"Expenser" Project Proposal

1. Product Definition

App name: Expenser

Description:
People spend money every day for various purposes. As students, we want to track our spending details to keep our budget balanced and maintain our long-term financial health. However, tracking and managing personal spending is not an easy task. We do need a tool to make the magic happen. By using our app, users can record their expenses easily by scanning receipts, split bill and collect money from friends, and track personal expenses.

The app offers three functions:
Use Case 1: Smart Scan
- Scan receipts to record expense details, including total amount, transaction date, expense category, intelligently.

Use Case 2: Accurate Split
- Split bill with Facebook friends and send them notifications via Messenger.

Use Case 3: Summary
- Show expense summary and details per category by month.

2. Solution Approach

Android

(1) Third-party web service: **Google Cloud Vision API** (Optical Character Recognition)
To support the scanning and resolving of receipts, we would use Google Cloud Vision API which is a third-party OCR API. Google Cloud Vision API enables us to understand the content of an image by encapsulating powerful machine learning models in an easy to use REST API[1].
The flow of data in the OCR procedure will be as following[1]:
    a) An image is uploaded to Cloud Storage.
    b) A Cloud Function is triggered, which uses the Vision API to extract the text, and queues the text to be translated into the configured translation languages.
    c) For each queued translation, a Cloud Function is triggered which uses the Translation API to translate the text and queue it to be saved to Cloud Storage.
    d) For each translated text, a Cloud Function is triggered which saves the translated text to Cloud Storage.

In terms of data exchange type, the Cloud Vision API is a REST API that uses HTTP POST operations to perform data analysis on images[2]. The API uses JSON for both requests and responses. Thus we will use URL connections with JSON serialization for the app.

(2) Social media: **Facebook SDK for Android**
Our app will integrate with Facebook via Facebook SDK for Android which supports Facebook login and sending messages to Facebook friends. The API uses RESTful protocol and responses are in JSON format[3]. Therefor we will use URL connections with JSON serialization for the app.

(3) Data persistence: **SQLite & Backend MySQL database**
For data persistence, we will use SQLite for local data storage to keep user accounting information. As we want to support data backup in case the user may want to switch to another phone without losing existing information, we would implement a backend MySQL database. If the user wants to backup, the app will interact with the backend database to transfer all the local user data to backend.

IOS

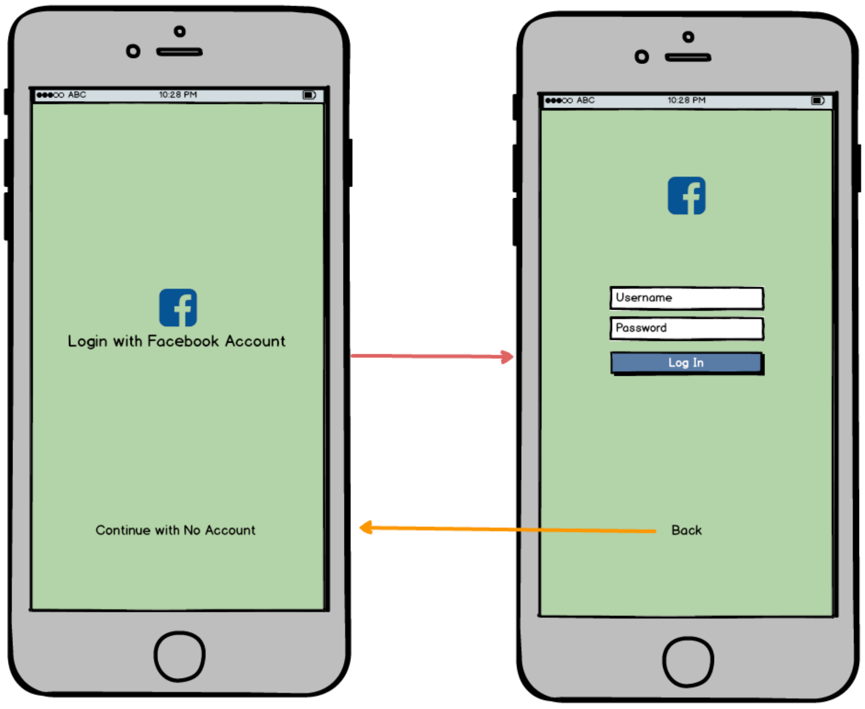
(1) Third-party web service: **Google Cloud Vision API** (Optical Character Recognition)
Same as Android, we will also use Google Cloud Vision API as the third-party OCR API which enables us to understand and resolve the content of an image using REST API. We will also use URL connections with JSON serialization for the app.

