Hirona Oku

1. Debug tool (simulator)

This application is developed and debugged under the environment of Chrome(web) using Apple M1 chip.

2. Introduction

The Tutorial Marks App has been designed and created for use by tutors at the University. It has three main functions: marking grades, managing student information, and checking grades. Grade Marks allows tutors to choose one mark type for each week from a range of grade styles (HD+, HD, DN, CR, PP, NN or A, B, C, D, E, F), numerical scores, attendance, and multiple check boxes. The student information management function assists tutors to manage student information, including each student's name, student number and grades, and to support students by allowing them to check and share their grades. Finally, the Grade Check function helps students to improve their understanding of a subject by providing a weekly summary showing the average score of all students for each week.

The design of this application follows the Usability Goals of Efficient, Failure-resistant, Learnable on first use, Forgiving, as well as Don Norman's six Design Principles; Visibility, Affordance, feedback, mapping, constrains. The development process began with a thorough reading of the requirements, prototyping, usability testing, android and iOS development, and then a review of the design with the results to determine the balance between the usability goals. As a result of considering the balance, Efficient was chosen as the most important challenge for this application used by tutor. This document begins with a presentation of the differences between this and previous assignments, followed by a description of the Usability Goals and Design Principles. Then, it will describe the class code of the application and illustrates it with a class diagram.

3. Differences to Assignment of Prototype and Android, iOS

The structure of the screen has been significantly changed from the prototype. This was achieved with a ListView, a drawer for week selection, tabs, dialogs, and partial view switching. The elimination of screen transitions would have greatly improved the consistency and efficiency of this application. In this section, it will explain the differences about on the Main Page, Marking Page and Student Detail Page, which have changed significantly since the prototype.

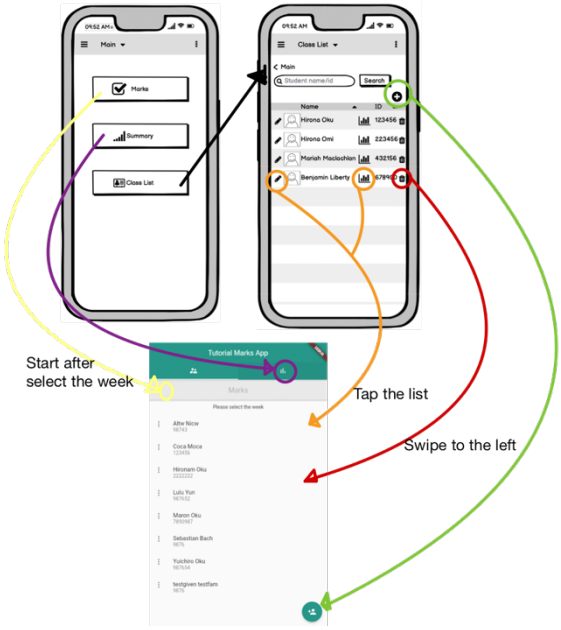


Figure 1, New Main page with the functionality of two pages . Above: Prototype (left: Menu, right: Class List), below: New Main page

3.1. Main Page

On the Main Page the menu screen has been removed and a new single page with the functionality of two pages; Menu and Class List has been successfully created. Figure 1 shows how each feature of the prototype menu and class list has been implemented on a single page. The page is even simpler than the prototype class list page, even though it combines two pages of functionality into one. Because, by making effective use of finger gestures such as tap and swipe, visual tabs and icons, and by changing the design from having all functions as icons and buttons in the prototype to showing the necessary icons.

This simplicity throughout will allow users to see at a glance what can be done and avoid getting lost among the pages to find a feature.

3.2. Marking page

There is no change in the way the marking pages are marked; marking all students on a weekly basis, however, there is a difference in the way the marking schema is created. The left screen in Figure 2 shows the prototype and the checkbox settings page that are displayed after selecting the mark type as checkbox. This screen requests the user to set the number of checkboxes, after which the user is taken to the marking page. On the marking page, the number of checkboxes set will be displayed. In this flutter design, [-][+] buttons have been adopted at the top of the marking page. This allows the user to dynamically change the number of checkboxes, for example pressing [+] will increase the number of checkboxes in the marking schema by one. The prototype method required a separate page to set the number of checkboxes, instead the new method eliminates the need for a configuration screen to set the number of checkboxes and can be completed on a single page.

