CONFIDENTIAL

QAIWAN INTERNATIONAL UNIVERSITY

UTM UNIVERSITI TEKNOLOGI MALAYSIA

MID TERM TEST SEMESTER I, 2021/2022

SUBJECT CODE	**:**	**SCSJ3623**
SUBJECT NAME	**:**	**MOBILE APPLICATION PROGRAMMING**
TIME / DURATION	**:**	**1:00 – 4:00 PM / 3 HOURS**
DATE/DAY	**:**	**7 DECEMBER 2021 / TUESDAY**

INSTRUCTIONS

- This is an **OPEN-BOOK** test.
- This is a practical test. You will attempt the test by doing coding on computer.
- You will be on eLearning only for downloading the question and submitting the answer.
- You will be writing the code offline using VS Code.
- Any kind of discussions whether verbal, chats, online, etc. is strictly prohibited.
- All COMMENTS you write in the program WILL NOT BE GRADED.
- Program that CANNOT COMPILE will get a penalty.
- You are required to track changes of your code using Git. Submission without the Git commits will get a penalty.
- Any question related to the test question will not be entertained at all. Read the question carefully.
- You do not have to do the video recording.

PLAGIARISM WARNING

- Be warned that your program will undergo a screening process for PLAGIARISM DETECTION.
- Programs that are found cheating will get a zero mark.

SUBMISSION

- Submission must be done via the eLearning system. Other than that (including email, telegram, whatsapp, etc) will not be accepted.
- You will need to submit your code packaged in a **git bundle file**
- You will be given another **15 minutes** at the end of the session for the submission.

GIT COMMANDS

These are the git commands that you may need to use in this test. Run the command on a gith bash terminal. Note that, the commands are case-sensitive.

git init:

- The codebase comes in a zip file. You will need to extract the file into a folder.
- Then initialize the folder with git by issuing a **git init** command:

```
$  cd your_codebase_folder
$  git init
```

git commit:

To perform a new commit:

```
$  git add -A
$  git commit -am "your commit message"
```

Optional: In case you need to update the recent commit:

```
$  git add -A
$  git commit --amend
```

git bundle:

To create a git bundle file for submission:

```
$  git bundle create ../your_matric_number.git HEAD master
```

Note: The git bundle file should be saved outside of your project folder.

Testing the git bundle file:

Optional: In case you want to verify the git bundle whether it has been created correctly, unbundle or unpack the git bundle file to a folder with a **clone** command.

```
$  cd ..
$  git clone your_matric_number.git another_folder
```

ABOUT GIT COMMITS

- **What to commit? -** Perform a git commit for each task, step or feature completed.
- **Does the order matter? –** It is not necessarily to follow the exact order as the question. For example, you are supposed to start with the Task 9 and 10 to create a mock database (see the Question section). Thus, you should commit those two tasks first.

Question [100 Marks]

This test came with a codebase of a Flutter project. You MUST USE this codebase to attempt the test. Also, you are not expected to add any additional packages to the codebase.

In this test, you will develop a prototype of Note application. The prototype must be fully functioning in terms of User Interface and User Experience (UI/UX). The application will proivide three basic operations: adding new notes, viewing notes and editing existing notes. Regarding data storage, all data are manipulated inside the memory. That means, you should expect that the data will lost once the app closes. You are not expected to use any real database. As for the state management, you will need to use the Stateful Widgets approach.

The application is composed from two screens, List screen (the main screen) and Note screen (the second screen), see Figure 1 and 2 at the end of this question book. The UI and functionalities of the screens are given below along with the assessment marks.

The List screen (main screen):

1. Showing the list of notes on the main screen
   a. The main screen will show all the notes that are read from the mock database. The information shown for each note includes the note's title and content.
   b. Besides, the main screen will also show the number of notes at the screen bar.

   See the ***video 1*** *for the expected result of this part.*

   (10 marks)

2. Showing and hiding the notes' content:

   a. When the **"Show Less"** button is tapped on, it will hide the contents of all notes from being displayed, and leaving only the notes' titles. Also, the button icon will change to the "Show More" icon.

   b. When the **"Show More"** button is tapped on, it will show the notes' contents and the button icon will turn back to the "Show Less" icon.

   See the ***video 2*** *for the expected result of this part.*

   (10 marks)

3. Turning on and off the editing tools:
    a. When a note tile is long-pressed , it will show the editing tools (which consists of Edit and Delete buttons) for the note.
    b. Long-pressing on the same note tile again will hide the editing tools.
    c. Long-pressing on another tile will show editing tools on the tile, and hide it from the previous tile. That means, the editing tool only visible on one tile at a time.

    *See the **video 3** for the expected result of this part.*

(10 marks)

4. Removing a note - when the **Delete** button on a note is tapped on, it will delete the note from the mock database and reflect the changes on the UI accordingly.

