JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

B.Sc. Computer Science

ICS 2209: DESIGN AND IMPLEMENTATION OF COMPUTER APPLICATIONS

GROUP MEMBERS:

1. Joseph Muema – SCT211-0039/2022
2. Brian Kipng’eno – SCT211-0078/2022
3. Joseph Njenga – SCT211-0040/2022
4. Kelly Kasina – SCT211-0038/2022
5. Terry Mutheu – SCT211-0073/2022
6. Jayden Mathenge Muriithi – SCT211-0030/2022

# PERSONAL FINANCE MANAGER INTRODUCTION

The following document is a proposal for the PFS (“Personal Finance Manager”) application, which is meant to offer a system from which users can manage their finances in a more efficient and effective manner, and provides them (the users) with various tools to achieve the said goal of better finance management.

The application will employ various tools/features, for example; transaction management, that tracks and categorizes various transactions by the user, Graphical spend representation, which provides a more convenient medium for spend analysis, to achieve the goal of providing an efficient, understandable and user-friendly environment to carry out financial management.

The app targets individuals with the need to find a cheaper/ easier way of managing their finances.

The text below is a simple look into the system’s functional and non-functional requirements, software requirements as well as the hardware requirements for the proposed program.

**Functional Requirements:**

* **User Registration and Authentication** – the system requires users to create an account and set a password so as to make it secure through registering and logging in.
* **Transaction Management** – the application has a feature for recording Income & Expenses enabling the categorization of transactions for better tracking.
* **Budgeting** – the application should provide a feature for users to set up budgets for different categories and track their spending against these budgets.
* **Savings Record** – the user can set apart a part of their income and existing finances for the purpose of saving. Saving goals can be created and personalized according to the user’s specification.
* **Financial Goal Setting** – users of the application can set financial goals to be achieved within a specified amount of time.
* **Loan and Debt management** – through this feature, users can manage the loans they give, keeping track of the loaner, payments and also their debtors and expected pay dates.
* Report and Analysis – the application offers an analysis of the recorded expenditures over time including graphical representation of the same.

## Non-functional Requirements

* Usability – the application is to offer a user-friendly and intuitive interface for smoother and better interactions with the account holders.
* Security – the user’s private financial data is kept safe through robust security measures beginning with mechanisms such as password authentication.
* Scalability – the application is to handle an increasing amount of work smoothly and without lags.
* Maintainability – the application should be designed in a way that makes it easy to update and maintain. This includes using coding standards and having a clear documentation.
* Reliability – the application should be reliable and not prone to crashes or errors.

## Software Requirements

Operating System: Windows 10 minimum supported OS version or higher OR macOS Big Sur 11.0 or higher.

IDE: Visual Studio 2022 or earlier

Language: Visual Basic

## Hardware Requirements

ARM64 or x64 processor; Quad-core or better recommended. ARM 32 processors are not supported.

Minimum of 4 GB of RAM.

A screen resolution of 1280 x 800 or higher.

