OVERVIEW

The purpose of this test assignment is to write a console application to index blocks information from the blockchain.

Required Features

Retrieve information from several endpoints of the Ethereum JSON-RPC (https://eth.wiki/json-rpc/API#json-rpc-methods) such as:

- eth_getBlockByNumber
- eth_getBlockTransactionCountByNumber
- eth_getTransactionByBlockNumberAndIndex

Tech Requirements

- .NET framework
- Can be written in VB.net or C# (preferably VB.net)
- Logging
- MySQL database

PART 1

Sign up for an Alchemy account here, https://auth.alchemyapi.io/signup and refer to the API documentation, https://docs.alchemy.com/alchemy/apis/ethereum, for examples on how to use the API.

Alternatively, sign up for Etherscan API here, https://etherscan.io/register and refer the API documentation, https://docs.etherscan.io/api-endpoints/geth-parity-proxy, for examples on how to use the API.

You will need to create an API key irrespective of whichever provider you opt to use for this test assignment

PART 2

Write a console application to index the blocks and transaction information. Required to start from Block #12100001 to Block #12100500.

For the application, follow the indexing sequence as below:

1. Get the desired block with method **eth_getBlockByNumber** (Tip: Need convert number to hex. Ex. 0xB8A1A1).
2. If block is found (exists), call **eth_getBlockTransactionCountByNumber** to get the count of transactions in the block. Insert the block record into the database
3. If count <> 0, call **eth_getTransactionByBlockNumberAndIndex** to retrieve the transaction information line by line. Insert the record into the database
4. For all above, the entire process should be logged accordingly to a text file and a timestamp and processing time should be logged in both console and the logfile.

Note: Refer DB table requirements in Appendix.

APPENDIX

Create 2 tables in MySQL database - blocks & transactions

blocks

- blockID - INT (20), primary, auto increment
- blockNumber - INT (20)
- hash - VARCHAR (66)
- parentHash - VARCHAR (66)
- miner - VARCHAR (42)
- blockReward - DECIMAL (50,0)
- gasLimit - DECIMAL (50,0)
- gasUsed - DECIMAL (50,0)

transactions

- transactionID - INT (20), primary, auto increment
- blockID - INT (20), foreign key = blocks.blockID
- hash - VARCHAR (66)
- from - VARCHAR (42)
- to - VARCHAR (42)
- value - DECIMAL (50,0)
- gas - DECIMAL (50,0)
- gasPrice - DECIMAL (50,0)
- transactionIndex - INT (20)

Note: For INT & DECIMALS column, HEX values should be converted before inserting.