(2) Social media: **Facebook SDK for iOS**
Our app will integrate with Facebook via Facebook SDK for iOS to support Facebook login and sending messages to Facebook friends. The API uses RESTful protocol and responses are in JSON format[3]. Therefore we will use URL connections with JSON serialization for the app.

(3) Data persistence: **Core Data Framework & Backend MySQL database**
For data persistence, we will use Core Data Framework for local data storage. Similar to Android, we will also support data backup by building backend MySQL database in case the user may want to switch to another phone without losing existing information. The app will transfer all the local user data to backend database when the user chooses to backup.

3. Prototype and User Study

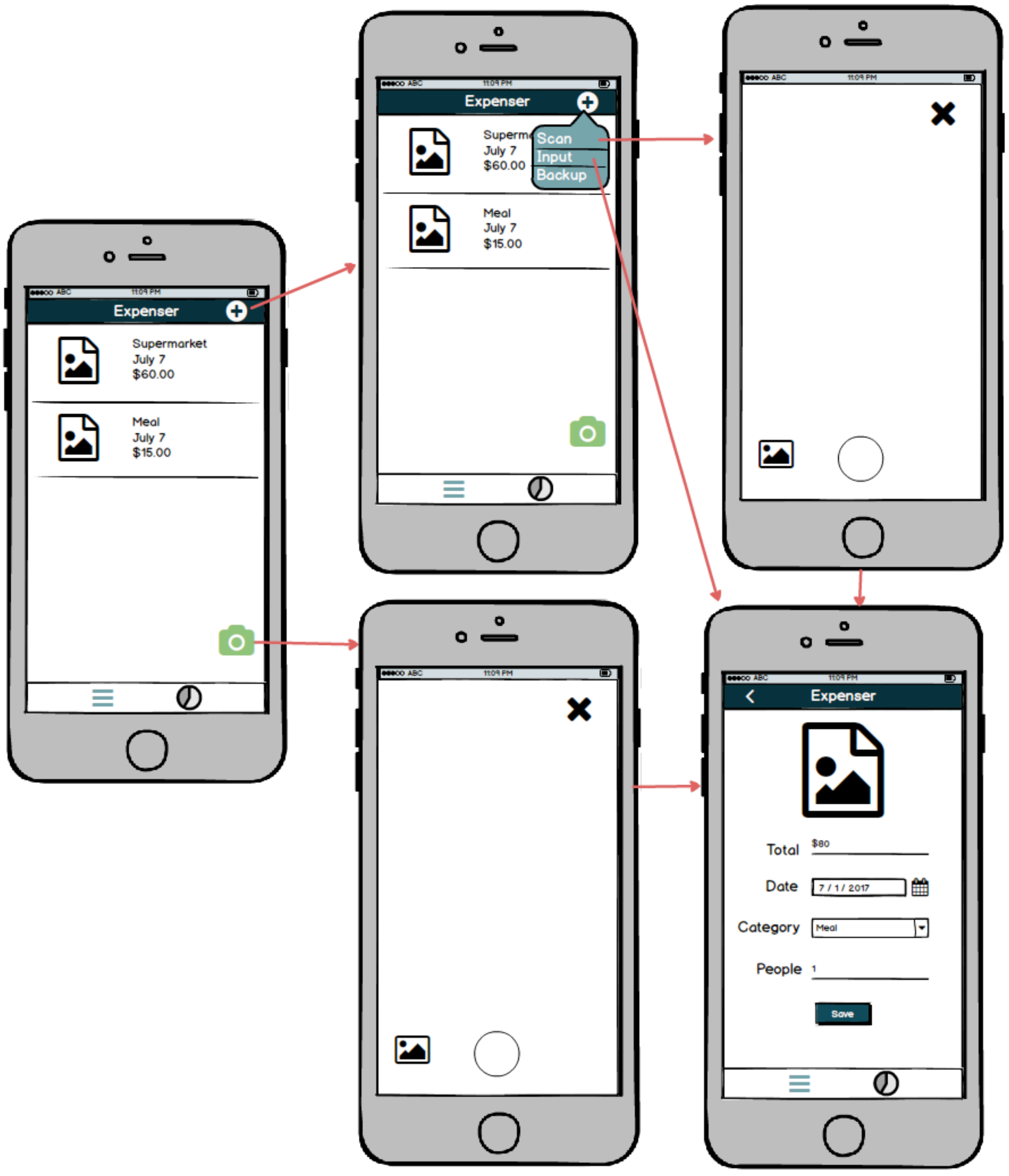
(1) When launching the app, the user can login with Facebook or just continue with no account. It will bring the user to home screen.



