Project Title: Banking System

Overview

In this project, we will create a **mini banking system** in Java. Our application will enable a user to deposit and withdraw money from an account, while using a nice interface in the console.

Requirements

1. **Interface**
    - Create an interface (`Transactable`) with two methods:
        - `boolean deposit(double amount)`
        - `boolean withdraw(double amount)`.
    - This interface will be implemented by any class that handles basic transaction operations.
2. **Custom Exception**
    - Define a custom checked exception class (`InsufficientFundsException`).
    - When the user attempts to withdraw more money than the current balance, the exception should be **thrown**.
3. **Account Class**
    - Implement the **interface**.
    - Contain at least:
        - A String field for the account **name**
        - A double field for the **balance** .
    - The `deposit()` and `withdraw()` methods should:
        - **Validate** inputs (avoid negative deposits or withdrawals).
        - Throw the **custom exception** if a withdrawal exceeds the balance.
    - Optionally include getters for balance and account name.

4. **Main Class**
    - Prompt the user with a **menu** to choose actions:
        - **Deposit**
        - **Withdraw**
        - **Show Balance**
        - **Exit**
    - Use **control flow** (`while` loop, `switch` statement, etc.) to handle user choices.
    - **Catch** and handle your custom exception. For example, if a user tries to withdraw more money than the balance, display an error message and continue the program gracefully (different messages for different errors)
    - Implement **input validation** (catching `InputMismatchException` for non-numeric inputs).

---

5. **Extras (recommended, not hard just more):**
    - Create another class that holds different accounts and have another level of choosing the account wanted.
    - Create another class that will represent a bank, will implement also the interface, but in addition to the normal “account” he will have a list of accounts that owes him money.
    - Create a class that will take the data in and from files, like a database. Meaning, at the start of each request we will “fetch” the file and read his data (don’t forget, its a bank account so make sure its encrypted - Serialized)