MATES Computer Science

Senior Capstone Project Bi-Weekly Progress Report

Project Title	Athena
Team Members	Me
Dates Covered by Report	4/22-5/2
Link to Github	https://github.com/AlexBH74/Athena

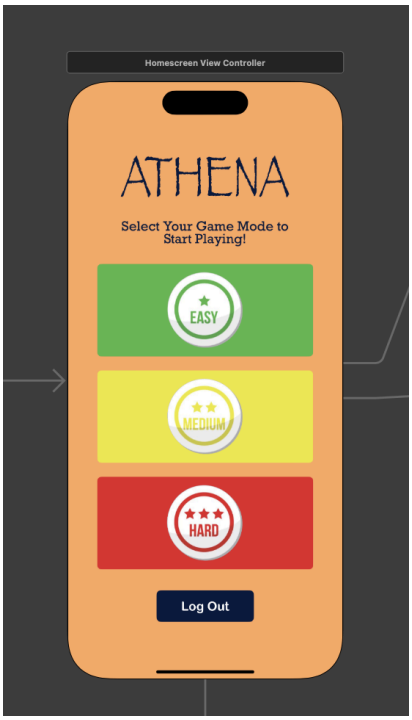
1. **Summary of Project** (Provide a one paragraph summary of your project. You can largely copy/paste this from one progress report to the next, unless there are significant changes.)

   My plan is to create an Apple game for daily trivia. The app will draw from a MySQL database of trivia questions and multiple choice answers. When the app opens there is a login and create account screen. After logging in there is an onboarding screen that has slides that list the game's summary and general instructions. After clicking next past all of the slides the screen changes and there is an interface with easy, medium, and hard modes. Each player will be able to play one question for each mode each day. Once the user presses the mode of the question they will be able to click a start button. A timer will start after the button is clicked and the screen will display the question with four multiple choice answers. The timer can be paused but the trivia question is blocked from view if the game is stopped. At the end whether or not you answered correctly and your time will be displayed in a pop-up. There will be a separate tab on the app containing a table of each user's career stats, line graphs/trend lines and bar plots showing progress over time

2. **Summary of Progress this Period** (Provide a high-level, one paragraph overview of what was accomplished this progress period collectively by the team.)

   In the past two weeks I have worked on developing the timer function. The first thing I did was get the timer actually counting and the pause/play button working. This code took me 2-3 days. After finishing the stopwatch I went on to implement the blur function when the pause button was clicked which took only one class period. Then I blurred the initial screen and added a start time button, which when clicked started the timer and displayed the trivia question. The last thing I did in relation to the timer function was edit the hierarchy and hide certain items so that each piece would appear properly when called. After finishing the timer I went on to create the popup screens after you answer the question. The screen blurs again and a container view appears telling you whether you got the question correct or not. This is where my major problem started this sprint. It has taken me the last three days to get to where I am now, and I'm still having issues with refreshing the view.

3. **Detailed Progress this Period, separated by Team Member** **(Provide detailed information on the progress that you made in the reporting weeks. Include screenshots of code, your game or website, etc. Each team member should have a separate subsection covering their accomplishments. Not including screenshots, this section should be 1-2 pages.)**



Above to the left is my home screen view controller. Above to the right is my easy view controller with the two container views off to the side.

```
55      private var correctTimes: [String] {
56          get {
57              if let storedTimes = UserDefaults.standard.object(forKey: "easyCorrectTimes") as? [String] {
58                  return storedTimes
59              } else {
60                  return []
61              }
62          }
63          set {
64              UserDefaults.standard.set(newValue, forKey: "easyCorrectTimes")
65          }
66      }
67
68
69      override func viewDidLoad() {
70          super.viewDidLoad()
71
72          self.blurEffect.isHidden = false
73          self.startButton.isHidden = false
74          self.pausedLabel.isHidden = true
75          self.correctPopUp.isHidden = true
76          self.incorrectPopUp.isHidden = true
77
78          loadDataFrameFromCSV()
79          displayRandomTrivia()
80
81          correctPopUp.layer.cornerRadius = 25
82          correctPopUp.clipsToBounds = true
83          incorrectPopUp.layer.cornerRadius = 25
84          incorrectPopUp.clipsToBounds = true
85      }
86
```

