Short Report on Banking System Code

Overview:

The program implements a console-based Banking System using C++. It supports:

- Account creation, viewing, deletion
- Deposits, withdrawals, and fund transfers
- Admin functionalities like listing users and checking logs

Files are used to persist user data across sessions.

Core Data Structure:

struct Account:

- accountNumber: Unique identifier for the account
- name: Name of the account holder
- email: Email ID
- contact: Phone number
- balance: Floating-point balance
- password: Used for login verification

Function-wise Breakdown:

main():

- Initializes the program.
- Loads all account numbers from database.bin into a set<int> database.
- Displays a user menu repeatedly.
- Calls the appropriate function based on user input.

createAccount():

- Handles new user registration.

- Checks for duplicate accountNumber.

- Takes input and stores data in separate files.

- Updates database.bin and logs the event.

viewAccount():

- Authenticates user via validityCheck().

- Displays account info after validation.

- Logs the viewing activity.

deposit():

- Authenticates the user.

- Adds deposit amount to balance and updates file.

- Logs the deposit transaction.

withdraw():

- Validates account and checks sufficient funds.

- Deducts amount and updates balance.

- Logs the withdrawal.

transaction():

- Authenticates sender.

- Verifies recipient and funds availability.

- Transfers funds and updates both balances.

- Logs the transaction.

deleteAccount():

- Authenticates user.
- Confirms and deletes account files.
- Marks the account as deleted in database.bin.
- Logs the deletion.

listAccounts():

- Admin-only function.
- Lists all user account details excluding sensitive data.

showLog():

- Admin-only function to display activity logs.

Utility Functions:

- dynamicFileName(): Constructs public file names.
- sensitiveContent(): Constructs sensitive file names.
- validityCheck(): Verifies account existence and password.

File Design Summary:

database.bin - Stores all account numbers (binary)

account_<id>.txt - Stores public user info (text)

sensitive_<id>.bin - Stores balance and password (binary)

log.txt - Records all activities (text)

Conclusion:

This system demonstrates basic banking operations using file-based storage and user authentication.

It effectively applies modular programming and data validation.

While not suitable for real-world deployment due to lack of encryption and database, it is an excellent learning project.