UDACITY

Sair

‹ Return to "Blockchain Developer" in the classroom

DISCUSS ON STUDENT HUB

Private Blockchain

REVISÃO | REVISÃO DE CÓDIGO 6 | HISTORY

Meets Specifications

Good job the project submission meets specification! well done 👍

Configure LevelDB to persist dataset

Depending on the Starter Code selected, includes the Node.js `level` library and configured to persist data within the project directory.
Good job using levelDB to persist data.

Modify the App functions to persist data with LevelDB

addBlock(newBlock) includes a method to store newBlock within LevelDB

addBlock method works perfectly ✓

Genesis block persist as the first block in the blockchain using LevelDB.

Additionally, when adding a new block to the chain, code checks if a Genesis block already exists. If not, one is created before adding the a block.

Genesis block persists as the first block upon restart of the blockchain

Modify "validation" functions

`validateBlock(height)` function to validate a block stored within levelDB. This function should get the `height` as a parameter and then retrieve the block and validate it.
The validation should verify that the `hash` stored in the block is the same as the hash recalculated.

validateBlock() function works perfectly ✅

Implement the `validateChain()` function to validate blockchain stored within levelDB.
You should retrieve the data and validate each block, also you need to validate that the `hash` of the block is equal to the next block `previousBlockHash`

Modify getBlock() function

`getBlock(height)` function retrieves a block by block height within the LevelDB chain.

Good job implementing the getBlock() method correctly 👍

Modify getBlockHeight() function

`getBlockHeight()` function retrieves current block height within the LevelDB chain.

The getBlockHeight() method works correctly.

BAIXAR PROJETO

6 COMENTÁRIOS DA REVISÃO DE CÓDIGO ›

RETORNAR