

## Assignment-2 Readme

Name: Chetan Umale

Student No:118221161

### Personal Contribution:

#### 1. Alert message:

- When the user clicks on 'fetch info from internet' button without entering the name of the player, an alert is displayed.
- When player information is not found on the internet, an alert is displayed.

```
324         DispatchQueue.main.async {
325             let alert = UIAlertController(title: "Alert", message: "Please enter player name.",
326                                           preferredStyle: .alert)
327             alert.addAction(UIAlertAction(title: "OK", style: .default, handler: { action in
328                 switch action.style{
329                     case .default:
330                         print("default")
331                         self.clear()
332                     case .cancel:
333                         print("cancel")
334                     case .destructive:
335                         print("destructive")
336                 }
337             })))
338         }
339         self.present(alert, animated: true, completion: nil)
340     }
341 }
342 }
343 }
```

#### 2. Fetch player information from internet:

- I have used a REST API called 'cric api' <https://www.cricapi.com> to fetch player data from the internet.
- For that purpose I have used 2 data tasks.
- The first data task will get the player id from this url:  
<https://cricapi.com/api/playerFinder?apikey=UGoeKO11HmhT3Q5eR1SRdT5a21E2&name=Tendulkar>
- The inputs for the first url are apikey and player name.
- The second data task will get the player info from the player id which was fetched by the first data task.
- The second data task will hit the following url:  
<https://cricapi.com/api/playerStats?apikey=UGoeKO11HmhT3Q5eR1SRdT5a21E2&pid=35320>

## Code for REST API:

```
133 //function to get data from internet using cric api
134 @IBAction func fetch_data(_ sender: Any) {
135
136     if(name_textfield.text != "")
137     {
138         let api_key = "UGoeKO11HmhT3Q5eR1SRdT5a21E2"
139         var url_info_text = ""
140         let search_text1 = name_textfield.text
141         let search_text2 = search_text1?.replacingOccurrences(of: " ", with: "%20")
142         let url_find_text = "https://cricapi.com/api/playerFinder?
143             apikey="+api_key+"&name="+search_text2!
144         var pid_search = ""
145
146         let configuration = URLSessionConfiguration.default
147         let session = URLSession.init(configuration: configuration)
148
149         let decoder = JSONDecoder()
150         decoder.keyDecodingStrategy = JSONDecoder.KeyDecodingStrategy.convertFromSnakeCase
151         let url_find = URL.init(string: url_find_text)!
152
153
154
155         //create player find datatask
156         let dataTask1 = session.dataTask(with: url_find) { (data, response, error) in
157
158             // check if there is any error
159             guard error == nil else {
160                 return
161             }
162
163
164             // check the status code
165             guard let response = response as? HTTPURLResponse else {
166                 return
167             }
168
169
170
```

```

171         guard (200...209).contains(response.statusCode) else {
172             return
173         }
174
175         guard let data = data else{
176             return
177         }
178         print(data)
179
180         //decode the response
181
182         let jsonResponse2 = try! JSONSerialization.jsonObject(with: data, options: []) as! [String
183             AnyObject]
184         // print(jsonResponse2)
185         //let pid_response = self.nullToNil(value: jsonResponse2["pid"])
186
187         let pid_arr: NSArray = jsonResponse2["data"] as! NSArray
188
189         // print(pid_arr.count)
190
```

```

// print(pid_arr.count)

if(pid_arr.count == 0)
{
    print("player not found")
    DispatchQueue.main.async {
        let alert = UIAlertController(title: "Alert", message: "Player not found",
            preferredStyle: .alert)
        alert.addAction(UIAlertAction(title: "OK", style: .default, handler: { action in
            switch action.style{
                case .default:
                    print("default")
                    self.clear()

                case .cancel:
                    print("cancel")


                case .destructive:
                    print("destructive")

            }}}))

