Case Study: Concurrent Money Transfer System

Build a simple concurrent money transfer system in Go that:

- Allows users to transfer money between each other.
- Ensures atomic updates to balances to prevent race conditions.
- Prevents overdrafts (users cannot send more money than they have).
- Provides a basic HTTP API to initiate transfers.

Requirements

- Use concurrency-safe data structures and explain your locking strategy in a brief write-up.
- Handle basic error cases, such as invalid users, insufficient funds or transfers to the senders own account etc
- The implementation should be tested and working software.
- Provide a ReadMe file detailing to another engineer how to set up the system.
- Setup initial balances: Mark has `$100`, Jane has `$50`, and Adam has `$0.`