a feature around it and integrate it. This is completely open ended. The only requirements are that you should use Alamofire to make at least 1 REST API call in your project. Please think of a feature that would actually be useful. If you want your project to be chosen as the one we all use, then this is one of the best places to differentiate it! This part should also help build you as a developer. Looking up tools and being able to learn how to use them from their documentation is a very important skill for any engineer.~~

~~A simple example of something you could do: add a tab that uses a REST API for movies to retrieve upcoming movies and suggest them to the user for potential socials. Another idea: use the Google Calendar REST API to add a social to the user’s google calendar. These are just a few ideas. Be creative though and implement something you think would be cool!~~