```

```

}
else{

    print("player found")
    let jsonResponse3 = try! JSONSerialization.jsonObject(with: data, options: [])  Ini...

    let decoder = JSONDecoder()

    let model = try! decoder.decode(ResponseModel1.self, from: data)

    //pid_search = pid_response as! String
    pid_search = String(model.data.first!.pid)
    url_info_text = "https://cricapi.com/api/playerStats?apikey="+api_key+"&pid="+pid_search
    let url_info = URL.init(string: url_info_text)!

    // create the task
    let dataTask = session.dataTask(with: url_info) { (data, response, error) in

        // check if there is any error
        guard error == nil else {
            return
        }
    }

```

```

// check the status code
guard let response = response as? HTTPURLResponse else {
    return
}

guard (200...209).contains(response.statusCode) else {
    return
}

guard let data = data else{
    return
}

```

```

255
256         let jsonResponse1 = try! JSONSerialization.jsonObject(with: data, options: []) as!
257             [String: AnyObject]
258         let fullName = self.nullToNil(value: jsonResponse1["fullName"]) ⚠ Initialization of i...
259         let playingRole = self.nullToNil(value: jsonResponse1["playingRole"])
260         let born = self.nullToNil(value: jsonResponse1["born"])
261         let battingStyle = self.nullToNil(value: jsonResponse1["battingStyle"])
262         let bowlingStyle = self.nullToNil(value: jsonResponse1["bowlingStyle"])
263         let imageURL = self.nullToNil(value: jsonResponse1["imageURL"])
264
265         DispatchQueue.main.async {
266
267             if(born == nil)
268             {
269                 self.dob_textfield.text = "January 1, 1900"
270                 self.place_textfield.text = "Unknown"
271             }
272             else {
273                 var born_info = born!.components(separatedBy: ",")
274                 self.dob_textfield.text = born_info[0]+born_info[1]
275                 self.place_textfield.text = born_info[2]
276             }
277
278             if(imageURL == nil)
279             {
280                 self.player_image.image = UIImage(named: "blank.jpg")
281             }
282             else {
283                 self.player_image.downloadImage(url: URL(string: imageURL as! String)!)
284             }
285
286             if(playingRole == nil)
287             {
288                 self.role_textfield.text = "Unknown"
289             }
290             else {
291                 self.role_textfield.text = playingRole as! String ⚠ Treating a forced downc...
292             }
293         }
294
295

```

```

        self.player_image.image = UIImage(named: "blank.jpg")
    }
    else {
        self.player_image.downloadImage(url: URL(string: imageURL as! String)!)
    }

    if(playingRole == nil)
    {

        self.role_textfield.text = "Unknown"
    }
    else {
        self.role_textfield.text = playingRole as! String ⚠ Treating a forced downc...
    }

```

```

        self.role_textfield.text = playingRole as! String ⚠ Treating a forced downc...
    }

    if(battingStyle == nil)
    {

        self.batting_textfield.text = "Unknown"
    }
    else {
        self.batting_textfield.text = battingStyle as! String ⚠ Treating a forced d...
    }

    if(bowlingStyle == nil)
    {

        self.bowling_textfield.text = "Unknown"
    }
    else {
        self.bowling_textfield.text = bowlingStyle as! String ⚠ Treating a forced d...
    }

    }

    }
    dataTask.resume()
}
dataTask1.resume()

```

### 3. Download player image from internet:

The JSON fetched by the REST API also contains an url for player image. I have downloaded the image of the player using that url.

#### Code:

```
452 extension UIImageView{
453     func downloadImage(url: URL){
454
455         let session = URLSession(configuration: .default)
456         let task = session.dataTask(with: url) { (data, response, error) in
457             if let data = data {
458                 if let image = UIImage.init(data: data){
459                     print("we have the image")
460                     print(image)
461                     DispatchQueue.main.async {
462                         self.image = image
463                     }
464                 }
465             }
466         }
467     }
468 }
469
470 }
471 task.resume()
472 }
473 }
474 }
```

#### Search Functionality:

The user has to select the search category before searching.

There are 4 categories available: Name, role, batting, bowling.

Valid queries for search are:

Name: name of the player

role: batsman, bowler, wk, allrounder

batting: right, left

bowling: right, left, fast ,off , leg

#### Internet data:

The following player names can be used to test this functionality:

- Steyn
- Langer
- tendulkar or any name from the player list on the first page