Above is the code for saving the correct times and hiding certain objects.

```
138
139         @IBAction func startClicked(_ sender: Any) {
140             startCounting()
141             self.startButton.isHidden = true
142         }
143
144
145         @IBAction func aClicked(_ sender: Any) {
146             let answer = correctAnswer
147             if answer == titleAnswer1.titleLabel?.text {
148                 correct = true
149                 incorrect = false
150             } else {
151                 incorrect = true
152                 correct = false
153             }
154             stopCounting()
155
156             if incorrect == true {
157                 print("Incorrect!")
158                 resetTimer()
159                 self.incorrectPopUp.isHidden = false
160             } else if correct == true {
161                 print("Correct!")
162                 correctTimes.append(timeString)
163                 //correctTimes = [] //comment out to make correct times save
164                 print(correctTimes)
165                 UserDefaults.standard.set(true, forKey: "correctShowing")
166                 self.correctPopUp.isHidden = false
167
168             }
169         }
```

Above is the code for the start button one of the answer buttons

```
240     func startCounting() {
241         timerCounting = true
242         timer = Timer.scheduledTimer(timeInterval: 1, target: self, selector: #selector(timerCounter), userInfo: nil,
                repeats: true)
243         let pauseImage = UIImage(named: "pause")
244         pauseplayImage.image = pauseImage
245         self.blurEffect.isHidden = true
246     }
247
248     func stopCounting() {
249         timerCounting = false
250         timer.invalidate()
251         self.blurEffect.isHidden = false
252     }
253
254     func resetTimer() {
255         self.num = 0
256         self.timer.invalidate()
257         self.timerLabel.text = self.makeTimeString(minutes: 0, seconds: 0)
258     }
259
260     @objc func timerCounter() -> Void {
261         num = num + 1
262         let time = secondsToMinutesSeconds(seconds: num)
263         timeString = makeTimeString(minutes: time.0, seconds: time.1)
264         if timeString == "60:00" {
265             incorrect = true
266             print("Incorrect!")
267             stopCounting()
268             resetTimer()
269         } else {
270             self.timerLabel.text = timeString
271         }
272     }
```

Above is the code for the start/stop timer functions, the reset functions, and the actual counter used for the to run the stopwatch

```
273
275     @IBAction func pauseClicked(_ sender: Any) {
276         if(timerCounting) {
277             stopCounting()
278             let playImage = UIImage(named: "play")
279             pauseplayImage.image = playImage
280             self.pausedLabel.isHidden = false
281             view.bringSubviewToFront(pauseplayImage)
282             view.bringSubviewToFront(pauseButton)
283             view.bringSubviewToFront(timerLabel)
284         } else {
285             startCounting()
286             self.pausedLabel.isHidden = true
287             view.sendSubviewToBack(timerLabel)
288             view.sendSubviewToBack(pauseButton)
289             view.sendSubviewToBack(pauseplayImage)
290             view.sendSubviewToBack(darkerSpace)
291         }
292     }
293
294     func secondsToMinutesSeconds(seconds: Int) -> (Int, Int) {
295         return (((seconds % 3600) / 60), ((seconds % 3600) % 60))
296     }
297
298     func makeTimeString(minutes: Int, seconds: Int) -> String {
299         timeString = ""
300         timeString += String(format: "%02d", minutes)
301         timeString += ":"
302         timeString += String(format: "%02d", seconds)
303         return timeString
304     }
}
```

Above is the code for the pause/play button and the code that calculates and returns the time as minutes and seconds. It then makes that time a string and returns it back.

