**BLOCKCHAIN LEDGER SYSTEM**

**RESULTS DOCUMENT**

By: Tofik Mussa

Submitted to: Eric Gieseke

1. **Command to run the program**

javac com/cscie97/ledger/\*.java com/cscie97/ledger/test/\*.java

java -cp . com.cscie97.ledger.test.TestDriver ledger.script

java -cp . com.cscie97.ledger.test.TestDriver myledger.script

Please review the file I attached named what-to-expect-when-running-scripts to see where I am going with myledger.script

1. **Deviations from the design document**

* The biggest deviation I have made from the design document is the ability of each block to process transactions before the hash for that specific block is computed. With this approach, I have also updated the creation of accounts and updating of their balances in real time. My objective was to model a real world blockchain application where the validation by Proof of Work happens only after a block is ready to be added to the chain. The balances of the accounts in the current block are updated after every transaction. This may or may not be a good idea and I would like to hear your thoughts on this during feedback. The catch is unless and otherwise the hash for the current block is computed and validation is successfully performed, the ledger won’t move to the next block.
* I used transactionOrderIdentifier and transactionId during implementation. transactionOrderIdentifier is what is input from the script to identify the order of transactions. However, transactionId is a UUID that gets generated internally once a transaction passes all of the validation. If a transactionId is not generated I can assume that there was some validation error which gets logged to the console.
* In my implementation, trying to retrieve something such as an account that doesn’t exist doesn’t throw exceptions. However, trying to perform a task like processing a transaction or creating a duplicate account results in an error message being returned.

1. **Approach**

* I have tried my best to not let the blocks be tampered somehow. Most of the setters are private or completely removed unless there is an actual need to expose some of the attributes to another class. Throwing exceptions when the setter methods are called once they have been set already might be doable too.
* I also gracefully caught all exceptions so that they won’t keep me from processing the next command. If the command is for say processing a transaction, an invalid transaction won’t have UUID assigned and the ledger will move to processing the next transactions.
* Exceptions are printed to the screen using a logger and the CommandProcessor just outputs the line number of what is failing.
* When there is a request to print the current block’s information, I print the accounts with their balances as well as the UUID that got generated for the transactions. I thought the more details the better.
* The bootstrap mechanism initializes the Ledger by creating the genesis block as required and initializes master. When a master is paying someone, my approach was not to deduct the fee from its account to begin with and not through getting reimbursed back. Master gets to send an amount for free.
* I had several helper methods for convenience and explained thoroughly how they function, and I have also tried to make their names as descriptive as possible.
* It seemed natural to have the CommandProcessor in the same folder with the TestDriver. It looks like they belong together and decouples client from backend.

1. **Findings**

* Initially, I wanted to use a serialization library to ease my development when generating the hash. It turned out I didn’t need to since the collections I was using were already serializable.

1. **Nice to haves**

* It would have been nice to incorporate some sort of authentication and authorization mechanisms to this project. I wanted to limit the level of access I grant to my users. A miner will have much more power in the real world as opposed to normal users. I enjoyed working on the project and it will be good to see this work in the next iterations.
* My utility method prints everything twice. Sorry for the inconvenience.

Overall the design document was extremely helpful and working on the project had been fun.