6COSSC004W Mobile Native Development Coursework 2 – iPad Application (2020/21)	
Module leader	PHILIP TRWOGA
Unit	Coursework 2
Weighting:	**6**0%
Qualifying mark	30%
Description	**Spending Management Tool for iPad**
Learning Outcomes Covered in this Assignment:	This assignment contributes towards the following Learning Outcomes (LOs): LO1 understand language features and programming practice required for native development LO2 apply industry standard tools for design and development LO3 communicate and defend work by both written and oral means
Handed Out:	15/03/2021
Due Date	**17/05/2021 by 1pm**
Expected deliverables	Submit on Blackboard a zip file containing: **Complete XCode 12.X solution - <u>must</u> be in Swift 5.x**
Method of Submission:	Electronic submission on BB via a provided link close to the submission time. The file you upload should have the following naming format: E.g. **6COSC004W**_StudentNumber_firstName_lastName.zip
Type of Feedback and Due Date:	Formative feedback will be provided during tutorial sessions. Verbal feedback on the submitted CW will be provided during an online viva. Students are encouraged to record this feedback at this time. Feedback and marks are due by the TBD. Feedback shall also be given on the Blackboard. Note: All marks will remain provisional until formally agreed by an Assessment Board.

Assessment regulations

See the Assessment guidelines https://www.westminster.ac.uk/current-students/guides-and-policies/assessment-guidelines

for a clarification of how you are assessed, penalties and late submissions, **what constitutes plagiarism etc.**

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website http://www.westminster.ac.uk/study/current-students/resources/academic-regulations

Note: By submitting the work through Blackboard you are acknowledging that this is solely your own work. Any code which is not created by you MUST be clearly commented as such. Any code discovered to not have been created by you will mean that the work will be submitted to academic standards for a potential assessment offence, which may result in a zero mark in the component or whole module.

6COSC004W Native Programming Coursework 2 Specification - Spending Management Tool – Individual coursework

Introduction

You are to create a master-detail application for iPad that utilises core data. You are free to choose the style and look and feel of the application, but the application must conform to the functional and non-functional requirements as stated in this specification.

General Functionality

You are to create a spending planning tool that allows the user to track their **expenses** against budget for different **categories** of expenses, such as travel, food, socialising, sports and hobbies etc.
The app shall use the **Master-Detail (Split-view controller)** or similar application pattern with the expense category details in the master view's **UITableView** and the corresponding 'Detail' view shall give a summary of expense category as well as list expenses against a selected category – see Figure 1. The user shall be able to add new expense categories to the master view (see Figure 2). The detail view shall give a graphical indication of expenses versus budget for the selected category (see Figure 1). The detail view shall also allow the user to add **add/delete/edit** expenses (see Figure 3). All the data entered shall be persistently stored using core data.

Entities

The **Category** entity consists of:

name
monthly budget in £
notes
colour – this sets the background colour of the table cell

Once a category is entered the user can then enter one or more **expenses** Expense details are as below:

The **Expense** entity consists of:

amount
date
occurrence – (can be daily, weekly or monthly or one off)
notes – some brief notes about the expense.
expense due reminder flag – if set true this set the expense due date in the Calendar app - optional for the user to set if the expense is in the future.

User Requirements

R1: The software shall allow the user to create a new category.
R2: The software shall allow the user to create one or more expenses for a category.
R3: The user shall allow the user to be able to edit/delete a category.
R4. The user shall allow the user to be able to edit/delete an expense.
R5: The software shall allow the user to set a reminder for an expense of an that occurs in the future in the calendar app.
R6: The software shall allow the user to sort the categories alphabetically (A-Z).

System Requirements

RS1: The software shall place the due date of an expense[1] in the Calendar app (if set by the user).
RS2: The software shall give a graphical[2] indication of the current expense as proportion of overall budget for a given category.
RS3: The software shall give a graphical indication of the total expenses as a function of the overall budget for a selected category[3].
RS4 The software shall show the four highest expenses for a given category as a pie chart.
RS5 The software shall show the remaining expenses in a pie chart (so those not listed in RS4) for a given category as a single segment.
RS6 The software shall show the total budget, Amount Spent, and Amount Remaining for a given category.
RS7 By default, the software shall sort the categories by most often selected.