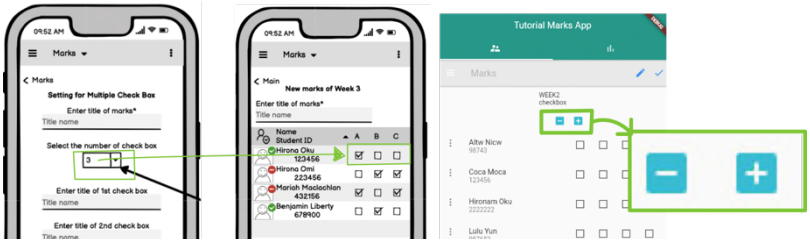


Figure 2 Different ways to change the number of checkboxes. (left, middle: marking from the prototype checkbox settings, right: changing the number of checkboxes in new marking)

3.3. Student Detail Page

In the same way, the student detail page combines the prototype student information update page and the grade confirmation page into a single page. This allows users to view and update their student details, check their grade results, and share them all in one page (Figure3).

As a result, many pages are structured differently, and the application is simple enough and easy to understand compared to the android and iOS versions.

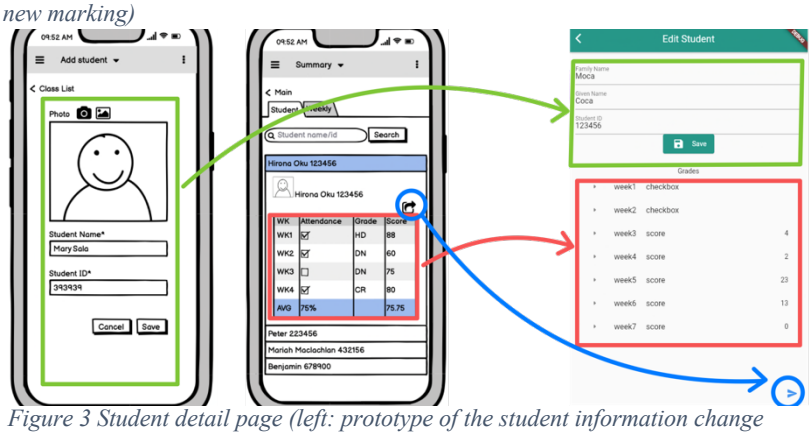


Figure 3 Student detail page (left: prototype of the student information change page, middle: prototype of the student grade confirmation page, right: student detail page in flutter)

4. Usability Goals and Design Principles Self-Critique

4.1. Usability Goals

According to Quesenbery (2004), it is also desirable that the balance of usability is equal in all aspects, however, the importance of each depends on the user and the purpose of using the application. Therefore, considering the users of this application, the most important usability goals were Efficient, followed by Failure-resistant and Learnable on first use (Figure 4).

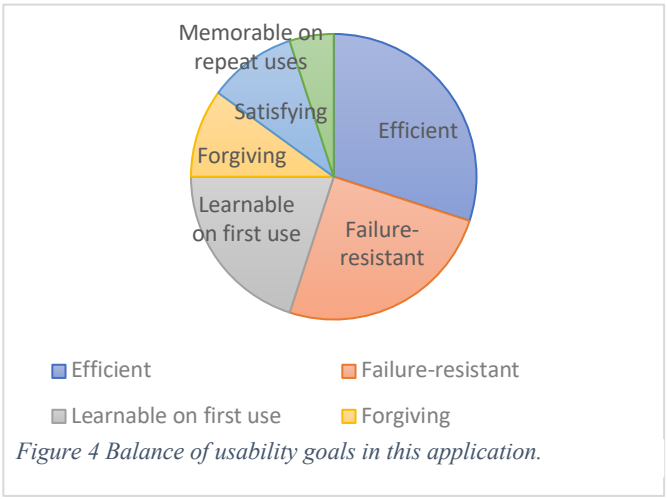


Figure 4 Balance of usability goals in this application.

In this section, it will introduce each perspective of usability goals; Efficient, Failure-resistant, Learnable on first use and Forgiving, in order to importance of balance in this application.

