Presented to the Industrial Engineering Department

De La Salle University – Manila Campus

Term 1, A.Y. 2023 - 2024

In partial fulfillment

of the course

LBYEC2B

Personal Finance Organizer

Team Apollo

Submitted by:

ANG, Lorenzo Jacob

CUA, Sean Austin

ABE, Kousuke Gutierez

EB4

Submitted to:

Professor Ruiz Ramon

Dec 4, 2023

I. Introduction

In the world today, there is one thing that most people’s world revolves around, and that is money. In today’s modern world, every person either spends money, earns money, gives money, saves money, and many more. Some people even dedicate their lives to money, while some people suffer just because of money, and that just shows how important money is today. Here comes the problem and issues surrounding money. People tend to lose track of their money and realize it later when it’s all gone. Moreover, according to Tuggle (2016), around 87% of teens don't have enough knowledge of personal finance. Therefore, a code or app that lets people successfully track their money is very significant. An app that is easily accessible and safe for everybody to use. An app that is easily understandable by everybody including children and adults can be used. Additionally, it can even be a start for young kids to be introduced to the financial world. Where they can cash in and cash out, and the code or app calculates it for them. Furthermore, an app that is capable of letting the user set their goals, to let the user see his/her progress as well. In short, an app that caters as a personal finance organizer for the user with safe security.

II. Objectives

The main objectives for this project are:

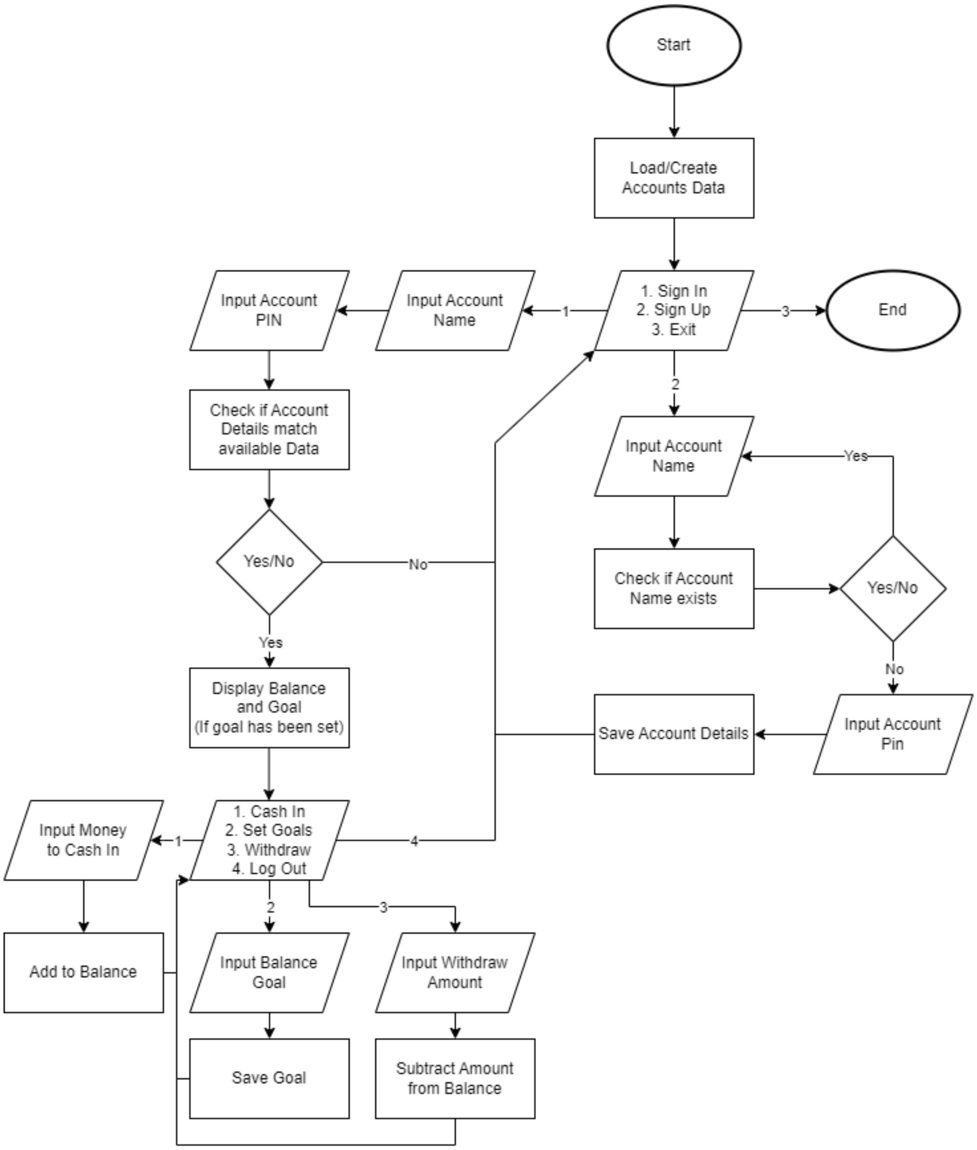
- To create a code/app that lets you login and logout to different accounts
- To create a code/app that has a very secured and safe security
- To create a code/app that shows the user his/her updated balance
- To create a code/app that lets the user cash in and cash out (proper tracking)
- To create a code/app that notifies the user’s status on their goal

