Note

My app was developed with the iPhone 11 emulator.

Introduction

Keeping grades consolidated is not easy when there is not on solution that ticks all the boxes. This leads to tutors using all sorts of different methods for keeping track of attendance, whether that is the convenience of just writing on paper but the downside of having to be digitised later or using mylo with grade being entered electronically from the start this has the big benefits of consistent formatting and having all the grades in a central location, but with its clunky UI it is not worth for some. So, how do we take grades in a way that is convenient, but also consistent and consolidated? Well, an app of course!

For this app to be a viable option for tutors they need to be able to enter and remove students, quickly and easily mark the students, be able to change what type of mark they are storing in each week and be able to share this data to other applications. So, in this report I will detail my implementation of just such an app on the iOS platform.

Differences from Prototype

Overall, my final product is very close to my prototype, the layout was kept basically the same with a bit of an apple twist to make iOS users feel more at home.

The biggest difference between my prototype and the finished app was that I added a modal to the landing screen to let the user set the start of the semester for the current week button and label to use. This was honestly just an oversight in the original prototype. Another difference between my prototype and the final application is that in the StudentDetailView and WeekDetailView allow the grades to be edited straight from the respective view instead of having a different View for it. I did this mainly for efficiency as it means multiple grades can be changed very quickly instead of having to wait for a different view to load and then having to double back to change another grade. Another big difference is that the week configuration changer has been made much simpler due to a clarification in the spec. In my prototype weeks could have multiple different types of marking schemes in the one week, whereas in the final implementation it was specified that each week only need one type of mark, so I just made it simpler and made it a popover modal.

A slightly smaller changes made was moving all the buttons that were below the tables in the prototype to the navigation bar. I did this as it felt more appropriate for an iOS app to have them up there instead of shrinking the table view potentially leading to a bit more scrolling. I also moved the image in the student details page to the side of the student's name and such to save a fair bit of vertical space while still keeping the image a decent size.

Usability Goals and Design Principles

My app was built with efficiency in mind. This is best shown in the StudentDetailsView and WeekDetailsView where all the grades are immediately present and changeable as opposed to having to go to a separate page for each grade so multiple grades can be entered rapidly both for the one student in StudentDetailsView or for multiple students in WeekDetailsView. This does reduce the forgiveness of the app though as once a grade has been changed there is no way to undo this save for just re-entering the old grade, and it could be quite easy to accidently enter a grade. This also highlights one of the biggest flaws of my app, the lack of feedback. When updating grades there is no feedback to tell the user that the grade has been successfully updated. I considered solving this by having a popup every time but that would very quickly destroy the efficiency these views and also just be plain annoying. This could potentially be improved with a 3rd party system that provides something similar to Androids toasts, or maybe even a first party system like this that I couldn't find. To get a bit more positive again I think another great display of efficiency is in the go to current week button on the ViewController main screen. This optimises for what is probably the most common used case of a tutor wanting to enter a grade for a student in the current week, so the app keeps track of how many weeks it has been since the start of semester (it does not factor in holidays or mid semester break unfortunately).

I also tried to make the app more visible and add in some mapping (MapInTime, 2013) with the overall style of the app. All the of buttons in the app are clearly label and can be seen straight away without cluttering up the UI.

I also used default iOS icons for all navigatiojn bar functions to try to line up with most users' mental models of how iOS apps should look and work. The TabView holding both the StudentTable and WeekTable give the users options that are more efficient for certain used cases. well as make the user aware about the two separate browsing methods.

I added constraints to the app in the form of the tables in StudentDetailsView and WeekDetailsView only display the type of grade that is configured for that week, and even more so for the score weeks as they make sure the user can only enter numbers and makes sure the number is less than or equal to the max score, changing a grade that is too big to the maximum score. Similar constraints are also present in AddStudentView and when changing a student's details in the StudentDetailsView where the student ID has to be a 6-digit number and is also checked to make sure it is unique with clear feedback if one of these criteria are not hit.

I also added some forgiveness to swiping to delete a user on the StudentUITableView in the form of a popup that makes sure the user is certain they want to delete this student by displaying the name for the student to make sure they are deleting the right student if they even meant to delete a student. Similarly to this, both the CurrentWeekConfigView, WeekConfigView and AddStudentView can be dismissed by swiping them away is they are accidently selected.

