Design Summary for UVU Bank Account Management System:

Overview
The UVU Bank Account Management System has been expanded to manage multiple customer accounts. This phase introduces the ability to create an unlimited number of accounts, search for accounts by name, and perform transactions on individual accounts.

Design Components
1. IAccount Interface
2. Account Class
3. SavingsAccount, CheckingAccount, and CDAccount Classes
4. AccountManager Class
5. Program (Driver) Class
6. AccountState and AccountType Enumerations

IAccount Interface
The IAccount interface remains largely unchanged, providing the contract for account operations.

Account Abstract Class
The Account class continues to serve as an abstract base class implementing the IAccount interface. It now includes a protected virtual MinimumBalance property that can be overridden by derived classes.

SavingsAccount, CheckingAccount, and CDAccount Classes
These classes inherit from the Account class, each with specific implementations:
- SavingsAccount: No service fee, minimum balance of $100
- CheckingAccount: Default service fee of $5.00, minimum balance of $100
- CDAccount: Default service fee of $8.00, minimum balance of $500

AccountManager Class
The AccountManager class has been significantly updated to use a Dictionary for storing multiple accounts:

Fields:
- private Dictionary<string, List<IAccount>> accountsByName

Methods:
- bool StoreAccount(IAccount account)
- List<IAccount> FindAccounts(string name)
- bool HasExistingAccount(string name

Program Class
The Program class has been expanded to provide a more robust menu-driven interface:

New Methods:

- static void CreateAccount(AccountManager manager)
- static void SearchAccount(AccountManager manager)
- static void DisplayAccountInfo(IAccount account)
- static void DepositToAccount(IAccount account)
- static void WithdrawFromAccount(IAccount account)

Workflow:

1. Main Menu: Options to create account, search account, or exit
2. 2. Create Account:
    - Prompt for account details and type
    - Check for existing accounts with the same name
    - Create appropriate account object
    - Store account in AccountManager

3. Search Account:

- Find account(s) by name
- If multiple accounts found, allow user to select one
- Display account info
- Offer deposit/withdraw options

4. Deposit/Withdraw: Pe-rform transactions on selected account

5. Display updated account info after each transaction

Key Changes and Additions:

1. Use of Dictionary in AccountManager to store multiple accounts for each name
2. Ability to create unlimited number of accounts
3. Search functionality by account holder's name
4. Support for multiple accounts per account holder
5. Enhanced error handling and input validation
6. More interactive user interface with deposit and withdrawal options after account search

Conclusion

This updated design significantly expands the functionality of the banking system. It now supports creating and managing multiple accounts for each customer, provides a robust search functionality, and offers a more interactive user experience. The use of a Dictionary in the AccountManager class allows for efficient storage and retrieval of accounts, improving scalability. The system is now more flexible and can accommodate future enhancements more easily.