BANK ACCOUNT MANAGEMENT SYSTEM

1. Introduction

The Bank Account Management System is a C-based application designed to demonstrate the use of random-access files for storing and managing structured data efficiently. The system allows users to perform common banking operations such as creating accounts, updating balances, deleting records, and generating account reports.

Things to be added:
A foundational program intended to teach random-access file handling in C.
An improved and more secure system that extends the basic version with additional features such as authentication, transaction logging, withdrawals, interest calculation, and input validation.

The basic program focuses on demonstrating how structured records can be stored and accessed directly from a binary file. It provides minimal functionality and assumes correct user input.

This improves reliability, usability, and safety by adding real-world banking features and defensive programming techniques.

3. Data Structure Used

Each bank account is represented using a structure containing account number, customer name, and balance. Each structure is stored as a fixed-size record, enabling direct access to any account.

4. File Organization and Storage

The program uses a binary random-access file named credit.dat. Records are accessed using the formula:
(accountNumber - 1) * sizeof(struct clientData)

5. Functional Modules already available:

- enterChoice: Displays menu options.
- newRecord: Creates a new account.
- updateRecord: Updates account balance.
- deleteRecord: Deletes an account.
- textFile: Generates a formatted text report.

6. Additional Features:

I have included password authentication, withdrawals, interest calculation, account search, listing all accounts, and transaction logging.

7. Input Validation and Error Handling

This validates user input, confirms critical operations, and prevents invalid data entry.

8. Output Files

credit.dat – Stores account records.
accounts.txt – Printable account report.
transactions.log – Transaction history.

9. Conclusion

This system extends a basic academic example into a secure and reliable banking application demonstrating practical file handling and data management in C.