iPhone Software Engineering
Study Period 3 (2019)
Assignment 1 - Milestone 2

Due: **Sunday 20th October, 2019 (11:59 P.M.)**

Assignment Type: **Individual / Group (2 maximum)**
WE STRONGLY ADVISE AGAINST ATTEMPTING THIS ASSIGNMENT INDIVIUALLY.

Total Marks: **15 marks (15%)**

Congratulations

You have completed the preliminary work to enable you to begin building your app.

The final deliverable for this assessment is an iOS app with a Swift 4.0 code base and an updated report:

What you need to do

You need to complete and submit:

1. Read this document throughly
2. Read the marking rubric throughly
3. Submit the following:
    a. An updated report
    b. A prototype Xcode project

You need to mock up the final app as a working solution for demonstration to your client.

This prototype should have the look and feel of the final product and allow the client to interact with the app to get a very clear idea of the finished product.

If you change your implementation significantly from the Assignment 1 (M1) plan, then you must ensure that you are still meeting all the requirements set out in the first specification and marking rubric.

IMPORTANT:

If modifying an existing app, you must clearly demonstrate how you are making the app unique.

Your app may not contain more than 25% of the app you are working to improve.

The app must have between 4 - 7 screens and utilise different types of Views.

Report

The report from your first assignment should be updated to reflect any changes you have made to the design and/or function of the app as a result of your iterative process of development.

In addition, it should now also include a set of high fidelity wireframes that represent your completed app.

You do not need to manually create these, you can just take screen shots from the simulator of your app as it is prior to submission.

Prototype Implementation

The prototype should be designed to run on an **iPhone SE.**

Your prototype should include all of the features you plan to have in your app, but the focus will be on building the UI, enabling interaction and displaying/editing data. The data will be hard coded.

Your *client* should be able to open the app on a simulator or device and use it as if it were the finished product. The essential idea behind this implementation is to show the client how the app works. The client should be unaware that the app is not using a database or networking.

Your prototype should behave as closely as possible to the finished product.

The prototype is **NOT A DRAFT** that is only partially built.

The app must be developed using storyboards and submissions that create the whole app from code will not be assessed and receive a zero grade.

Using code to create some elements of the app will be accepted, but the app must demonstrate an understanding of building apps from a storyboard with a minimum of 4 scenes loaded from the storyboard.

All code in the app should be modelled using objected oriented principles. Data should be represented by: Enums, Structs and Classes. Modelling data as simple string values will incur significant marking penalites.

Design Pattern (Architecture)

The app must be designed on an MVVM architecture.

UITests

You should also implement a UITest for at least four scenes in your app. Every UITest should be complete by validating that all the elements are present, have the correct values and checks are made of the results of any user interaction.

You should also include another UITest that is recorded demonstrating a user accessing the various features of the app. Typically this would be the use case that you provided in your app statement.

Cocoa Framework

You should implement a feature that makes use of the device hardware features such as the camera.

(Networking, Persistence, Facial Recognition & AI)

The prototype must be designed to enable the inclusion of some advanced features in the final version.

Networking:
Research three networking libraries, complete a comparitive analysis to justify an implementation in the context of your app.

Implement the networking feature using the library you chose.

Persistence:
You should now implement your database using the Core Data framework.

Please note other database solutions such as: SQLLite, Firebase, Realm or other cloud based solutions that manage remote and local data are not permitted to satisfy this requirement.

IMPORTANT:

You do not need to implement the networking, database, facial recognition or advanced features of your app at this stage.

However, you should 'mock' up the database and networking features, meaning that you hard code the data using Arrays and/or Dictionaries.

You can choose to implement such solutions in addition to the core requirement but not in place of a core data implementation.

Example Mockup

You can simply simulate the database functions by ensuring your data is represented as objects at runtime and those objects are updated when performing the C.R.U.D. operations.

For example:

In your table view if an item is deleted, it should be removed from the table view and the underlying datasource (array / dictionary) should be updated, so the deleted data does not reappear at a later point.

In a detail view, if a property of some object is updated then that should be reflected anywhere else in the app where that object is displayed.

****

Read the marking rubric very carefully to ensure that you are meeting all the requirements.

****

Group Work

You will continue to work with the group from your first assignment.

Do not work completely independently and then try and merge work together at the end.

If using Git to manage the work, then make sure that commits are made to the repository as soon as the feature you are working on has been completed and tested. This will ensure that if anything goes wrong that you won't have lost too much time.

Include an appendix in your report that has a summary log of the contribution of each group member with dates of contribution.

If you are having trouble with a group member participating you should address this **IMMEDIATELY**.

Within your team a work schedule should be set and if a group member fails to meet that schedule on two occassions you should report this to the lecturer immediately.

Non functional groups will be split up and the members will be required to work individually.

