**Student Name: Weight: 10%**

**Student ID:** **Marks:** **/60**

# Assignment: Navigation

## Introduction

In this assignment students will use the concepts they learned in class to demonstrate their understanding for React Navigation (Stack Navigation). Students are required to build a React Native application, **BrokeBuddy**, this application will help people to keep track of their expenses. This assignment will test students on their understanding of React Navigation (Stack Navigation) and some other concepts like useState, useEffect hooks and FlatList.

Plagiarism detection software will be used on all student submissions. Please refer to SAIT’s policies and procedures on Student Academic Conduct ([A.C.3.4.3](https://www.sait.ca/assets/documents/about-sait/policies-and-procedures/academic-student/ac-3-4-3-student-academic-conduct.pdf)) for more information.

## Instructions

1. This is an individual assignment. You should expect to work on this assignment outside of class time.
2. Please refer to Brightspace for the due date.
3. Check the presentation and Video attached on D2L to get a feel of the application. The presentation and video are just for reference, you don’t have to stick to the UI shown, be creative and try give your best shot while developing this application.
4. The application should have below functionality:
   1. User should be able to add new transaction.
   2. Each new transaction should have a Title, Description, Amount and Transaction type (Essential, Leisure, Others). All these fields are mandatory.
   3. User should be able to View details of add transaction.
   4. User should be able to Edit an existing transaction.
5. The home screen of the application should display a list of all transaction user has saved.
6. In list view, based on the transaction type, the background color of each item is selected. Below are the colors you can use:
   1. Essential Transactions : Green (#C9E9D2)
   2. Leisure Transaction : Red (#FFCFB3)
   3. Other Transaction : Blue (#CAF4FF)
7. If the list is empty, i.e. user has not entered any transaction, an empty view should be displayed with proper message.
8. To add a new transaction, a **Floating Action Button** ([FAB](https://medium.com/@harsimrans329/floating-action-button-fab-in-mobile-reactive-programming-bfb217876f3c)) should be added on home screen.
9. On clicking the FAB, user should navigate to a new screen (Add transaction) to fill the details of the new transaction.
10. The Add transaction screen should have a form to take input from user (refer point 4.b to see the required fields).
11. When click on **submit** button on Add transaction screen **while adding a new transaction**, a unique id should be generated, and the entry should be persisted.
12. To generate unique id, you can use **getNewId()** function from **utility.ts** file.
13. For persisting the data, **utility.ts** file has an array **TRANSACTION\_DATA**, and methods to add, update a transaction, retrieve transaction using transaction id. Check utility.ts file for more information.
14. At home screen, by clicking on any list item (transaction entry), user should navigate to details screen which should display a card with details of the transaction selected.
15. The details screen should have an **Edit** button in the header section, by clicking this button user should navigate to add screen, but in edit mode, which means all the details of the selected transaction should be pre-filled. Check the video for better understanding.
16. When the Add transaction screen is opened in Edit mode, clicking submit button should not generate a new transaction ID, you will use the existing transaction ID of that transaction entry.

## Marking Criteria

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Needs Improvement** | **Good** | **Excellent** | **Marks** |
| **Project Folder Structure** | All files are create in project’s root directory.(0 marks) | All files are created in the src folder but not inside the components folders.(1marks) | Project folder structure is correct and new components are created inside components folder.  (3 marks) | **/3** |
| **User Interface** | 1 or more UI component is missing (2 marks) | All components are present, but the layout is not properly styled and all components are close to one another or not centered aligned (5 marks) | UI is properly styled, and each component is centered aligned with proper spacing around it. (10 marks) | **/10** |
| **Navigation** | No Navigation is added to the application (0 marks) | Navigation container is added and all 3 screens are added but some navigation is missing. (7 marks) | All the navigation is properly added and user is able to move between screen as per the explained behaviour. (15 marks) | **/15** |
| **Add transaction Validation** | No validation is added in add transaction screen (0 marks) | Validations are added but error messages are not displayed properly. (3 mark) | All validation is added and proper error message is displayed. (7 marks) | **/7** |
| **Details Screen** | Details screen is not well formatted and Edit button is missing (0 marks) | Detail screen is well formatted and edit button is added but not in header. (5 mark) | Details screen is well formatted and edit button is added on header section with proper functionality. (10 marks) | **/10** |
| **Add Transaction Screen Edit mode** | Add transaction screen is not utilized to edit an existing transaction (0 marks) | Add transaction screen is opened in edit mode but data is not prefilled or while saving, new transaction is added instead of editing the previous one (5 marks) | Add transaction screen is opened in edit mode and data is prefilled and no new transaction is added after edit transaction. (10 marks) | **/10** |
| **Home Screen** | 1.)Empty view is not added 2.)FAB is not used. 3.)FlatList is not used to create list of transactions. (0 marks) | 2 out of 3 are implemented (2 marks) | All 3 are implemented (5 marks) | **/5** |
| Total | | | | **/60** |