III. Related Work

This project is similar to other existing apps such as “You Need A Budget” (YNAB) and PocketGuard. These applications and ours are quite similar as all can keep track and set goals for people. Missing components to our app compared to theirs would be the lack of proper interface and more functionalities. However, using our app, changing accounts for different people is quite easier compared to theirs while still being secure and at the same time providing memory of past usages. (YNAB, n.d.) (Holzhauer, 2023)

IV. Methodology

Before the development team starts coding a personal finance app, they first develop a simple flowchart. This flowchart lays out the app's key functions such as user login, financial management, and balance tracking. It serves as a tool for organizing the app's operations and ensures that all important features are addressed. The flowchart is crucial for early identification of potential issues, facilitating better collaboration among the team, and ensuring clear communication. The team continuously refines this flowchart based on collective feedback and user input. It acts as a guide during the coding process, helping them build the app in a structured manner that meets user needs effectively. This preparatory step is vital for streamlining the development process and enhancing the overall quality of the app.



1. **Start:**

   The user begins the process.

2. **Load/Create Accounts Data:**

   The application either loads existing account data or creates a new data set if none exists.

3. **Main Menu Options:**

   The user is presented with three options

   - Sign In
   - Sign Up
   - Exit

4. **Sign In Process:**
   - The user inputs their account name.
   - The user inputs their account PIN.
   - The system checks if the account details match the available data.
   - If the details match, the app displays the balance and, if set, the goal.
   - The user is then presented with options to Cash In, Set Goals, Withdraw, or Log Out.

5. **Cash In:**
   - The user inputs the amount of money to cash in.
   - The amount is added to the balance.

6. **Set Goals:**
   - The user inputs their balance goal.
   - The goal is saved in the system.

7. **Withdraw:**
   - The user inputs the withdrawal amount.
   - The amount is subtracted from the balance.

8. **Sign Up Process:**
   - The user inputs a new account name.
   - The system checks if the account name already exists.
   - If it doesn't, the user is prompted to input a PIN.
   - The account details are saved.

9. **Exit:**

   The process ends, and the user exits the app.

Each step includes decision points where the user's input determines the next step in the process. If at any point the user's input does not match the expected or required data (such as an incorrect PIN or an existing account name during sign-up), the flowchart indicates that the user will be prompted to re-enter the correct information.

V. Results and Discussion

```
% run program %
function eWalletProgram()
if exist('accData.mat', 'file')
     load('accData.mat', 'accounts');
else
     accounts = struct('name', {}, 'pin', {}, 'balance', {}, 'goal', {});
end
 while true
    fprintf('\n1. Sign In\n2. Sign Up\n3. Exit\n');
    choice = input('Enter your choice: ');
    % matching user input %
    switch choice
        case 1
            accounts = signIn(accounts);
        case 2
            accounts = signUp(accounts);
        case 3
            fprintf('Exiting...\n');
            break;
        otherwise
            fprintf('Invalid choice. Please try again.\n');
    end
end
save('accData.mat', 'accounts');
end
% sign in to existing account %
function accounts = signIn(accounts)
   accountName = input('Enter your account name: ', 's');
   pin = input('Enter your 6-digit PIN: ', 's');
   index = find(strcmp({accounts.name}, accountName));
       if ~isempty(index) && strcmp(accounts(index).pin, pin)
           fprintf('Logged in successfully.\n');
           loggedInAccount = accounts(index);
           loggedInAccount = userMenu(loggedInAccount, accounts);
           accounts(index) = loggedInAccount;
       else
           fprintf('Invalid account name or PIN.\n');
       end
end
% create account %
```

```
function accounts = signUp(accounts)
% check name %
while true
    newAccount.name = input('Create account name: ', 's');
    if any(strcmp({accounts.name}, newAccount.name))
        fprintf('Account name already exists. Please choose a different
name.\n');
    else
        break;
    end
end
% check pin %
while true
    newAccount.pin = input('Create a 6-digit PIN: ', 's');
    if numel(newAccount.pin) == 6 && all(isstrprop(newAccount.pin, 'digit'))
        break;
    else
        fprintf('Invalid PIN. Please enter exactly 6 digits.\n');
    end
end
confirmPin = input('Confirm your 6-digit PIN: ', 's');
% matching pin for confirmation %
while ~strcmp(newAccount.pin, confirmPin)
    fprintf('PINs do not match. Please try again.\n');
    confirmPin = input('Confirm your 6-digit PIN: ', 's');
end
% initializations %
newAccount.balance = 0;
newAccount.goal = 0;
accounts(end+1) = newAccount;
fprintf('Account created successfully.\n');
end
% access functionalities of the code %
function loggedInAccount = userMenu(loggedInAccount, accounts)
while true

    % checking if goal has been set and progress display %
    fprintf('\nCurrent balance: %.2f PHP\n', loggedInAccount.balance);
    if loggedInAccount.goal ~= 0
        goalProgress = loggedInAccount.balance/loggedInAccount.goal*100;
        fprintf('Current goal: %.2f PHP\n', loggedInAccount.goal);
        fprintf('Progress to goal: %.2f%%\n', goalProgress);
        if goalProgress >= 100
           fprintf('Goal Achieved!\n');
        elseif goalProgress > 50 && goalProgress < 75
           fprintf('More than halfway there! Consider investing your money in
stocks to grow your money further and faster!\n', goalProgress);
        elseif goalProgress > 75
           fprintf("We're almost there! Just a bit more.\n", goalProgress);
        end
    end
    fprintf('\n1. Cash-in\n2. Set Goals\n3. Withdraw Money\n4. Log Out\n');
    choice = input('Enter your choice: ');
    % match user input %
    switch choice
        % cash in %
        case 1
            while true
               cashInAmount = input('Enter the amount to cash-in: ');
               % catching error %
               if cashInAmount >= 0
                   break;
               end
               fprintf('Invalid Amount. Please try again')
               % ~~~
            end
            % update balance %
            loggedInAccount.balance = loggedInAccount.balance + cashInAmount;
            fprintf('Cashed in %.2f PHP.\n', cashInAmount);
            goalStatus = loggedInAccount.goal - loggedInAccount.balance;
        % setting goal %
        case 2
            while true
               goalAmount = input('Enter the goal amount: ');
               if goalAmount > 0
                   break;
               end
               fprintf('Goal amount must be positive. Please try again.');
            end
            loggedInAccount.goal = goalAmount;
            fprintf('Goal set successfully. Current goal: %.2f PHP\n',
loggedInAccount.goal);

        % withdrawing money %
        case 3
            withdrawAmount = input('Enter the amount to withdraw: ');
            if withdrawAmount < 0
                fprintf('Cannot withdraw a negative amount');
            elseif withdrawAmount <= loggedInAccount.balance
                loggedInAccount.balance = loggedInAccount.balance -
withdrawAmount
            else
                fprintf('Insufficient Balance\n')
            end

        % log out of account %
        case 4
            fprintf('Logging out...\n');
```

```
            return;
        otherwise
            fprintf('Invalid choice. Please try again.\n');
    end
end
end
```