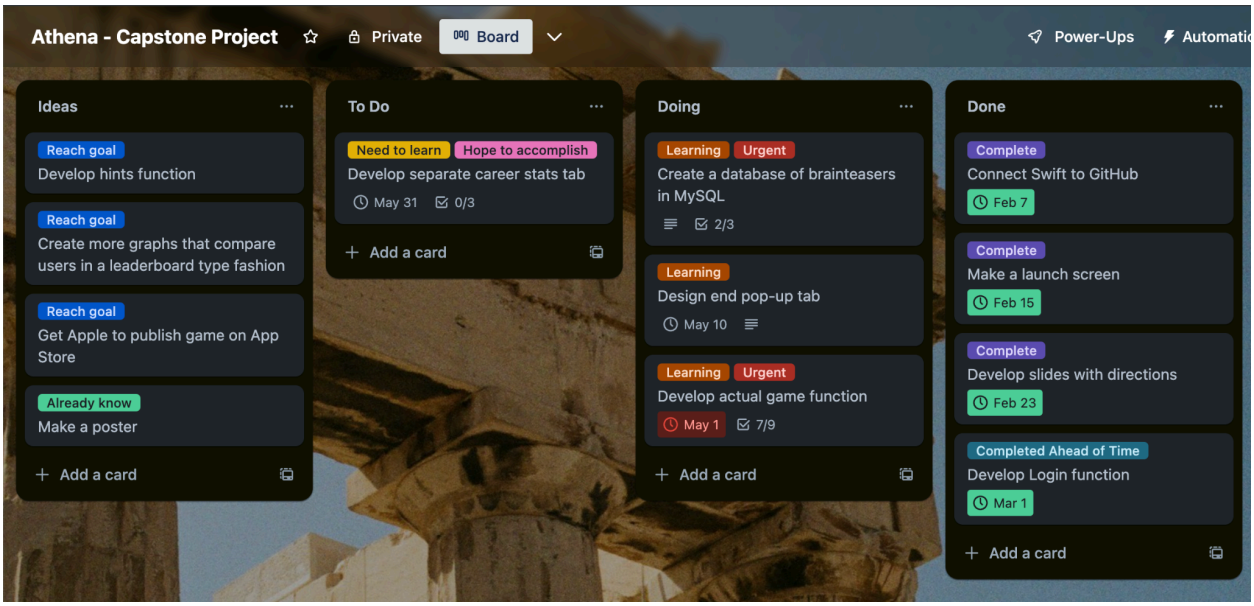
I attached the screen recording to the classroom showing the swift code output from the homescreen onwards.

4. **Difficulties Encountered this Progress Period** (Provide detailed information on the difficulties and issues that you encountered in the reporting weeks. Discuss mitigation strategies for how you got around or plan to get around these issues.)

   Over the last few weeks the only issue that has taken significant time to fix is refreshing the pop up. When I load the question view controller, the container views load with it. The problem is I want to display the time it took for the player to answer correctly on the container view, so I have to reload the container view code after the question was answered. So far I have not been able to get the view to refresh. In the next two weeks I hope to accomplish my goal of displaying the time by using a refresh feature, but as a last resort I will create a makeshift popup using conditional labels in the original game controller.

5. **Updated Trello Board and Discussion** (Provide screenshot of and link to updated Trello board. Discuss any changes made to board since last progress report and why.)

   https://trello.com/b/ldYqXQeq/athena-capstone-project



6. **Tasks to Be Worked on in Next Progress Period** (Discuss the tasks to be worked on in the following two weeks. Discuss who is working on each.)

   I will finish editing the pop ups and fix the refresh. Then I will edit the other two game controllers to display their own questions and function the same way as the easy view controller.

7. **Additional Information** (Provide any additional information that you want to provide in this section; for example, one of your teammates is going away next week, your Github account is gone, etc. It could be good news as well.)