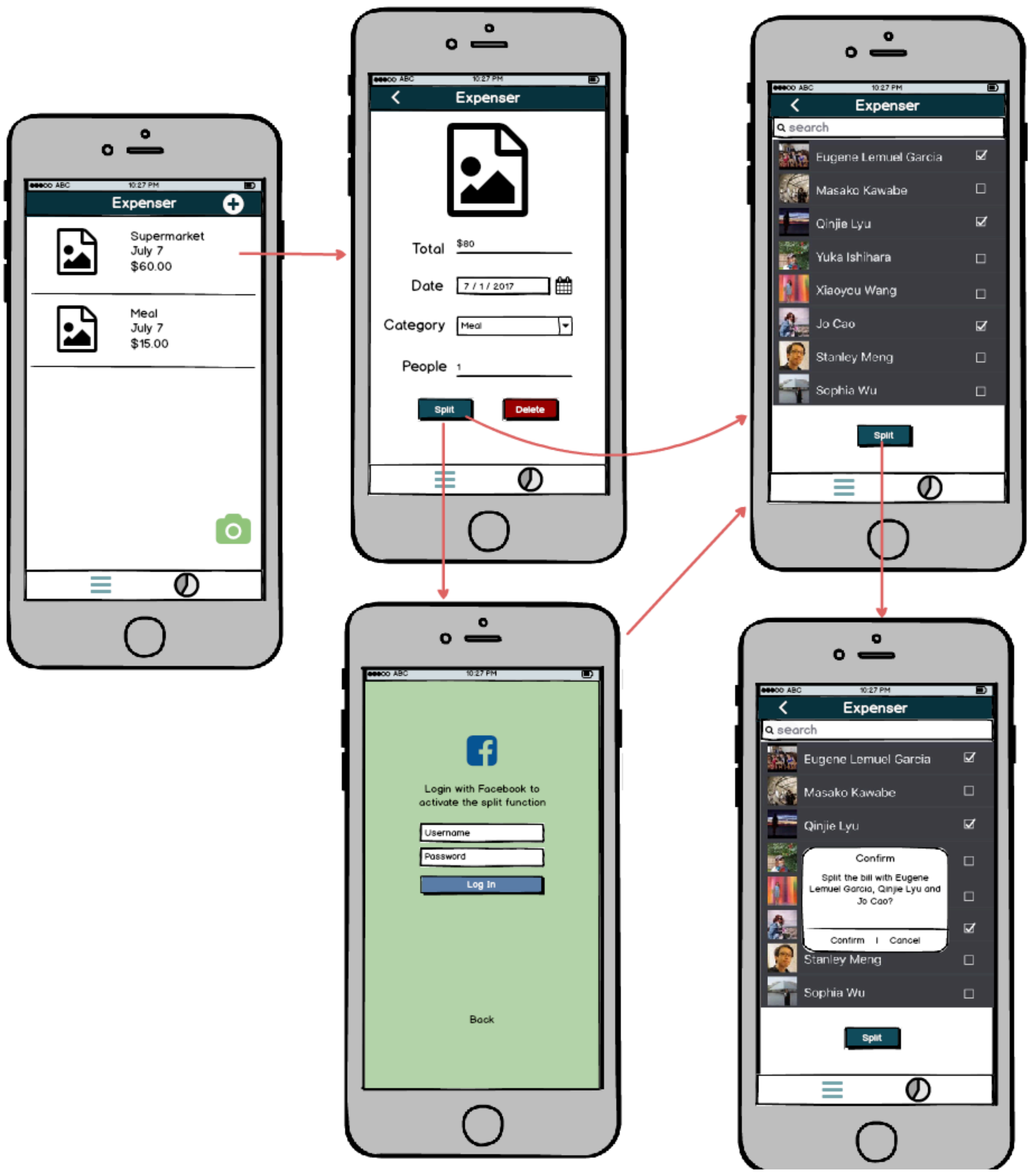
(2) In home screen, there is a plus icon on the top right corner, which has three options: Scan, Input, and Backup.