   *See the **video 4** for the expected result of this part.*

(5 marks)

5. Opening the second screeen:
    a. When a note tile is tapped on, it will open the second screen (the Note screen) for the note in "View" mode.
    b. When the **Edit** button on a note is tapped on, it will open the second screen in "Edit" mode.
    c. When the **Add** button at the bottom of the screen is tapped on, it will open the second screen in "Add" mode. This will add a new note to the list.
    d. All the above operations 5(a)-(c) are achieved by making use of the same Note screen. You do not create a different screen for each operation

    *See the **video 5** for the expected result of this part.*

(20 marks)

The Note screen (second screen):

6. Showing the screen bar of the Note screen according to the operation mode:
    a. If this screen is opened for view (from 5(a) above), the screen bar will have the title **"View Note"** and only the **Cancel** button.

   b. If this screen is opened for editing a note (from 5(b) above), the screen bar will have the title **“Edit Note”** and both the **Save** and **Cancel** buttons.
   c. If this screen is opened for adding a new note (from 5(c) above), the screen bar will have the title **“Add new Note”** and both the **Save** and **Cancel** buttons.

   *See the **video 5** for the expected result of this part.*

(15 marks)

7. Show only or allow editing the note’s data based on the operation mode:
   a. If this screen is opened for view (from 5(a) above), the user can only view the note’s data (i.e., title and content). The data cannot be edited at all.

   b. If this screen is opened for edit or add (from 5(b) and (c) above), The user is allowed to edit the note’s data.

   *See the **video 5** for the expected result of this part.*

(10 marks)

8. Confirming or cancelling changes:

   a. When the **Save** button is tapped on, it will bring the user back to the main screen. This operation also saves the note to the mock database and reflect the changes on the UI of the main screen accordingly.
   b. When the **Cancel** button is tapped on, it will cancel any changes and bring the user back to the main screen.

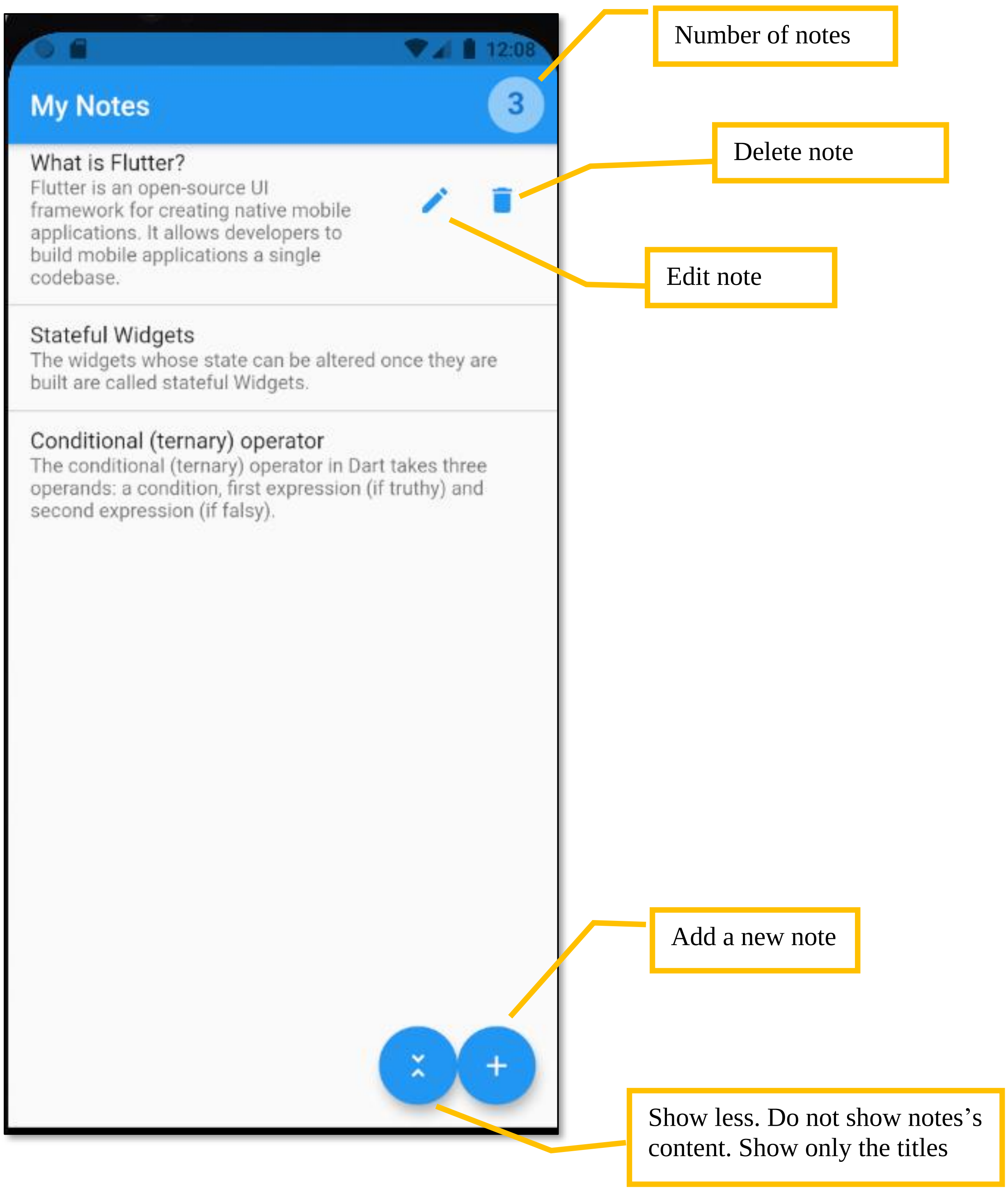
   *See the **video 6** for the expected result of this part.*