### Proposed UI

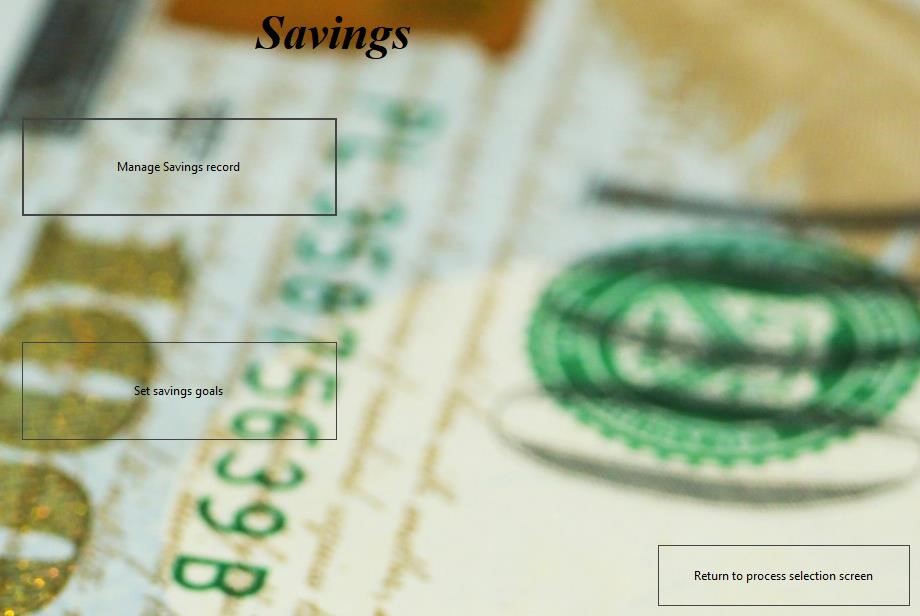
The images shown below are a simplified display of the application in action, and are neither final nor indicative of the overall quality of the program:

### i) Start screen showing user registration and access authentication elements

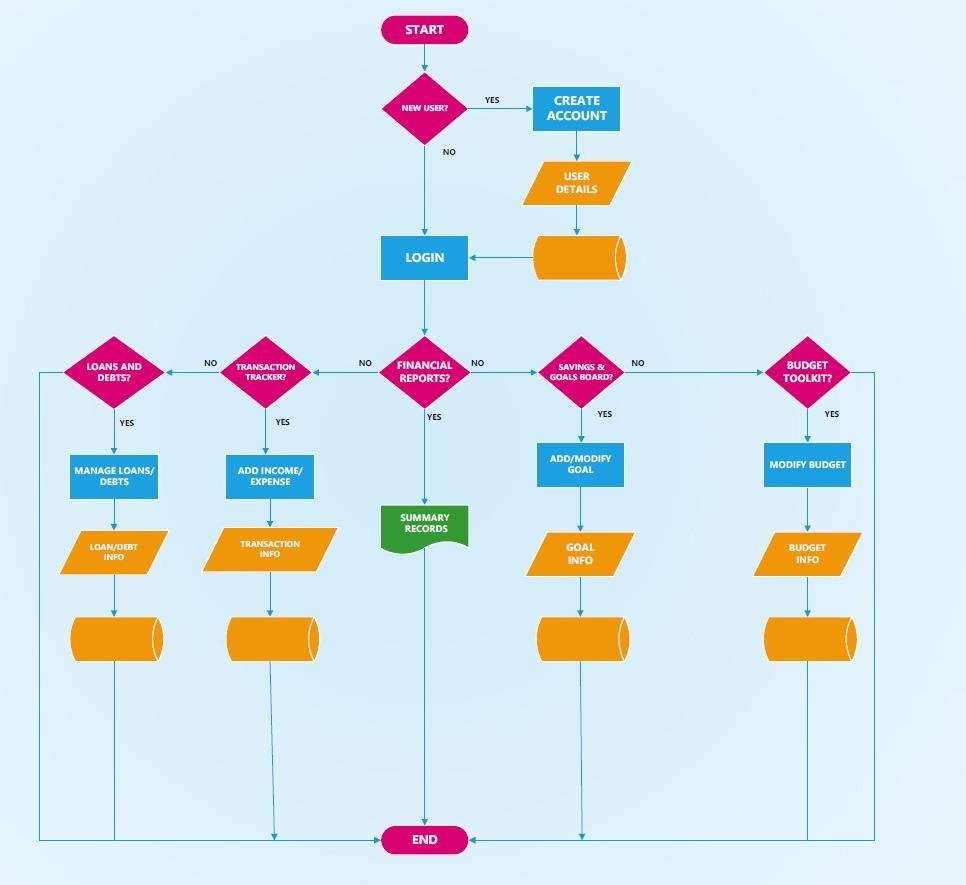
ii)