- When the user chooses Scan, it will activate the camera sensor. Then the user can take photo of receipt. And, the user can also choose to input an image from album. The app will recognize the total cost and transaction date automatically. At the same time, the user can choose category for this expense. Then the user can go back to home screen.
- When the user chooses Input, he can input the total cost, date, and category directly without scanning the receipt.
- When the user chooses Backup, it will synchronize the local data to cloud.

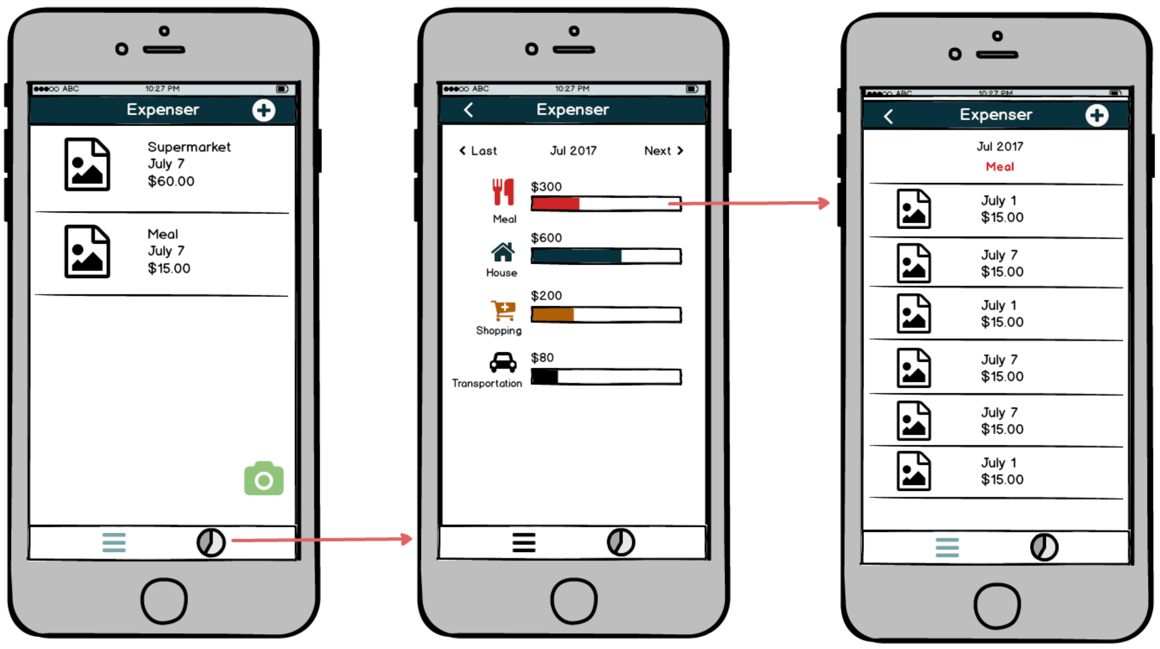
In addition, there is a camera icon on the bottom right of the home screen. It is a fast way for user to scan receipts.



(3) When user wants to split an expense, user should click one expense record on the home screen. Then, it would switch to expense detail, which includes the total amount, the transaction date, and the category it belongs and the number of people involved. There are two buttons on the bottom of the screen, Split, and Delete. When the user clicks the Split button, the app would ask the user to login with the Facebook username and password if the user logins with no account. Otherwise, the user would switch to the Split screen, where the user can choose his friends on the friends list to share the expense with. Finally, the user can send the request notification to the chosen friends via Facebook after confirming these friends.



(4) When the user wants to check his monthly summary, he can click the second tab to go to the Summary screen. In this screen, it will show the total cost and each cost for each category for the current month by default. And the user can also change the month on the top of the screen to check the previous monthly record. When the user click a category, it will show a list of images, which are the receipts of this category.



Reference:
[1] https://cloud.google.com/functions/docs/tutorials/ocr
[2] https://cloud.google.com/vision/docs/request
[3] https://www.programmableweb.com/api/facebook