Ledger System

Introduction

A ledger is a book or digital record used to keep track of financial transactions. It serves as the central repository where all financial data is recorded, organized, and categorized for a business or individual. The ledger is typically used to record information such as debits, credits, account balances, and other financial activities.

Problem Statement

With the rising cost of living and increasing financial pressures, many people struggle to manage their finances effectively, often facing financial instability before the end of the month.The goals is to develop a ledger system with features like user registration, login, recording debits and credits, and viewing account balances, you need to consider several technical aspects, from the database design to the implementation of secure authentication methods.

Basic Requirement (8 marks)

Registration and Login (1 mark)

Allow new users to create an account by providing their name, email, and a password. Authenticate existing users with their email and password. (Adding additional information is also allow)

- **User Registration Validation**:
    - Ensure that the email entered is in the correct format (e.g., `name@example.com`).
    - Check that the password meets complexity requirements (e.g., minimum length, special characters).
    - Confirmation password
    - Ensure that the name is alphanumeric and does not contain special characters.

```
== Ledger System (name can created yourself) ==
Login or Register:
1.Login
2.Register

>2

== Please fill in the form ==
Name: Yeah Jing Ze
Email: jingze@gmail.com
Password: 123abc

Register Successful!!!

Login or Register:
1.Login
2.Register

>1

== Please enter your email and password ==
Email:jingze@gmail.com
Password:123abc

Login Successful!!!
```

Record Debit and Record Credit (1 mark)
Enable users to add a debit and credit entry with a description, date, and amount. (Date is automatically recorded)

- **Transaction Validation**:
    - Ensure that the transaction amount is a positive number and does not exceed a certain limit.
    - Validate the transaction date to ensure it's not set in the future.
    - Check that the transaction description does not exceed a specific length (e.g., 100 characters).

```
== Welcome, Yeah Jing Ze ==
Balance: 0.00
Savings: 0.00
Loan: 0.00

== Transaction ==
1.Debit
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout

>1


== Debit ==
Enter amount: 1300.50
Enter description: Living Expense

Debit Successfully Recorded!!!



== Welcome, Yeah Jing Ze ==
Balance: 1300.50
Savings: 0.00
Loan: 0.00

== Transaction ==
1.Debit
```

```
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout

>2


== Credit ==
Enter amount: 240.57
Enter description: Rental Fee

Credit Successfully Recorded!!!
```

Savings (1 mark)

Allow users to choose the percentage of debit to be saved in savings on the next debit. By the end of the month, the savings should auto transfer to their balance.

```
== Welcome, Yeah Jing Ze ==
Balance: 1059.93
Savings: 0.00
Loan: 0.00

== Transaction ==
1.Debit
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout

>4


== Savings ==
Are you sure you want to activate it? (Y/N) : Y
Please enter the percentage you wish to deduct from the next debit: 10


Savings Settings added successfully!!!
```

View Account Balance (0.5 mark)

Display the user's current balance based on all debit and credit transactions based on month. It will be automatically calculated if any new debit or credit is recorded.

```
== Welcome, Yeah Jing Ze ==
Balance: 1059.93
Savings: 0.00
Loan: 0.00


== Transaction ==
1.Debit
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout
```

View Transaction History (1 mark)

Enable users to view a history of all their debits and credits based on month. Also enable users to export reports as CSV files. (It should remember previous transaction history from the last session.)

```
== Welcome, Yeah Jing Ze ==
Balance: 1059.93
Savings: 0.00
Loan: 0.00

== Transaction ==
1.Debit
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout

>3

== History ==
Date            Description         Debit         Credit         Balance
8-10-2024       Living Expense      1300.50                      1300.50
```

```
25-10-2024       Rental Fee                          240.57         1059.93
File Exported!
```

Logout (0.5 mark)

Able users to logout from their account.

```
==
Welcome, Yeah Jing Ze ==
Balance: 1059.93
Savings: 0.00
Loan: 0.00

== Transaction ==
1.Debit
2.Credit
3.History
4.Savings
5.Credit Loan
6.Deposit Interest Predictor
7.Logout


>4

Thank you for using "Ledger System (name can created yourself)"
```

Credit Loan (1 mark)

- Two option : Apply and Repay
- Apply
    - Allow users to apply for credit loans with a specified principal amount, interest rate, and repayment period.
    - Calculate the total repayment amount based on the principal and interest rate.
    - Schedule repayments with monthly or periodic installments.
- Repay
    - Allow users to repay the credit loans they own.
- If the loan is not paid within the period, debit and credit is not allowed.