4.1.1. Efficient

Efficiency is important to tutor who have many responsibilities and a large number of students. Therefore, in this application, "Efficient" is of paramount importance. Efficient is defined as the amount of effort spent by the user to achieve a goal, and the system should enable efficient use so that once the user has learned about it, they can be highly productive (Nielsen 1994). The application makes it possible to complete the scores of all students on a single page for each week, and also to perform detailed settings for each score without having to go to another page (Figure 2). Moreover, the weekly summary can be viewed with a single click, no matter which screen the user has open.

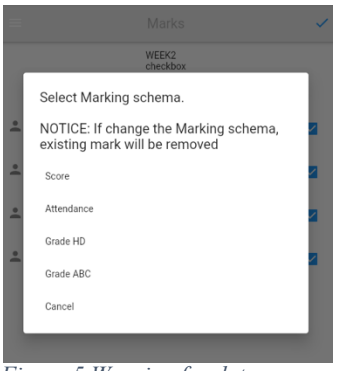


Figure 5 Warning for data removal

4.1.2. Failure-resistant

The loss of student data is a fatal and important issue for universities. Failure-resistant requires that errors should be less likely to occur (Nielsen 1994). In order to prevent the loss of data due to user accidents, this application always displays a warning when changing the mark type in the same week. The warning clearly states that the data will be removed and prevents the user from making an error (Figure 5). In addition, each time a user makes a mark, the data is saved in the database, thus ensuring that no data is lost due to power failure, even during the marking process. In addition, if the user's input is incorrect and the data cannot be saved in the database, a temporary message like toast will be displayed to inform the user of the error.

4.1.3. Learnable on first use

Learnable on first use is defined as a system that can be easily learned so that users can start working with it immediately (Nielsen 1994). There are many tutors at the University, each with a different preferred point of view. In this application we have tried to remove unnecessary pages and to make it more intuitive by introducing swipes, taps and icons(Figure 1 and Figure 2). The result is an application that feels like a familiar application and can be used by new users.

4.1.4. Forgiving

If the failure-resistance of an application is perfect, it might be assumed that forgiving is not necessary. However, Quesenbery (2004) argues that failure-resistance alone is not enough to deal with user error. Although not yet being implemented in this application, the implementation of an undo button and database rollback in the event that a student's information or grades are deleted would increase forgiving and allow users to use this application with assurance.

4.2. Design Principles

Norman(2013) states that the design does not need to be complex, it should be user-centred and ease of use is important. In this section it will explain with examples of the applications created and the Design Principles presented by Norman.

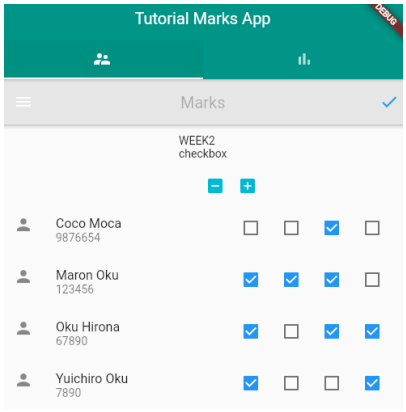


Figure 6 Visibility of this application

4.2.1. Visibility and affordance

Norman (2013) defines visibility as the ability to see with one eye what actions the user can take there. And by recognising affordances, the user can know what actions are possible without labels or instructions(Norman 2013). The visibility of this application is created through the use of simple icons and without the use of unnecessary words. Therefore, the icons show what the user can do. For example, Figure 6 shows the marking schema for a multiple checkbox. Icons that cannot be operated are hidden. Most of what is shown is a simple arrangement of icons and checkboxes. The icons are not labelled; however, the user will be able to recognise the affordances.

However, there is room for improvement in the visibility of this application. For example, the visibility of the result page of the multiple checkboxes is not satisfactory, as Figure 7 shows. To improve the visibility of this, the checkboxes should be disabled.

Figure 7Results pages that require increased visibility.

4.2.2. Feedback

Norman (2013) states that feedback is "communicating the results of an action". And the feedback must be immediate, because the later it is, the more likely users are to give up. The application's feedback checks the input values every time the user enters a number and will immediately display an error message if the user enters an incorrect number (Figure 8). This is much more efficient than checking all the input values at once and displaying errors. And this encourages correct input from the outset, preventing the user from having to do it twice. In order to further improve the feedback of this application, it is necessary to display a notice not only after the error, but also after the completion of SAVE and editing.

