iPhone Software Engineering
Study Period 3 (2019)
Assignment 2

Due: **Sunday 10th November, 2019 (11:59 P.M.)**

Assignment Type: **Individual / Group (3 maximum)**
WE STRONGLY ADVISE AGAINST ATTEMPTING THIS ASSIGNMENT INDIVIUALLY.

Total Marks: **25 marks (25%)**

Congratulations.

You job now is to complete your app (complete with all the features) to a professional standard that is ready to submit to the App store.

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 Xcode project as your finished app.

If any bugs or points of improvement were discovered in the prototype these should now be fixed prior to implementing the additional features.

You should begin working on the assignment on the day the specification is released.

IMPORTANT:

Your completed app needs to be finished to a professional standard.

All features should now be implemented.

You should have good quality code that is cohesive. This means you should break large classes into smaller classes and large methods into smaller methods. Your code should also be well commented.

No unused code should be left in the solution as commented. If the code is not being used it should be removed.

The completion of your app should focus on:

1. implement design pattern(s) of your choice
   (other than a singleton and in addition to MVC).

2. implement a Cocoa Framework.

3. implement a **LOCAL** persistance option
   (either core data or sqlite).

4. implement remote retrieval of data via a REST based service.

5. implement Unit Tests that test the business logic of your app.

6. demonstrate the use of Adaptive Layout

Complete the build of your app based on your prototype. There should be no 'unused' features in your app, if you have a feature that you will not be implementing in this final version, then the feature should be removed from the app.

It is OK if you change some of your screens and the way the app operates as this is a normal aspect of an iterative approach to software development.

However, the final app should still be 'recognisable' as the completion of your prototype.

Finally

Enjoy! You will learn a lot while creating an app for this assignment. 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 2 discussion.

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 the report should now identify how and where you have met the requirements of this assessment.

It should identify the code files and scenes that have been used to complete each of the requirements.

Finished App

Your finished app should include all of the features you plan to have in your app.

The features should now be fully operational, dynamic and make use of local persistence, networking and accessing hardware features of a device.

Your *client* should be able to open the app on a simulator or device and use it as the finished product. The essential idea behind this implementation that it should represent an app that is ready to be submitted to the app store.

The app must be developed using storyboards and submissions that create the whole app from code will incur a minimum of a 50% marking penalty.

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.

Additional Requirements

The finished app must include some advanced features as indicated in this specification document and the use of Unit Tests.

The UITests from your prototype should be kept up to date and still function correctly.

Adaptive Layout must also be demonstrated.

The full requirements for this assessment are detailed in this specification and in the marking rubric.

****

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

****

Advanced Features (Networking & Persistance)

The app must now fully implement the advanced features that were mocked up in the prototype.

These advanced features must include:

- Networking (retrieving data from a REST based web service)
- Persistence (Core Data)

REST Based API Resources

You must now implement the REST service in your app that you identified in the earlier assessments.

Please make sure that the API is a REST based web service and not utilising a framework.

You can use the links provided here as a starting place for finding a suitable API. If you cannot find one at these sites you will need to google search using your topic and searching for REST API and/or Web API.

The API and the data returned should be applicable to your app and its data requirements.

https://market.mashape.com/ijameelkhan/instatunes

http://www.programmableweb.com/apis/directory

https://github.com/toddmotto/public-apis

Testing Your API

If you have not worked with a REST based API before here are some guidelines to ensure you are using the right kind of API.

- The API you have chosen, must explicitly state that it accepts REST requests (URL's).
- Typically you need to sign up to the service and they will provide you with a unique API key that must be included with each URL Request.
- You should be able to copy the URL (Request) and paste it into your browser and it should display a response in the form of structured data (usually a JSON format).
- If the instructions for your API ask you to download and install a 3rd party framework or use cocoapods, then it is not a suitable API for this aspect of the assignment specification.

If you have identified an API and you are not sure if it is REST based, then check with your lecturer or tutor.