To help with the visibility of the app and learnability for users who are used to iOS I tried to stick to Apple's human interface guidelines (Apple, 2019) such as consistency on iOS in general by using the standard icons for the navbar actions on most screens, direct manipulation with swiping to delete users.

As previously mentioned, I think the StudentDetailsView and WeekDetailsViews highlight one to the biggest flaws in my app being feedback, I think these two views also suffer badly from a lack of forgiveness when changing grades. On top of not giving feedback that a grade has changed, the user has no way to revert a grade change aside from just updating the grade again back to what it was before. This was another one of the trade offs I made for efficiency that I feel is a little more justified than the lack of feedback, but at the same time could cause problems for users. Another potential problem I can foresee is users potentially not noticing features, the most obvious one in my eyes is not knowing you can swipe to delete students. To fix this I would probably implement a first time launch tutorial to guide the user through how to use the app.

Code Documentation

Grades – A structure to hold a student's grades.

Student = A structure to hold a student's details; first name, last name, student ID, and name of their image if they have one.

Cell's folder – This folder contains all the cells used in the table views; StudentCell, WeekCell, AttendanceCell, CheckpointsCell, FToACell, NNToHDCell, ScoreCell. The ScoreCell stands out as it has an extension to the UIView class so that it can loop through its parent views until it find a view it can display an alert from.

ViewController – This is the starting screen of my app that gets all initial data like the students and week types.

CurrentWeekConfigViewController – This modal allows the used to change what date the semester starts on to change which week the ViewController's current week button will take you to.

TabBarViewController – This view displays the StudentUITableView controller and WeekUITableViewController in two separate tabs.

StudentUITableViewController – This view displays a list of all students that can be clicked to see the students details and grades. It contains the numberOfRowsInSection function to tell the tableview how many rows to display, the numberOfSections to tell the tableview how many sections of the table to render and cellForRowAt to setup, return the appropriate cell for the table row and trailingSwipeActionsConfigurationForRowAt to setup swiping to delete students.

WeekUITableView – This view displays a list of the 12 weeks in the semester to click to see all students grade for that week. It has the same functions as the StudentUITableView minus the swipe to delete.

AddStudentViewController – This modal pops up from a nav bar button in the student view page and allows the user to add a student with or without an image.

StudentDetailsViewController – This view displays and allows the user to change a students details and grades. This view has all the same table functions as the previous two table views as well as some functions to update grades, load images, display alerts and update details.

WeekDetailsViewController – This view displays the grade for every student in the selected week. It has the same table view functions as mentioned previously, but it also contains the GradeUpdateDelegate protocol that defines that all GradeUpdateDelegates need to implement an updateGrade function (shocker!). This delegate is used in both this view and the StudentDetailsView to let the cells in their tableviews call the parent views updateGrade function so they can update grade averages.

WeekConfigViewController – This modal pops up from the WeekDetailsView and lets you change the grade type for the week it was opened from.

Nearly all of my code was sourced from the lectures, tutorial content or Apples own documentation. Some stand out exceptions are how to setup table views in a UIViewController https://guides.codepath.com/ios/Table-View-Quickstart and swiping to delete students https://programmingwithswift.com/uitableviewcell-swipe-actions-with-swift/

Conclusion

To round things out, I think I have taken all the good aspects of my prototype and refined them into a feature rich and efficient iOS app. Like my prototype my iOS app was an efficient first design, making sure the most common tasks can be done with as little clicks as possible, while still staying intuitive. While I am happy with the outcome, I learnt how much of a balancing act it can be to try and make the app as easy to understand as possible while still being quick and minimal. There was only a feature or two I missed, and I am fairly confidant in the stability of the app, so overall, I am very happy with what I have managed.

References

1. MapInTime. (2013). Don Norman's Design Principles. [online] Available at: https://williamgrimes12.wordpress.com/2013/01/22/don-normans-design-principles/ [Accessed 25 Apr. 2021].
2. Apple (2019). Themes - iOS - Human Interface Guidelines - Apple Developer. [online] Apple.com. Available at: https://developer.apple.com/design/human-interface-guidelines/ios/overview/themes/.