Objective:

Debug and refactor an existing C# console application that simulates a simple banking system. The provided code has several bugs and is poorly structured. Your task is to fix the bugs, ensure the application runs correctly, and then refactor the code to improve readability, efficiency, and adherence to best practices.

Provided Code:

You will be given a C# project with the following functionality:

1. Adding new accounts.
2. Depositing money into an account.
3. Withdrawing money from an account.
4. Displaying account details.

Known Issues:

1. Bugs causing incorrect balances after transactions.
2. Errors when trying to display account details.
3. Code not following best practices (e.g., lack of encapsulation, poor naming conventions, inefficient algorithms).

Tasks:

1. **Debugging:**
    - Identify and fix all bugs that prevent the application from running correctly.
    - Ensure that all functionality (adding accounts, depositing, withdrawing, and displaying account details) works as intended.
2. **Refactoring:**
    - Improve the code structure by applying object-oriented principles.
    - Use appropriate design patterns where necessary (e.g., Repository Pattern for data handling).
    - Rename variables and methods to be more descriptive and follow naming conventions.
    - Optimize any inefficient algorithms or redundant code.
    - Add comments and documentation to make the code easier to understand.
3. **Testing:**
    - Write unit tests to verify that the fixed and refactored code works correctly.
    - Ensure the tests cover various edge cases and scenarios.

Bonus:

- Implement additional features such as account transfers or transaction history.
- Add exception handling and validation to ensure robust input handling.

Submission Guidelines:

1. Create a GitHub repository and push your code there. Share the link with us.
2. Include a README file that explains how to set up and run your application, as well as any assumptions you made.
3. Provide instructions on how to run the unit tests.
4. Your code will be evaluated based on correctness, code quality, design patterns usage, problem-solving skills, and attention to detail.