**Savings creation screen**





# Flowchart of proposed system



**IMPLEMENTATION DETAILS**

A further explanation of the application’s module features is as shown below:

**User Registration and Authentication**

This will be implemented in 2 different stages;

1. Registration(for new users)
2. Login(for returning users)

Registration functionality

The registration stage will require several inputs that will be stored and used to create a user account. These inputs include;

* Full official name
* Username
* Email
* Password

The inputs will require validation in their entry fields to avoid invalid input formats and to make input data more secure. For validation, these measures should be followed;

* Ensuring all required(non-optional) input fields are filled before submission. User should not be able to proceed without filling these fields to avoid missing important data.
* Validating email format – at base form level, the 3 main parts of an email address, which are username, the @ symbol and domain name, should be confirmed if present and if in correct sequence. Additional checks can be performed to check validity of special characters and character length restrictions. On a lower level, domain names should be verified using DNS(Domain Name System) checks to check if the addresses actually exist.
* Checking password strength – involves setting lower limits to password character lengths and encouraging varying character combinations to enhance strength and complexity of passwords and avoid situations of accounts with matching keys/hashes.
* Checking if account already exists – queries should be performed to confirm whether the username or email already exists in the database. If no match is found, a new account should be created and user details recorded. Otherwise, the user should be redirected to the login page to access their existing account.

If any of the validation conditions are not met, error-handling measures like try...catch blocks and exception throws should be used to notify the user of invalid credentials and database mismatches.

Passwords should be reconfirmed for similarity to confirm the user’s password choice as it is a sensitive and semi-permanent input. Once confirmed, passwords should be stored securely using one-way hashing algorithms before storing them in the database in an encrypted form to avoid personal account access from external sources(attackers/hackers).

After these steps have been conducted successfully, all user information should be saved in the database and proceed with account creation.

Login functionality

Just like in registration, inputs should be filled where required. However, the authentication process is slightly different and is performed as follows;

* Retrieve user information from the database based on the provided username/email address.
* Hash the entered password using the same algorithm used during account registration then compare the hashed password entered with the hashed one stored in the database.
* If they match, authenticate the user and grant access to their user account.

Error handling should also be implemented to inform the user of invalid usernames or non-matching passwords.

Aside from authentication, most other techniques used in the registration stage can carry on to this stage since they are mostly similar.

Optional functionalities for the registration/login module that can be implemented after the base is complete include;

* Implementing temporary account lockout mechanisms after multiple failed login attempts.
* Offer a password-reset functionality in a case of forgotten account details. Can be done through email redirects or storing security words/codes during account registration for later confirmation during account recovery.

**Transaction management**

The application aims to keep track of each and every transaction carried out by the user. Each transaction has a transaction ID, transaction date, amount sent, purpose (say family/friends, taxes, loans/debts or general) and transaction type. In addition, the application can provide a summarized version (say a generalized daily/monthly/yearly summary) or an in depth, individual look at the user’s transactions.

Transactions within the system are categorized into two for simplicity, those being withdrawals and deposits.

Deposits affect the system in a much more indirect manner, that being they can be treated as part of income, or sent directly to the savings account. The deposits that are treated as part of income can be used to append the budget for the month/desired time period (either increase it or decrease it) or can be simply treated as part of the user’s income.

Withdrawals affect the system more directly, as they are usually a majority of a user’s transactions, and therefore affect the user’s budget, income, financial goals and even savings. The system tracks each withdrawal from the system, and updates each of the affected fields appropriately. Say, for example, when a user transacts generally, they transact from their income and what they have withdrawn is directly reflected onto the budget they have set for themselves.

This in turn affects their financial goals. Whenever a user reaches their budget cap/limit, the system warns them against further usage as it will reflect negatively onto their financial goal setting. Withdrawals can still be done using the savings account, and reflect onto the user’s saving goals.

**Loan and Debt management**

A user can add a loan that they have taken to their account; giving details on the amount in cash or its equivalent in the case of an asset, the person, business or organization loaned from, the start date and the interest rate.