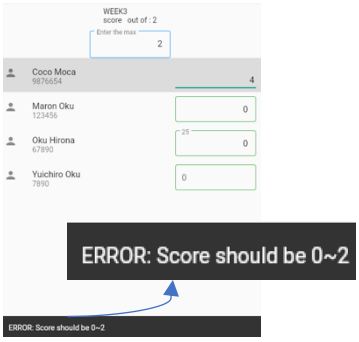


Figure 8 Display error as feedback

4.2.3. Mapping

Norman (2013) also describes natural mapping, which is concerned with the order in which people move their gaze when taking action. In this application, it was designed to assume that the user would look clockwise from the top when interacting with the device. Therefore, the mapping was designed to show the mark from the top left, and the summary, edit and finish buttons from the top right.

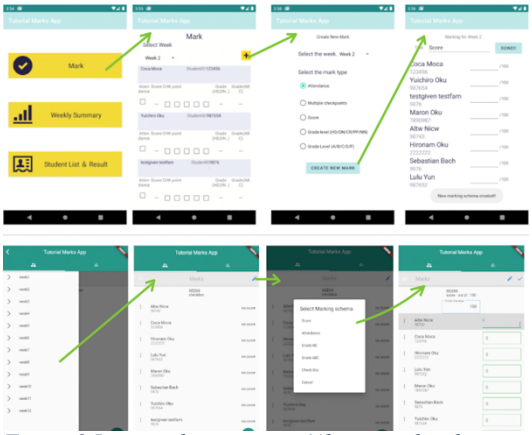


Figure 9 Improved consistency(Above: android, below: cross platform)

4.2.4. Constraints

Constraints are also important for marking. Too many buttons can be confusing for the user. Therefore, the application focuses on constraints in every detail, such as using dropdown buttons to mark grades, allowing the user to select HD or DN, and displaying a numeric-only keyboard to enter scores.

4.2.5. Consistency

The importance of consistency is that users will remember a system once they are familiar with it, and once they are familiar with it, they will continue to use it and be reluctant to change it. It also prevents confusion about the user's operation (Norman 2013). The application is designed around a large ListView, with few transitions between screens and a similar ListView on different screens, so it is consistent and users are unlikely to get lost.

5. Code Documentation

5.1. main.dart

Class Name	main
Description	Run and call MyApp at startup.
Resources	

Class Name	MyApp
Description	StatelessWidget Called by main at startup to build the base page.
Resources	**“Work with tabs” - Flutter(n.d.)** https://flutter.dev/docs/cookbook/design/tabs I followed this tutorial to create a TabController to display the "marks" and "weekly" tabs at the top of the application screen.

Class Name	MarkingPage
Description	StatefulWidget WidgetClass for creating a marking scheme
Resources	

Class Name	_MyMarkingPageState
Description	State<MarkingPage> This class is used to select the marking schema mark type, mark and display the result
Resources	**“Add a Drawer to a screen” - Flutter(n.d.)** https://flutter.dev/docs/cookbook/design/drawer I have followed the instructions on this page to add a drawer to the Scaffold to allow the user to choose what week it is. **“SimpleDialogOption class” – Flutter.dev(2021)** https://api.flutter.dev/flutter/material/SimpleDialogOption-class.html I have followed this instruction on this page to add dialog alert to choose select marking type. Created function (void selectMarking), then implemented this SimpleDialogOption. **“How to create Toast in Flutter” – stack overflow(2017)** https://stackoverflow.com/questions/45948168/how-to-create-toast-in-flutter I have followed this answer to display toast like message. “**DropdownButton<T> class” - Flutter(n.d.)** https://api.flutter.dev/flutter/material/DropdownButton-class.html I have followed the instructions on this page to add a dropdown button to the grade marking schema to allow the user to select grade HD, DN, CR…. **“Checkbox class” – Flutter.dev(2021)** https://api.flutter.dev/flutter/material/Checkbox-class.html I have followed the instructions on this page to add a checkbox to the grade marking schema for multiple checkbox and attendance.