(10 marks)

Mock database

You are required to use a mock database in order to provision data to the application. In this regard, you need to do the following tasks:

9. Define a class to represent each note which consists of title and content. (5 marks)
10. Use the class to create some mock data for the application. (5 marks)

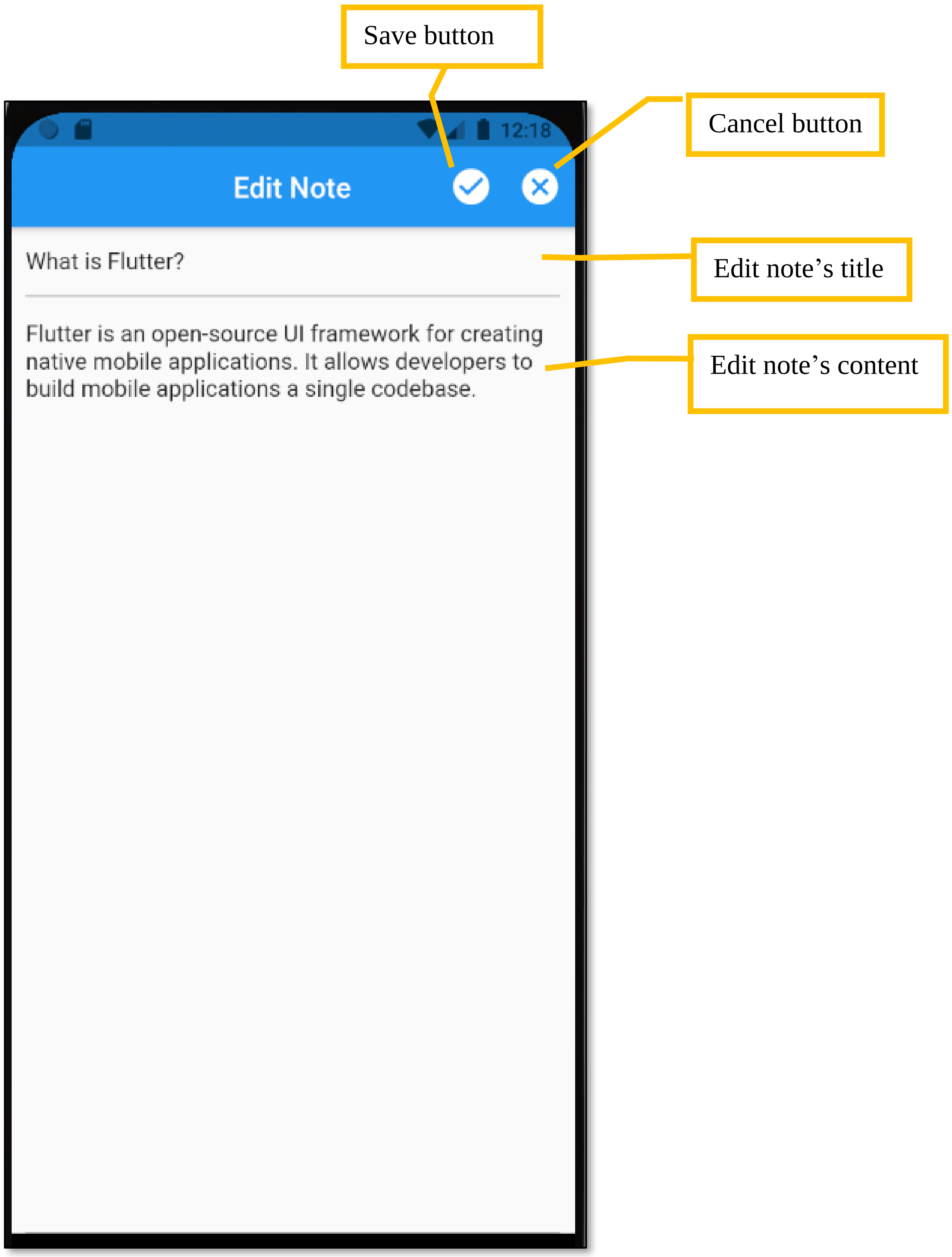


What is Flutter?
Flutter is an open-source UI framework for creating native mobile applications. It allows developers to build mobile applications a single codebase.

Stateful Widgets
The widgets whose state can be altered once they are built are called stateful Widgets.

Conditional (ternary) operator
The conditional (ternary) operator in Dart takes three operands: a condition, first expression (if truthy) and second expression (if falsy).

Figure 1: List screen (the main screen)



What is Flutter?

Flutter is an open-source UI framework for creating native mobile applications. It allows developers to build mobile applications a single codebase.

Figure 2: Note screen (the second screen)