Finally

Enjoy! You will learn a lot while creating this prototype. Who knows it could lay the foundation of your next career!

Any questions regarding this assignment or any of the requirements discussed in the specification should be directed to the Assignment 1 (M2) discussion.

Marking Guide

The marking guide/rubric can be found in the assignment submission section.

You must read both this specification and the marking rubric in detail to fully understand the requirements of this assessment.

If anything is unclear, you must post your queries to the discussion forum well in advance of the due date to ensure clarifications can be made in a timely manner.

Please note that whilst we permit you to work on your own idea, this must be within the context of what is being assessed as indicated in this marking guide.

> For example: Asking if you have to implement a database because your app doesn't need one is not a reason for skipping this part of the assessment.

If your idea doesn't conform to the marking guide, you will either need to modify your idea or choose a different idea.

Marking Penalties (20%)

Please note the rubric is a more detailed version of the specification listed here, so you should familiarise yourself with both versions for a complete understanding of the requirements.

All of your Swift code must be object oriented and follow good principles of objected oriented design. Code that does not follow good object oriented design will incur a 20% marking penalty.

Submitting a corrupted or missing file or code that contains compile time or runtime errors will also incur significant marking penalties. Make sure that any files you add to your Xcode projects are located and referenced inside the project and not to an external drive or location.

IMPORTANT:

- A single compilation error will result in a 10% marking penalty.
- Two or more compilation errors will result in a 20% marking penalty.
- Runtime errors will incur a 5% marking penalty for each runtime error encountered up to a maximum of 20%.

****

Read the marking rubric very carefully to ensure that you are meeting all the requirements.

****

Submission Instructions

Where you make use of other references, please cite them in your work.

You should submit:

1. An updated version of your report in pdf format.

2. A zipped version of your entire Xcode project.

This means that you must make two submissions for this assessment. Failure to do this will incur late marking penalties for the number of days it takes to discover the problem and for you to be able to provide the missing files.

You should submit your assignment each time you complete a major section of the assignment.

This will make sure that you are familiar with the process and don't have any last minute issues with your final submission.

These submissions will not be looked at or reviewed unless you post a comment with the submission asking specific questions that you would like help in resolving.

If you ask specific questions we will attempt to answer those questions so that you get feedback on your work prior to submission. The more often you do this, the more feedback you will get.

Please also note that close to the time of final submission we may not have the opportunity to respond due to the amount of queries, so make sure that you seek feedback early.

Ensuring a correct submission

1. You should download your submission after you have submitted to make sure you have submitted correctly.
2. Submitting the wrong file or a corrupted file will result in significant late penalties being applied.
3. It is your responsibility to download what you have submitted and check it carefully.
4. It is highly recommend that you submit your regular progress on your assignment.

****

Please note whilst only one group member needs to complete the submission process, it is the responsiblity of all group members to ensure all the submission guidelines have been met.

This includes checking that the assessment has been submitted by the due date and that the submission is correct.

****

Backups

****

You should make regular backups of your projects and store them in safe locations

1. Multiple submissions to Canvas
2. Privately hosted Git repositories
3. Local backups on your system
(copy and paste your project to back up locations
4. Store on your RMIT google drive. (make sure it is private)

****

Extension and Late submission

All requests for extension should be emailed to the lecturer (rodneyian.cocker@rmit.edu.au).

If you are unable to submit by the due date you **WILL** be required to provide additional documentation such as a medical certificate as per university policies.

A penalty of 10% per day of the total marks will apply for each day late, including weekends.

After five days, your submission will not be marked or reviewed and you will receive zero marks.

Your instructor can only grant extensions of up to 7 days provided the correct form and documentation has been submitted. If you require an extension you must apply prior to the due date by filling out an 'extension of time' request and attaching the requisite documentation.

The form and documentation should be e-mailed to the lecturer.

- http://www1.rmit.edu.au/students/assessment/extension

For extensions of greater than 7 days or for those with EAA provisions organised by the DLU:

- http://www.rmit.edu.au/students/specialconsideration

If you are a student who has EAA provisions organised by the DLU then you must negotiate any extension you may require with the lecturer well in advance of the submission deadline (at least 72 hours in advance of the on-time submission deadline would be ideal).

For further information, please see:

Extensions
https://tinyurl.com/ybojmpfj

Late Submissions
https://tinyurl.com/yc8d2gyj

Plagiarism

All assignments will be checked for plagiarism; any student found to have plagiarised will be subject to disciplinary action. Plagiarism includes

- submitting code that is not your own or submitting text that is not your own
- copying assignments of the students from previous semester - allowing others to copy your work via email, printouts, social media etc. - posting assignment questions (in full or partial) on external technical forums.
- Never upload your assignment work to a public repository such as Git or Bitbucket. You must make sure that any repositories are private.