UDACITY

Sair

‹ Return to "Blockchain Developer" in the classroom

DISCUSS ON STUDENT HUB

Architect a Blockchain Supply Chain Solution - Part B

REVISÃO | REVISÃO DE CÓDIGO | HISTORY

Meets Specifications

Well done! Your project meets all the requirements! Congratulations on completing this project👍👏

Write Up

Project write-up include the following UML diagrams:

- Activity
- Sequence
- State
- Classes (Data Model)

If libraries are used, the project write-up discusses why these libraries were adopted.

If IPFS is used, the project write-up discusses how IPFS is used in this project.

A general write up exists to items like steps and contracts address.

Write smart contracts with functions

Smart contract implements functions to track.

For example:

Product ID
Product UPC
Origination Information
Farm
Misc organization info
Longitude & Latitude of geo coordinates
Product notes

Ownable.sol has required functions that establish owner and the transfer of ownership.

ConsumerRole.sol has required functions that manage the consumer role.

RetailerRole.sol has required functions that manage the consumer role.

DistributorRole.sol has required functions that manage the consumer role.

Student has implemented additional roles correctly.

Test smart contract code coverage

Project contains tests for the boiler plate functions and all tests are approved without error.

Awesome! Everything works as expected. All tests pass!💯

```
  Contract: SupplyChain
    ✓ Testing smart contract function composeItem() that allows an artist to compose a musi
c (163ms)
    ✓ Testing smart contract function payRoyaltyItem() that allows a Record company to pay
royalties for music (382ms)
    ✓ Testing smart contract function createMusicMix() that allows a record company to mix
the music (252ms)
    ✓ Testing smart contract function createMusicMaster() that allows a record company to m
aster music (445ms)
    ✓ Testing smart contract function sellItem() that allows a distributor to buy the item
 (409ms)
    ✓ Testing smart contract function buyDiscsAndCases() that allows a distributor to buy c
ases and package for item (117ms)
    ✓ Testing smart contract function packItem() that allows a distributor to pack the item
(99ms)
    ✓ Testing smart contract function packItem() that allows a distributor to pack the item
(143ms)
    ✓ Testing smart contract function receiveItem() that allows a retailer to mark music re
ceived (106ms)
    ✓ Testing smart contract function purchaseItem() that allows a consumer to purchase mus
ic (174ms)
    ✓ Testing smart contract function fetchItemBufferOne() that allows anyone to fetch item
details from blockchain
    ✓ Testing smart contract function fetchItemBufferTwo() that allows anyone to fetch item
details from blockchain
```

Deploy smart contract on a public test network (Rinkeby)

Smart contract is deployed on on the Ethereum RINKEBY test network.

Project submission includes a document (.md, .txt) that includes:

- Transaction ID
- Contract address
- - Hint: You can view Transaction ID and Contract ID from a blockchain explorer (e.g. Etherscan). Example Contract ID: https://rinkeby.etherscan.io/address/0xfb0720c0715e68f80c0c0437c9c491abfed9e7ab#code

Modify client code to interact with a smart contract

Front-end is configured to:

- Submit a product for shipment (farmer to the distributor, distributor to retailer, etc).
- Receive product from shipment.
- Validate the authenticity of the product.

BAIXAR PROJETO

RETORNAR