Micheal Willard
932480626
CS 496
Assignment 4: Mobile Development
Demo Video:
http://web.engr.oregonstate.edu/~willardm/cs496/cs496-a4.mov

Overview

The mobile app that was developed for this assignment is an iOS app that was built in Xcode. It takes the API backend that was developed for Assignment 3 and uses that for retrieving and storing data. The purpose of the application is for a manager of a baseball team to store scheduled games and allow players on that team to view the games.

The first view that is presented when opening the app is a table view showing all of the games that are stored in the database owned by the API. They are retrieved from https://hw3api.appspot.com/game. This data populates class objects and builds a table. From the first view, a user can enter two other views.

The second view uses a cocoa touch class where the user can select a game from the table view, simply by touching it. This launches the Game View. In the Game View, the user can see the opponent, location, date and time. Additionally in this view, there is a MapKit object. The MapKit MapView displays the location of the game on the map, and in certain circumstances the user's location. Finally, the user can exit this view and return to the table view by selecting the Back button in the upper left of the navigation header.

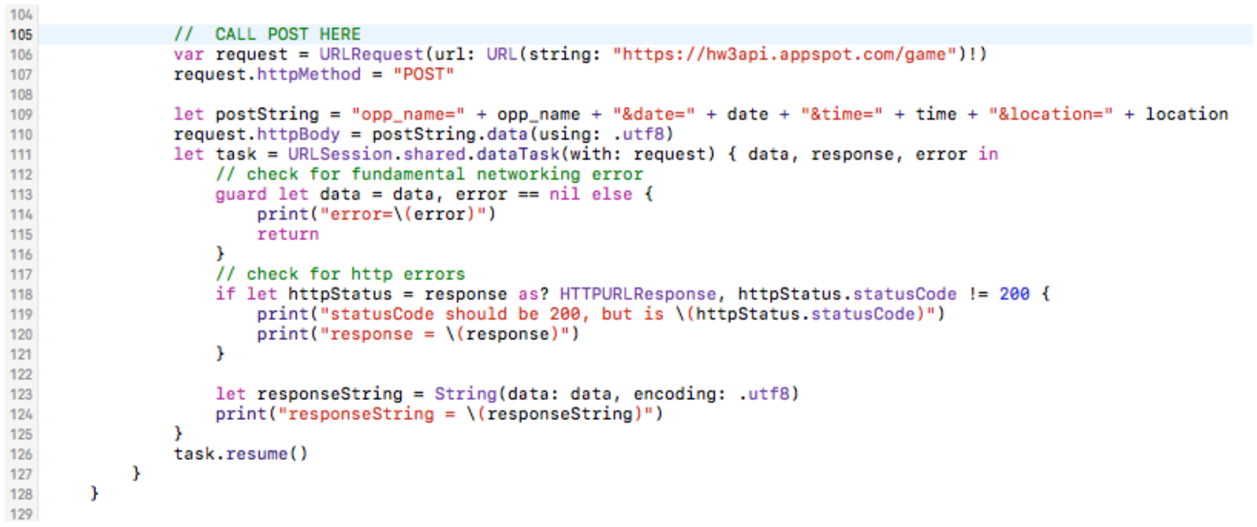
The third view is accessible from the Table View by a "+" icon in the upper right corner. This is the Game Add View. This view has 4 text fields that allow a user to input an opponent, location, date and time. The date/time picker native to iOS wasn't incorporated in this version due to time constraints and the already steep learning curve. From the Game Add View, the user can choose to Save the game or Cancel. Either option returns them to the home Table view. The Save button is disabled until all fields are filled in. These are all required entities, so this is how invalid, empty data is controlled. Once the user clicks Save, a POST request is made to the API. The user returns to the Table View and sees the new game added.

Mobile Feature

As discussed in the overview, the mobile feature leverage in this app is the MapKit MapView. The location string retrieved form the API database is reverse-geocoded and displayed as a pin on the Map. Clicking the Pin reveals the correct location. In the demo video the iOS simulator was not displaying the user location, despite the MapView being configured to display that. The user location failing in the iOS simulator is apparently a common problem. However, that would be another use of a Mobile Feature, by using the user's location.

API Interaction

The POST Request for adding a game:



```swift

            //  CALL POST HERE
            var request = URLRequest(url: URL(string: "https://hw3api.appspot.com/game")!)
            request.httpMethod = "POST"

            let postString = "opp_name=" + opp_name + "&date=" + date + "&time=" + time + "&location=" + location
            request.httpBody = postString.data(using: .utf8)
            let task = URLSession.shared.dataTask(with: request) { data, response, error in
                // check for fundamental networking error
                guard let data = data, error == nil else {
                    print("error=\(error)")
                    return
                }
                // check for http errors
                if let httpStatus = response as? HTTPURLResponse, httpStatus.statusCode != 200 {
                    print("statusCode should be 200, but is \(httpStatus.statusCode)")
                    print("response = \(response)")
                }

                let responseString = String(data: data, encoding: .utf8)
                print("responseString = \(responseString)")
            }
            task.resume()
        }
    }

```

This is called within the segue function that is activate by selecting the “Save” Button in the Add Game View Controller.