Deposit Interest Predictor(1 mark)

- Calculate interest earned over a specified period (e.g., daily, monthly, or annually).
- The predictor is just to show the earned interest on their balance based on the bank interest.

Bank	Interest Rate(%)
RHB	2.6
Maybank	2.5
Hong Leong	2.3
Alliance	2.85
AmBank	2.55
Standard Chartered	2.65

Calculation : Monthly Interest = (Deposit x Interest Rate) / 12

Data Storage (1 mark)

Store each data in its respective CSV file (if using database is acceptable)
Database is a collection of information that organizes data in prede ned relationships where data is stored in one or more tables (or "relations") of columns and rows, making it easy to see and understand how different data structures relate to each other. Relationships are a logical connection between different tables, established on the basis of interaction among these tables. You may use Oracle Database, MySQL, Firestore or other relational databases to do so

Example of CSV should have: **it's not necessary to follow all**

User CSV

- `user_id`: Primary key, auto-incremented.
- `name`: String, stores the name of the user.
- `email`: String, stores the unique email of the user.
- `password`: String, stores the hashed password of the user.

Transactions CSV

- `transaction_id`: Primary key, auto-incremented.

- `user_id`: Foreign key linked to the Users CSV.
- `transaction_type`: String, ('debit', 'credit'), defines whether it's a debit or credit transaction.
- `amount`: Decimal, stores the amount of the transaction.
- `description`: String, details about the transaction.
- `date`: Date, records when the transaction was made.

Savings CSV

- `Savings_id`: Primary key,auto-incremented
- `User_id`: Foreign key linked to the Users CSV.
- `Status`: String,define whether it is active or not
- `Percentage`: Integer,the percentage that will be deduct from the next debit

Loans CSV

- `loan_id`: Primary key, auto-incremented.
- `user_id`: Foreign key referencing the User CSV.
- `principal_amount`: Decimal, the initial loan amount.
- `interest_rate`: Decimal, annual or monthly interest rate as a percentage.
- `repayment_period`: Integer, number of months or days to repay the loan.
- `outstanding_balance`: Decimal, remaining amount to be repaid.
- `status`: String ('active', 'repaid'), indicates the loan status.
- `created_at`: Timestamp, records the date of loan approval.

Bank CSV

- `bank_id`: Primary key, auto-incremented
- `bank_name`: String , name of the bank
- `interest_rate`: Decimal, annual or monthly interest rate as a percentage.

Account Balance CSV (Optional: Can be dynamically calculated)

- `user_id`: Foreign key linked to the Users CSV.
- `balance`: Decimal, stores the current balance of the user.

Extra Feature (4 marks) **choose only 4**

1. **Filtering and Sorting on History (1 mark)**
   Enable users to filter and sort their transaction history based on different criteria to easily analyze their financial activity.

   - **Filtering Options**:
     - **Date Range**: Users can filter transactions within a specific date range (e.g., last week, last month).
     - **Transaction Type**: Filter by debit or credit transactions.
     - **Amount Range**: Filter transactions that fall within a specified amount range.

   - **Sorting Options**:
     - **By Date**: Sort transactions from newest to oldest or vice versa.
     - **By Amount**: Sort transactions from highest to lowest amount or vice versa.

2. **Graphic User Interface(GUI) (1 mark)**
   Design a user-friendly graphical interface to allow users to interact seamlessly with the ledger system.

3. **Password Hashing (1 mark)**
   Implement secure password hashing to protect user passwords. Passwords should never be stored in plain text; instead, use a hashing algorithm like bcrypt.

   - **Hashing Function**: Use bcrypt to hash passwords during registration and compare hashed passwords during login.

4. **Data Visualization with Graphs (1 mark)**
   - Display spending trends, savings growth, and loan repayments over time using graphs or charts.
   - Include pie charts or bar graphs to show spending distribution by category.

5. **Reminder System for Loan Repayment (1 mark)**
   - Implement a notification feature to remind users of upcoming loan repayments.
   - Alerts can be sent when they login into their account.

Contact Me

If you need more clarifications about this assignment, can contact me **Yeah Jing Ze**, through

- WhatsApp (+6018-9874098)
- Email (23004995@siswa.um.edu.my)