Example of a REST Based API

URL Request: **http://api.androidhive.info/contacts/**
(You should be able to copy and paste a request like this into the address bar of your browser and it should return a valid response i.e. some data.

Note: This one does not contain an API key and is shown here for simplicity.)

Example Response:

```
{
      "contacts": [
      {

            "id": "c200",                                          Dictionary: i.e. key/value
            "name": "Ravi Tamada",
            "emails: ["homeemail": "ravi@home.com",                Array of dictionaries
                        ["workemail": "ravi@work.com"],
            "address": "xx-xx-xxxx,x - street, x - country",
            "gender" : "male",
            "phone": {
                        "mobile": "+91 0000000000",
                        "home": "00 000000",                       Embeded Dictionary:
                        "office": "00 000000"
            }
      }
}
```

Persistence (local)

You should now implement your database as either Core Data or Sqlite. Please note other database solutions such as Firebase, Realm or other cloud based solutions that manage remote and local data are not permitted to satisfy this requirement.

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

Unit Tests & UITests

Any UITests should be updated to ensure that they are still operating correctly.

You should also implement unit testing on your model classes.

Auto Layout & Adaptive Layout

The final implementation must ensure that all scenes operate in Portrait and Landscape orientation via the use of auto layout.

In addition it must demonstrate the use of Adaptive Layout through the use of size classes.

Adaptive layout needs to demonstrate a change in layout between portrait and landscape orientations.

Restrictions

The idea for your app should not rely on the following:

1. Social Media type applications that require sharing of data between a user base.
2. Apps that attempt to work interactively with other apps on the device such as the calendar.
3. Features that can only be implemented with the use of a Apple Developer certificate i.e. push notifications.
4. Gaming applications are generally too complex and require advanced knowledge to be considered a fit application in this course.
5. App that require the implementation of e-commerce features or payment systems.
6. Private API's that are unavailable during the assessment process.
7. Any app that by its nature makes it difficult to assess due to restricted access.
8. The implementation of the design will occur in Assignments 1 (M2) & Assignment 2 and must be developed using Xcode 9.0.x, Swift 4.x, Storyboards.
9. Do not plan on using Firebase, Realm or any other cloud based solution for your persistence requirement.

Whilst we give you the freedom to work on your own idea, it must still meet the requirements in this specification to ensure that the learning outcomes of the course and the assessment requirements are met.

You should consult and review these guidelines and ensure that your app conforms with the recommendations in relation to:

> Safety, Performance, Business, Design, and Legal
>
> https://developer.apple.com/app-store/review/guidelines/

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.

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

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

You should submit the following in a single archive file (zip), please do not use other archive formats:

1. your entire XCode project including any dependencies so your app can be run without the assessor having to re-configure your app or download and install dependencies.

2. a pdf report that meets the requirements specified above should be included in the root of your project so it is contained in the zip archive that is uploaded to Blackboard.

You may submit your assignment more than once. The last submission will be the one used for assessment.

We cannot use multiple submissions for the assessment only the latest version will be assessed.

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.

YOU MUST download and check your final submission after you have submitted to ensure that it uploaded correctly and is the correct version.

Incorrect submissions will incur significant late marking penalties that will require you to re-submit when the error has been discovered.

Extension and Late submission

All requests for an 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.

Extenisons will not be granted without supporting documentation.

A penalty of 10% per day of the total marks will apply for each day late, including weekend. After five days, you will receive zero marks and the submission will not be marked.

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 the RMIT website.

You can google search:

- RMIT Extension of Time
  (current link: **https://tinyurl.com/y8n8sydu**)

  OR

- RMIT Special Consideration
  (current link: **https://tinyurl.com/ybpuywnl**)

Plagiarism

All assignments will be checked for plagiarism. Any student found to have plagiarised is 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

RMIT electronic submission of work for assessment

I declare that in submitting all work for this assessment I have read, understood and agree to the content and expectations of the Assessment declaration at the following URL:

https://tinyurl.com/yarcuetd