—---

Overall, the code was able to let people create accounts with a specific pin number. It also remembers them when you create one. So meaning once you create one, no one can reuse your name that you put. Making it limited for the user and safe. PIN numbers were used for proper verification of accounts. Moreover, just like any other accounts in social media, the code also has an option for you to log out after you are done using, so that the next user can use the app or code. At the same time, even after closing the program, the application will still remember your credentials and you can continue to use the app where you last left off.

Additionally, the code lets you put or cash in money and cash out or withdraw money. And of course, it affects the balance that you have in your account. Any cash in or withdrawal would automatically subtract or add your balance. The balance is shown every transaction so that it can show transparency of the money.

Lastly, the feature of setting goals. The code asks you to set the money that you want to achieve and notifies you if you are nearing the goal or getting further. This could help the user to see if he/she has done well financially and would serve as for the user. It also suggests some advice like investing in stocks.

Moreover, after creating the code, proper evaluation was done to test out the validity of the code. First done is the integration testing, where the code was tested to run perfectly and execute all the features from start to finish without experiencing any errors. As can be seen in this video attached below, where the final and finished code is demonstrated, and all the features are working.

Error testing was also done in order to make the code more reliable, especially in terms of security. If you try to login to someone's account, it won't let you in unless the pin number you put is right. It would just say "PINs do not match. Please try again". Furthermore, if you don't select a number in the choice provided, it would tell you that you selected an invalid choice and return you back to where you can choose again. Withdrawal of insufficient money is also not allowed in the code, and would notify the user immediately.

Youtube link: https://www.youtube.com/watch?v=DiTUob2IT0I

VI. Conclusion and Future Work

In essence, the code worked perfectly well after proper evaluation, and was able to accomplish all the objectives mentioned above. All of the features of the app works perfectly and all errors have been fixed. Furthermore, it is very easy to understand to make sure that it can be used by everyone, kids, teens, and adults. All the instructions are in the code, and all the people have to do is input the needed information.

For future works, the app could have a feature where it can easily transfer the money from account to account. Furthermore, the programmers can scale up the application to be more wide scale and at the same time have better security to come with it.

VII. Contributions (Individual contributions)

Student Name	**Tasks Done**
Ang, Lorenzo Jacob	- Related Work - Conclusion and future work - Code
Cua, Sean Austin	- Introduction - Objectives - Results and Discussion - Code
Abe, Kousuke	- Methodology - Results and Discussion - Code

VIII. References

Holzhauer, B. (2023). *PocketGuard budgeting App Review 2023*. Forbes. https://www.forbes.com/advisor/banking/pocketguard-budgeting-app-review/

Tuggle, K. (2016). *Teaching gap: 83% of teens don’t know how to manage money*. Fox Business. https://www.foxbusiness.com/features/teaching-gap-83-of-teens-dont-know-how-to-manage-money

YNAB. (n.d.). https://www.ynab.com/