Class Name	FullScreenText
Description	It's a helper widget to avoid runtime errors, and if it is not included in the MaterialApp, it can be used for this.
Resources	

5.2. summary.dart

Class Name	WeeklyPage
Description	StatefulWidget WidgetClass for creating a summary page
Resources	

Class Name	_MyWeeklyPageState
Description	State<WeeklyPage> This class is designed to show the weekly average of students' results and includes functions for calculating the scores.
Resources	

5.3. week.dart

Class Name	Week
Description	Conversion to and from Json to map week data classes to the database
Resources	

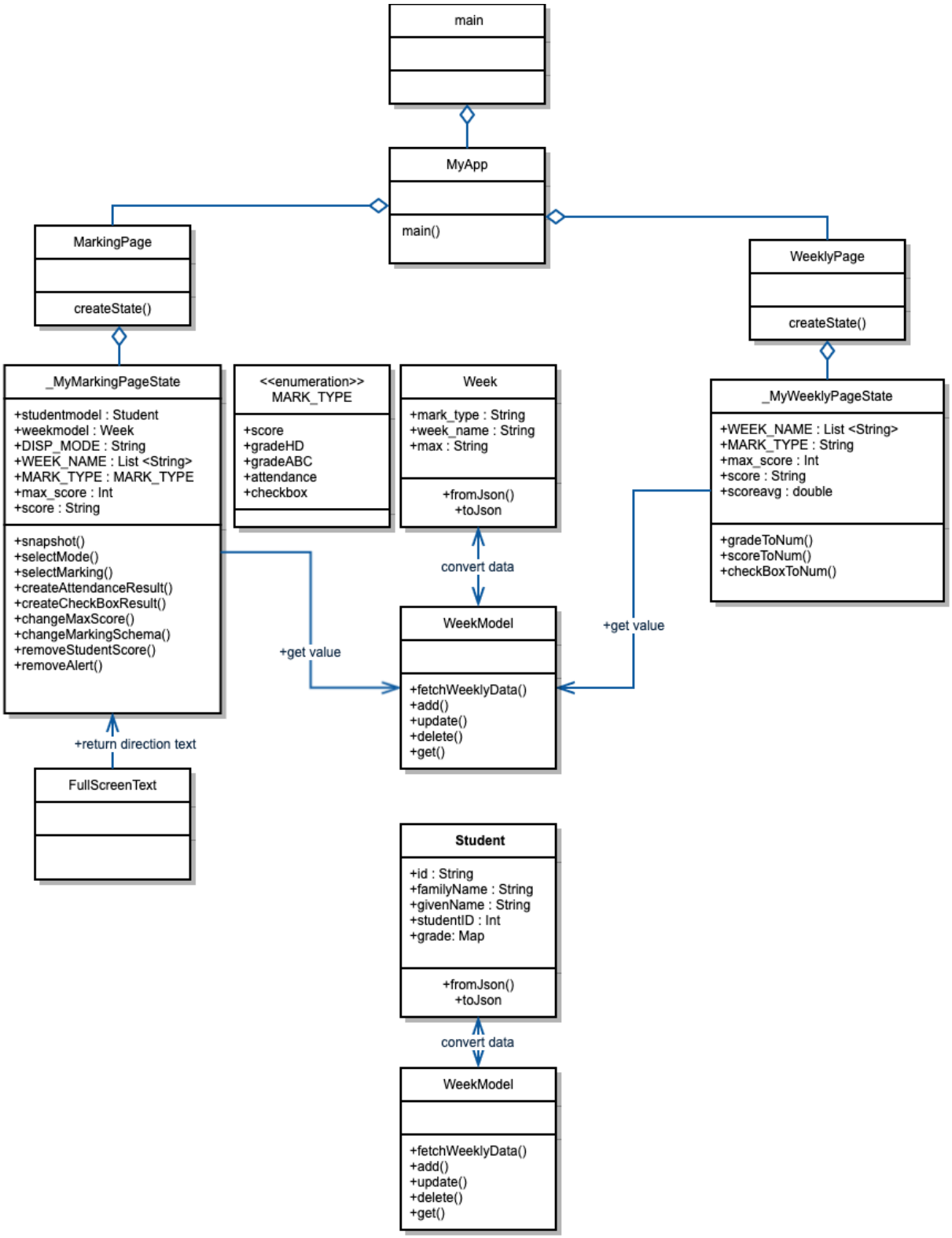
Class Name	WeekModel
Description	Class to connect to the database and manipulate the data in the weeks collection.
Resources	

