

Sprint challenge for Sprint 8 of Lambda's iOS course

Edit

Manage topics

5 commits

1 branch

0 releases

2 contributors

Branch: master

New pull request


Create new file

Upload files

Find file

Clone or download

This branch is even with LambdaSchool:master.

 SpencerCurtis Call completion in UI testing helper file.

Message Board.xcodeproj

Add starter project and README.md with sprint challenge

Message Board

Call completion in UI testing helper file.

Message BoardUITests

Add starter project and README.md with sprint challenge instructions

5 months ago

MessageBoardTests

Add starter project and README.md with sprint challenge instructions

5 months ago

.gitignore

Initial commit

7 months ago

README.md

Update README.md

5 months ago

README.md

Debugging and Testing Lambda Message Board

Instructions

Please read this entire README to make sure you understand what is expected of you before you begin.

This sprint challenge is designed to ensure that you are competent with the concepts taught throughout Sprint 8, Code Quality.

Begin by forking this repository. Clone your forked repository to your machine. Use the provided Xcode project in the repository. It includes the project to debug and create tests for. Commit as appropriate while you work. Push your final project to GitHub, then create a pull request back to this original repository.

You will have 3 hours to complete this sprint challenge

If you have any questions about the project requirements or expectations, ask your PM or instructor. Good luck!

Requirements

The goal of this application is threefold. You must debug the Lambda Message Board project, write unit tests, as well as UI tests for it.

Please provide your own Firebase Database URL so that you and your classmates don't mess up the database for everyone.

As this is a project that you have written before, do not use past code. It will not benefit yourself at all in the long run.

NOTE: For the **UI tests**, code has been provided in the project that will use mock data in order to avoid making network calls. There are a few comments that show where code has been added to support this feature. You may disregard those parts when debugging the app. Also, there are two threads with messages that will load every time the app runs but the threads and messages you create will not persist. **This is normal and should not be regarded as a bug.**

The requirements for this project are as follows:

1. Find the bugs in the project. **Take note of them** then write unit tests that will fail until you fix them.
2. Once you have written the unit tests, fix the bugs.
3. Write UI tests for the project using the `TestPage` pattern. Test the functionality of the app such as creating threads, creating messages, etc. You are welcome to use a code snippet of the `expect` methods if you wish.