General

#SBHACK19 This is a test case blockchain for #SBHACK19 for agriculture and food vertical. This complete test case blockchain is based on Hyperledger v.1.2 by Amazon Managed Blockchain service. The chaincode is adopted from https://github.com/aws-samples. Codes from ngo.js file are added and modified for CHAINCODE, Customer functions, Farmer functions, PP functions, Payment functions, Spend functions, SpendAllocation functions, and Ratings functions test-chaincode-aws.sh file is modified from original folder named ngo-rest-api is added without any change from https://github.com/aws-samples/non-profit-blockchain for this test project.

The blockchain code is in the file name chaincode.js. To run the code follow the procedures in https://github.com/aws-samples/non-profit-blockchain for the SOMO website frontend is developed with PHP and VisualStudio v.2012. The website link is http://simsits-001-site1.htempurl.com/SOMO/ all the files related to website development are in AgroFood/SOMO/website/.

Github link

https://github.com/AgroFood/SOMO

Description of chaincode

Chaincode contains -

1. GENERAL FUNCTIONS: Executes a query using a specific key

2. CHAINCODE:
    > Executes defined blocks

2.1 Customer functions:
    > Creates a new customer,
    > Retrieves a specfic customer,
    > Retrieves all customers,

2.2 Farmer functions:
    > Creates a new farmer,
    > Retrieves a specfic farmer,
    > Retrieves all farmers

2.3 PP functions:
    > Creates a new PP (Payment Provider),
    > Retrieves a specfic PP,
    > Retrieves all PP

2.4 Payment functions:
    > Creates a new Payment,
    > Retrieves a specfic payment,
    > Retrieves payments for a specfic customer,
    > Retrieves payments for a specfic farmer,
    > Retrieves payments for a specfic PP,
    > Retrieves all payments,

2.5 Spend functions:
    > Creates a new Spend,
    > Retrieves a specfic spend,
    > Retrieves spend for a specfic PP,
    > Retrieves all spend,

2.6 SpendAllocation functions:
    > Retrieves a specfic spendAllocation,
    > Retrieves the spendAllocation records for a specific Payment,
    > Retrieves the spendAllocation records for a specific Spend record,
    > Retrieves all spendAllocations

2.7 Ratings functions:

> Creates a new Rating,
> Retrieves ratings for a specfic PP,
> Retrieves ratings for an PP made by a specific farmer,

2.8 Blockchain related functions:
> Retrieves the Fabric block and transaction details for a key or an array of keys

Run chaincode

Deploy the chaincode in Febric Network, follow the commands below. Run all the command in AWS Cloud9 terminal

```
> ssh ec2-user@ec2-3-85-245-111.compute-1.amazonaws.com -i
~/agrofoodnet-keypair.pem

> cd ~
git clone https://github.com/AgroFood/SOMO.git

> cd ~/non-profit-blockchain/ngo-fabric
source fabric-exports.sh

> cat ~/peer-exports.sh

> source ~/peer-exports.sh

> docker inspect cli

> cd ~
mkdir -p ./fabric-samples/chaincode/chaincode
cp ./SOMO/agrofood/src* ./fabric-samples/chaincode/chaincode -r

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
   -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" -e "CORE_PEER_ADDRESS=$PEER"
\
   cli peer chaincode install -n chaincode -l node -v v0 -p
/opt/gopath/src/github.com/chaincode

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
   -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" -e "CORE_PEER_ADDRESS=$PEER"
\
```

```
    cli peer chaincode instantiate -o $ORDERER -C mychannel -n chaincode -v v0 -c
'{"Args":["init"]}' --cafile /opt/home/managedblockchain-tls-chain.pem --tls

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
    -e "CORE_PEER_ADDRESS=$PEER" -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode query -C mychannel -n chaincode -c '{"Args":["queryAllCustomer"]}'

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
    -e "CORE_PEER_ADDRESS=$PEER" -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode invoke -C mychannel -n chaincode \
    -c  '{"Args":["createCustomer","{\"customerUserName\": \"edge\", \"email\":
\"edge@def.com\", \"registeredDate\": \"2018-10-22T11:52:20.182Z\"}"]}' -o $ORDERER
--cafile /opt/home/managedblockchain-tls-chain.pem --tls

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
    -e "CORE_PEER_ADDRESS=$PEER" -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode invoke -C mychannel -n chaincode \
    -c  '{"Args":["createCustomer","{\"customerUserName\": \"braendle\", \"email\":
\"braendle@def.com\", \"registeredDate\": \"2018-11-05T14:31:20.182Z\"}"]}' -o $ORDERER
--cafile /opt/home/managedblockchain-tls-chain.pem --tls

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
    -e "CORE_PEER_ADDRESS=$PEER" -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode query -C mychannel -n chaincode -c '{"Args":["queryAllCustomer"]}'

> docker exec -e "CORE_PEER_TLS_ENABLED=true" -e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain-tls-chain.pem" \
    -e "CORE_PEER_ADDRESS=$PEER" -e "CORE_PEER_LOCALMSPID=$MSP" -e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode query -C mychannel -n chaincode -c
'{"Args":["queryCustomer","{\"customerUserName\": \"edge\"}"]}'
```