Non Functional Requirements

NF1: The software shall adhere to Apple design guidelines as regards fonts and layout and general usability.
NF2: All user data shall be persistent using core data.
NF3: The software shall be intuitive and simple to use.

[1] This place an event in the iOS Calendar app.
[2] Graphical means the use of custom graphics. *UIProgressViews* can be used but for ½ marks of that available.
[3] This shall be shown a custom view on the detail view – see RS4 and RS5.

Indicative Mark Scheme

R1: The software shall allow the user to create a new category. 10%
R2: The software shall allow the user to create one or more expenses for a category. – 10%
R3: The user shall allow the user to be able to edit/delete a category. – 5%
R4. The user shall allow the user to be able to edit/delete an expense. – 5%
R5: The software shall allow the user to set a reminder for an expense of an that occurs in the future in the calendar app. – 5%
R6: The software shall allow the user to sort the categories alphabetically (A-Z). – 5%

RS1: The software shall place the due date of an expense[4] in the Calendar app (if set by the user). – 10%
RS2: The software shall give a graphical[5] indication of the current expense as proportion of overall budget for a given category. – 5%
RS3: The software shall give a graphical indication of the total expenses as a function of the overall budget for a selected category[6]. – 20%
RS4: The software shall show the four highest expenses for a given category as a pie chart[7]. – 0%
RS5: The software shall show the remaining expenses in a pie chart (so those not listed in RS4) for a given category as a single segment[7]. – 0%
RS6: The software shall show the total budget, Amount Spent, and Amount Remaining for a given category. – 5%
RS7: By default, the software shall sort the categories by most often selected. – 10%

NF2: All user data shall be persistent using core data - 10%

Note that each requirement in the indicative marks section has 4 marking categories (also see note on Usability below):

1) Complete and correct – full marks (as shown above)
2) Minor defect – approx. 70% of full marks (as shown above)
3) Major defect – approx. 10-50% of full marks (as shown above)
4) Not done or some non-functioning code – 0-10% of full marks (as shown above)

Notes:

All work must be demonstrated in the viva where you will receive feedback and an indicative mark. **The viva is part of the coursework and so is obligatory.**

[4] This place an event in the iOS Calendar app.
[5] Graphical means the use of custom graphics. *UIProgressViews* can be used but for ½ marks of that available.
[6] This shall be shown a custom view on the detail view – see RS4 and RS5.
[7] Marks for this task are included in RS3.

Work marked without a viva shall receive between 0-30% maximum.

Usability: Apps that are fully or partially functioning can still be marked down for poor usability. (Poor UI – difficult to use and inconsistent – 15% max, UI has major issues, complex to use, non-intuitive – 10% max, some minor usability issues – 5% max) This is at the discretion and judgment of the tutor marking the work. See NF1 and NF5.

This is individual work – do not collaborate or share any code.

End of document

Appendix A – Example Design

Note: you are free to design this app how you wish, and these screenshots are for guidance only and you strongly encouraged to produce your own design