The user can also add entries showing the payments made to the loaner and basing on the expected date to finish payment, the system can track the progress of the user in paying back the loan.

With regards to amounts loaned, the user can make use of the feature to record a debtor and their details such as contact and name along with more details of the debt itself including the date loaned, the expected dates to finish payment of the loan and the progress made so far in payment. Subsequent payments are also recorded through the application.

Calculation of the total amount loaned and to be paid can be generated making the application very helpful in offering the user’s loans and debts situation at a glance from the dashboard.

**Financial Goal Setting**

A user has the ability to set a financial goal by specifying the goal amount, target date, and any other relevant details.The user can then view their financial goals along with their current progress.

The user is also allowed to manipulate their current goals i.e through updating or deleting their goal. In updating, the user can e.g modify the target amount or change the target date.

The application will also have a Track Progress Function: This function calculates and displays the user's progress toward each financial goal, showing how much they've saved or invested compared to their target. This also goes hand in hand with a Goal Analysis Function: This function provides users with insights and analysis on their financial goals, such as whether they're on track to meet them based on their current saving/investment rate.

These functions together would create a comprehensive financial goal setting module within a personal finance tracker application, helping users set realistic goals, track their progress, and achieve their financial goals.

**Budgeting**

A personal budget is a financial plan prepared to track the income and expenses for a specific period, usually weekly or monthly. It often includes a portion dedicated to saving money or investing for future goals, such as emergencies, education, or retirement.

1. Recording user income

Records how much money is coming in and when. If user doesn’t have a regular income, they can input an average amount.

Makes a list of all the money coming in, including:

* how much
* how often (weekly, fortnightly, monthly or yearly)

This money could be from the users’ wages, pension or payment, or income from investments.

2. Adding up expenses

Expenses are the essential items needed to be paid for to live. These include:

Fixed expenses, for example:

* rent or mortgage payments
* electricity, water bills
* household expenses like food and groceries
* medical costs and insurance
* transport costs like car registration or public transport
* family costs like child care and school fees

Debt expenses, for example:

* personal loan repayments
* credit card payments
* mortgage repayments

Unexpected expenses, for example:

* car repairs and services
* medical bills
* extra school costs

Expenses are deducted from the income to remain with surplus that is used for spending and savings.

3. Setting spending limits

The money left after expenses is the spending money for 'wants' such as entertainment, eating out and hobbies and saving money.

User can make a plan for what they want to do with their spending money. This will help them see where it goes and keep within their spending limit.

4. Setting a savings goal

The budget can be used to work towards a savings goal.

Once how much money the user has for 'wants' is calculated, how much the user would like to save can be worked out.

5. Budget adjustment

The budget needs to work for the user and their lifestyle so it is important to be able to adjust the budget as things change. For example, if expenses start to increase a user may need to reduce their spending, or change their savings goal. Or a user might be able to save more if income is increased or some debt is paid off.

**SAVING RECORD**

1. Emergency Fund

**Monitoring Savings**: Keep a dedicated section for user’s emergency fund. Regularly update it with contributions and withdrawals. This ensures that the users have a safety net for unexpected expenses, like medical emergencies or car repairs.

2. Investments

**Portfolio Tracking:** For those involved in investing, maintain a record of user’s investment accounts. Track the performance of individual investments, note any dividends received, and periodically rebalance user’s portfolio as needed.

3. Tax Planning

**Income and Deduction Records:** Keep a detailed record of all user’s income sources, including employment income, dividends, and interest. Additionally, document potential tax deductions such as charitable contributions, business expenses, and medical expenses. This aids in accurate tax preparation and potentially lowering user’s tax liability.

4. Financial Security

**Net Worth Calculation**: List all user’s assets (real estate, investments, savings) and subtract user’s liabilities (mortgages, loans, credit card debt). Regularly update this calculation to track user’s net worth over time. This provides a holistic view of user’s financial health.