5.4. student.dart

Class Name	Student
Description	Conversion to and from Json to map week data classes to the database
Resources	

Class Name	StudentModel
Description	Class to connect to the database and manipulate the data in the students collection.
Resources	

5.5. Class diagram



6. Conclusion

Although the same requirements were used in the design and development of the prototype, Android and iOS, the resulting design is quite different, and the cross-platform design has eliminated some of the complexity and improved usability. However, there is still room for improvement in terms of visibility and feedback. Furthermore, there are many improvements to be made in coding and class structure. For example, looking at the diagram, it is clear that it relies on a huge amount of coding within _MyMarkingPageState. This meant that minor errors could cause the application to crash and become completely unusable, and made it difficult to fix the code. Therefore, I learned the need to consider the class structure in detail from the prototyping stage.

7. References

Nielsen, J 1994, *Usability Engineering*, *Google Books*, pp. 23–33, viewed 8 June 2021, <https://books.google.co.jp/books?hl=en&lr=&id=95As2OF67f0C&oi=fnd&pg=PR9&ots=3cAACo9qUu&sig=eIDERPkhmFFGFHOYHTYW_nlOJvo&redir_esc=y#v=onepage&q&f=false>.

Norman, D 2013, *The Design of Everyday Things: Revised and Expanded Edition: Norman, Don: 8601400351710: Amazon.com: Books*, viewed 6 June 2021, <https://www.sunyoungkim.org/class/old/hci_f18/pdf/The-Design-of-Everyday-Things-Revised-and-Expanded-Edition.pdf>.

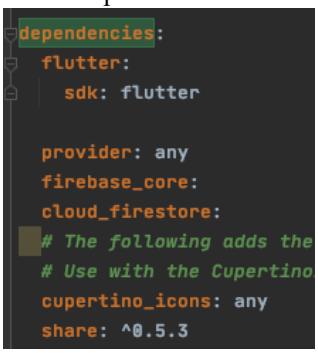
Quesenbery, W 2004, 'Balancing the 5Es: Usability', *Cutter IT journal*, vol. 17, Cutter Information LLC, pp. 4–8, viewed 8 June 2021, <https://wqusability.com/articles/5es-citj0204.pdf>.

8. Appendices (optional)

8.1. Sharing function problem

I tried to install the share function, but I got a missing plugin exception error in the console, and it couldn't work well. In order to preserve my work, I will specify the steps here.

1. Added dependencies "share: ^0.5.3"



2. Import the package into student_details.dart

3. Create a text to share and pass the text to share.share().

```
floatingActionButton: Column(
  mainAxisAlignment: MainAxisAlignment.end,
  children: <Widget>[
    if (!adding) IconButton(
      icon: Icon(
        Icons.send_rounded,
        color: Colors.blue,
      ),  // Icon
      onPressed: ()  {
        print(shareText);
        Share.share(
            shareText
        );
```

However, when the share button is tapped, no event is fired on the emulator. The console shows an error. The text created is not a problem because the student's grades are printed weekly.

```
Coco Moca Student ID: 9876654
week1
No Mark : -
week2
checkbox : 1/5     20 %
week3
score : 25/50     50 %
week4
No Mark : -
week5
gradeABC : A     100 %
week6
No Mark : -
week7
No Mark : -
week8
No Mark : -
week9
gradeHD : HD     80 %
week10
attendance : 1/1     100 %
week11
No Mark : -
week12
No Mark : -
Error: MissingPluginException(No implementation found for method share on channel plugins.flutter.io/share)
    at Object.throw_ [as throw] (http://localhost:64826/dart_sdk.js:5333:11)
    at MethodChannel._invokeMethod (http://localhost:64826/packages/flutter/src/services/system_channels.dart.lib.js:954:21)
    at _invokeMethod.next (<anonymous>)
    at http://localhost:64826/dart_sdk.js:39031:33
    at _RootZone.runUnary (http://localhost:64826/dart_sdk.js:38888:58)
    at _FutureListener.thenAwait.handleValue (http://localhost:64826/dart_sdk.js:33874:29)
```

I did some research and found some articles stating that the sharing feature is not available in the web chrome emulator.