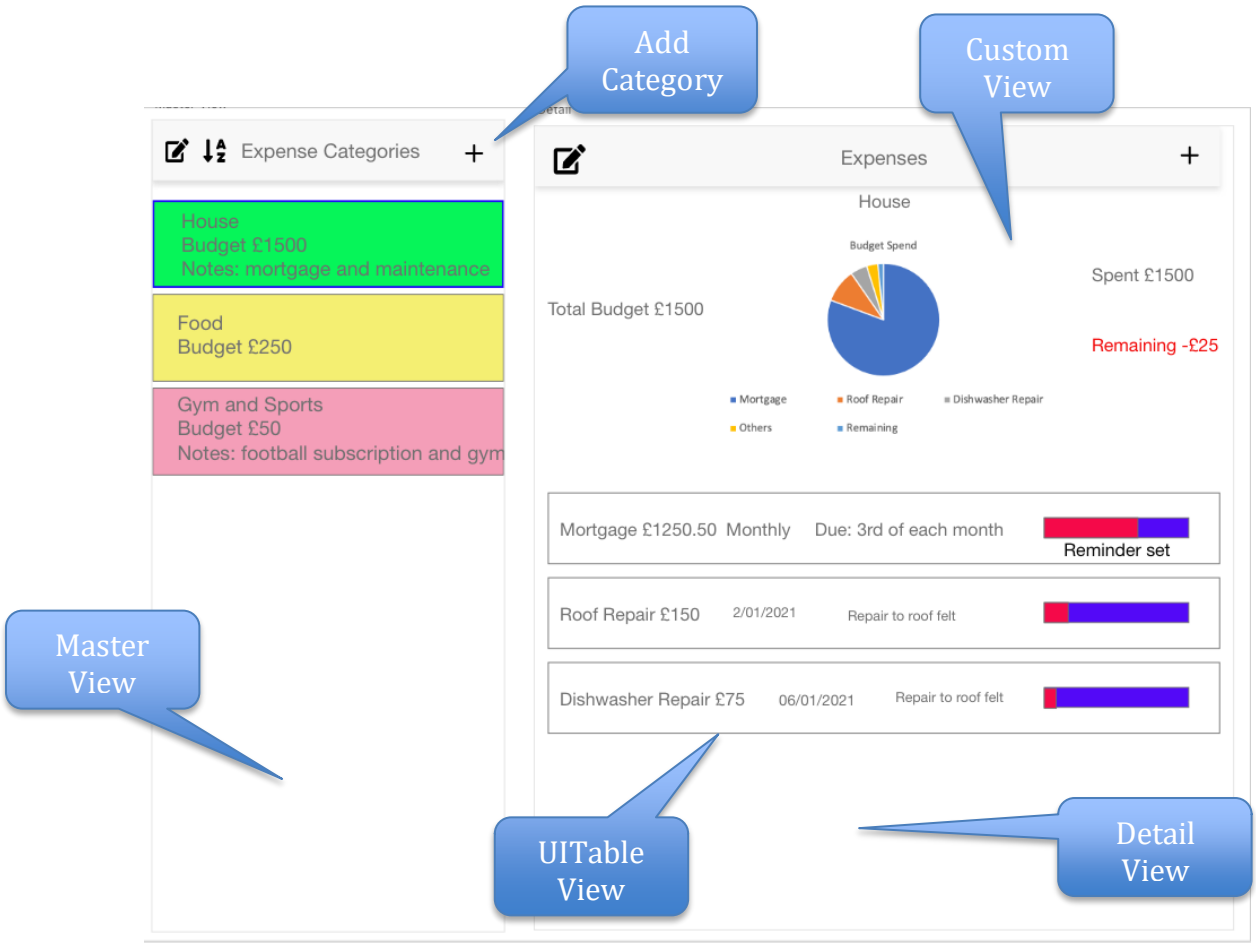

Figure 1 – General layout – note you are free to implement your own UI design

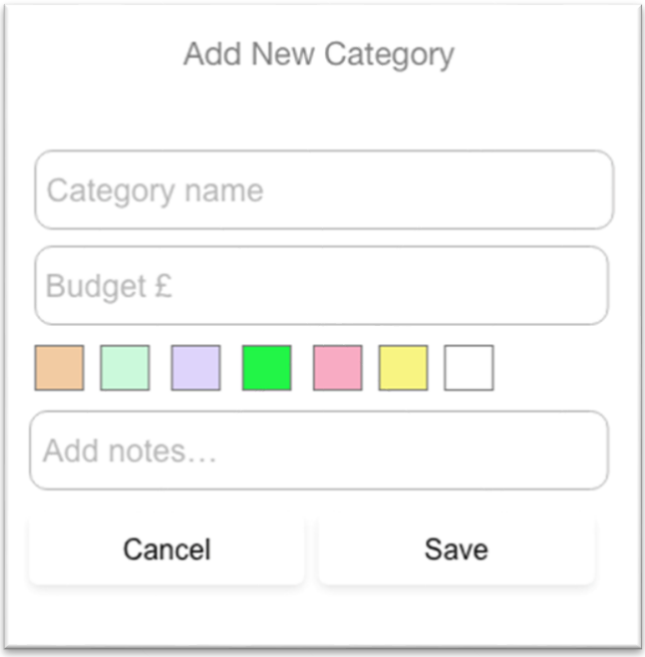

Figure 2 Add new Category pop over – note that colour squares are buttons

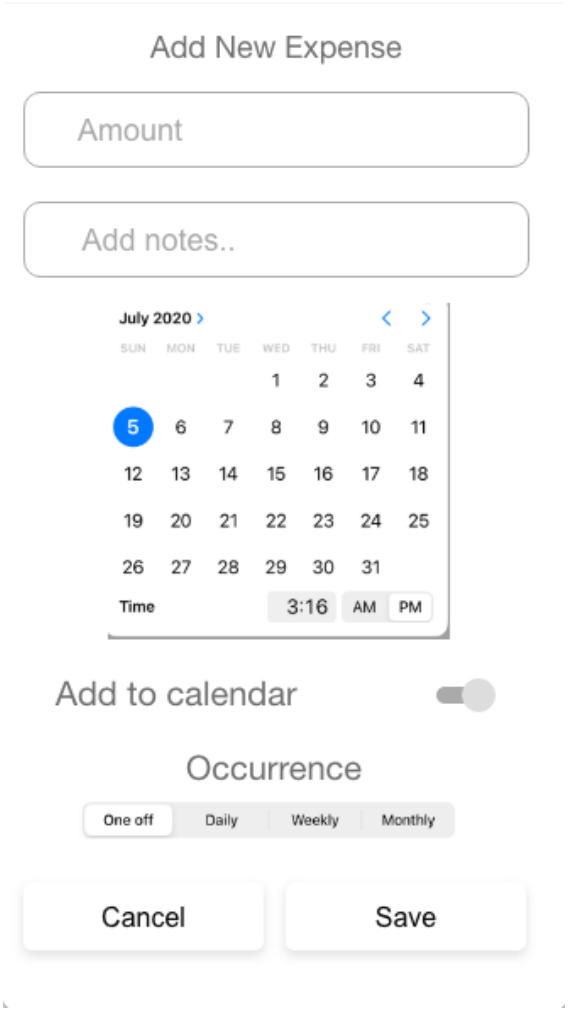

Figure 3 Add new expense pop over

Grade Descriptors

Marking Grade Descriptors - for guidance only

80-100: An outstanding piece of work: All assessment criteria have been met at an exceptionally high standard

- Displays exceptional initiative, creativity, sophistication and originality.
- Demonstrates originality and rigour in technique
- Demonstrates complete and fit for purpose design and analysis – meet all usual software quality criteria
- Demonstrates substantial independent research
- Communicates complexity clearly and succinctly with excellent standard of presentation and understanding

70-80: An excellent piece of work: All assessment criteria have been met at a high standard

- Demonstrates a sophisticated application and originality
- Draws on a range of techniques and frameworks
- Demonstrates creativity
- Provides a robust solution (reliable, expandable, understandable, maintainable)
- Provides a high-quality for for purpose analysis and design
- Demonstrates substantial independent research
- Communicates ideas clearly and succinctly with good standard of presentation

60-69: A good piece of work: Nearly all assessment criteria have been met at a good standard with just a few defects

- Demonstrates breadth and/or depth of understanding across frameworks and language
- Demonstrates research and critical use of resources.
- Communicates ideas clearly with a good understanding

50-59: A reasonable piece of work: Nearly all assessment criteria have been met at a good standard with a number of small defects.

- Demonstrates breadth and/or depth of understanding across frameworks and the language
- Demonstrates some research and critical use of resources
- Communicates ideas clearly with a good understanding but with some deficiencies in knowledge

40-49: An adequate piece of work: Assessment criteria have mostly been met but with minor and major defects.

- Demonstrates understanding of appropriate range of framework concepts and practical approaches
- A basic application with some serious defects such as crashes

- No demonstrable original thinking in design
- Only partial understanding demonstrated

30-39 FAIL: An inadequate piece of work: Relevant assessment tasks are not met.

- Provides clear limitations in the practice and approach to development
- Demonstrates lack of design and analysis
- Uses a narrow range of techniques/frameworks to support development
- Communication is unclear with significant weaknesses in explanation of code
- Significant defects

0-29 FAIL: Poor piece of work: Most of the assessment tasks have not been met.

- Demonstrates poor understanding of key concepts and techniques
- Shows significant weaknesses and omissions in completing the tasks
- Omits key analysis/design
- Communication is unclear with significant weaknesses in understanding

End of document