The Chinese University of Hong Kong

Department of Information Engineering

FTEC5520 –Applied Blockchain & Cryptocurrency

Lab2 Report 1

Hyperledger Fabric Setup & Practice on AWS(Task1,2,3)

1 Questions Answering:

Q1: Basic concepts and knowledge about Hyperledger Fabric. (2 mark)

i. What's the difference between member and peer node in Hyperledger fabric?

Key Difference:Members are organizations, while peer nodes are the actual machines that execute the network's operations on behalf of those organizations.

ii. What's the channel in Hyperledger Fabric? Why do we need channel?

Channel: t is used to isolate and protect data, ensuring that only authorized members can access the transactions and ledger associated with that channel. Why We Need It:Privacy Scalability Customization

iii. What's chaincode in Hyperledger Fabric? What did you do in task 1~3 about chaincode?

Chaincode: Chaincode is a program (smart contract) written in Go, JavaScript, or Java that runs on the Hyperledger Fabric network. What I Did in Task 1~3:Task 1: Developed a basic chaincode Task 2: Tested and deployed the chaincode Task 3: Invoked the chaincode to perform transactions and validated the results on the ledger.

iv. What's Membership Service Provider (MSP)? Why do we need it? What did you do about this component in task1~3?

Membership Service Provider (MSP): MSP is a component in Hyperledger Fabric that manages the identities of network participants. Why We Need It:Authentication , Authorization. What I Did in Task 1~3:Task 1: Configured MSP for the network Task 2: Used MSP to authenticate and authorize peers and clients Task 3: Validated that only authorized identities could invoke the chaincode and access the ledger.

Q2:

1) How to create blockchain network, member and node using AWS CLI instead of GUI?

Create a Blockchain Network, Create a Member, Create a Node

2) What's the CAEndpoint of here? How can we get the value in Amazon Managed Blockchain service dashboard? How to enroll an Admin User with admin permissions to your member's CA? **Hint:** `fabric-ca-client enroll`

CAEndpoint, Get CAEndpoint, In the Amazon Managed Blockchain dashboard, Enroll an Admin User

3) How to create a channel and join your peer node to the channel your created?

Create a Channel, Join Peer Node to the Channel

4) How to install, instantiate, query and invoke the Chaincode? Please use the example we used in task3.

Install Chaincode, Instantiate Chaincode, Query Chaincode, Invoke Chaincode

Q3: What's the difference between Hyperledger Fabric and Ethereum after you learnt about both the two frameworks including the hands-on lab session? (1 mark)

1. Architecture and Design

Hyperledger Fabric:Permissioned Blockchain: Only authorized participants (members) can join the network.

Consensus: Uses customizable consensus mechanisms (e.g., Kafka, Raft).

Ethereum:Permissionless Blockchain: Anyone can join the network and participate.

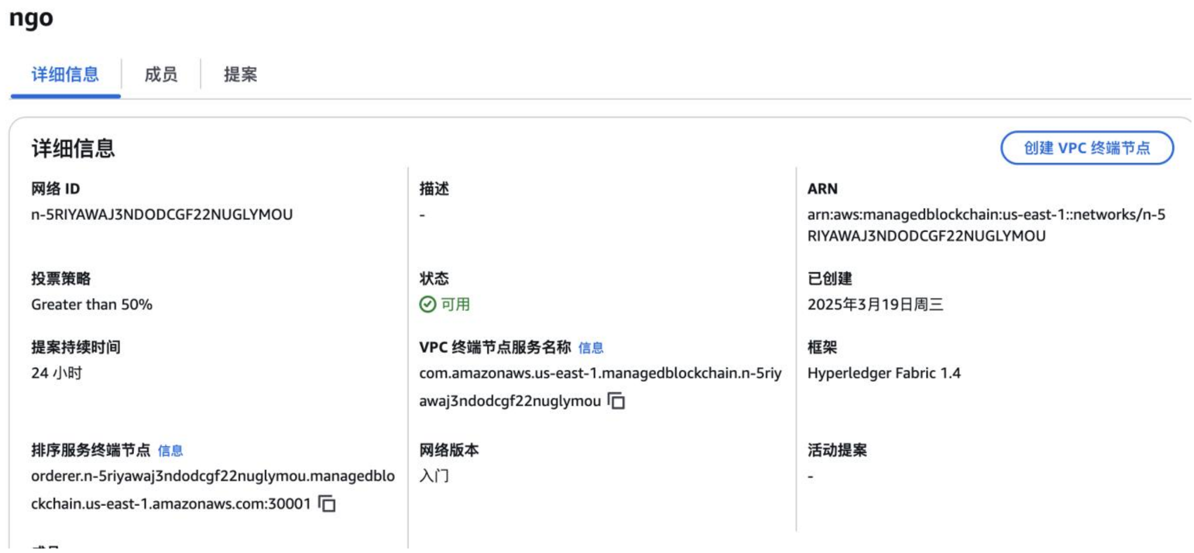
consensus: Uses Proof of Work (PoW) or Proof of Stake (PoS) for consensus.

2 Screen Capture of Main Steps:

Please replace the sample photos with your own results.

Your screenshots must include these parts at least and detailed description of each screenshots:

1 Your Hyperledger Fabric blockchain network on Amazon Managed Blockchain dashboard:



ngo

详细信息

网络 ID
n-5RIYAWAJ3NDODCGF22NUGLYMOU

描述
-

ARN
arn:aws:managedblockchain:us-east-1::networks/n-5RIYAWAJ3NDODCGF22NUGLYMOU

投票策略
Greater than 50%

状态
可用

已创建
2025年3月19日周三

提案持续时间
24 小时

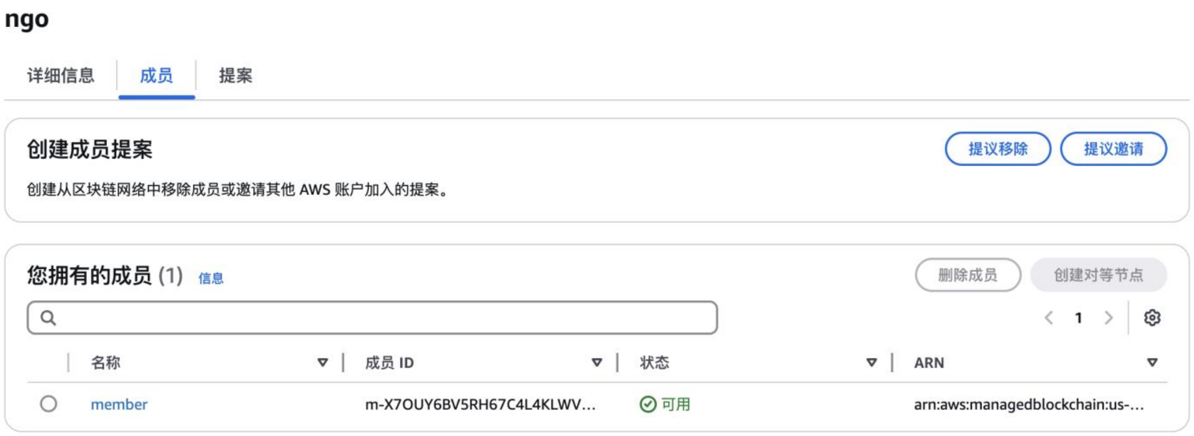
VPC 终端节点服务名称 信息
com.amazonaws.us-east-1.managedblockchain.n-5riyawaj3ndodcgf22nuglymou

框架
Hyperledger Fabric 1.4

排序服务终端节点 信息
orderer.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30001

网络版本
入门

活动提案
-



ngo

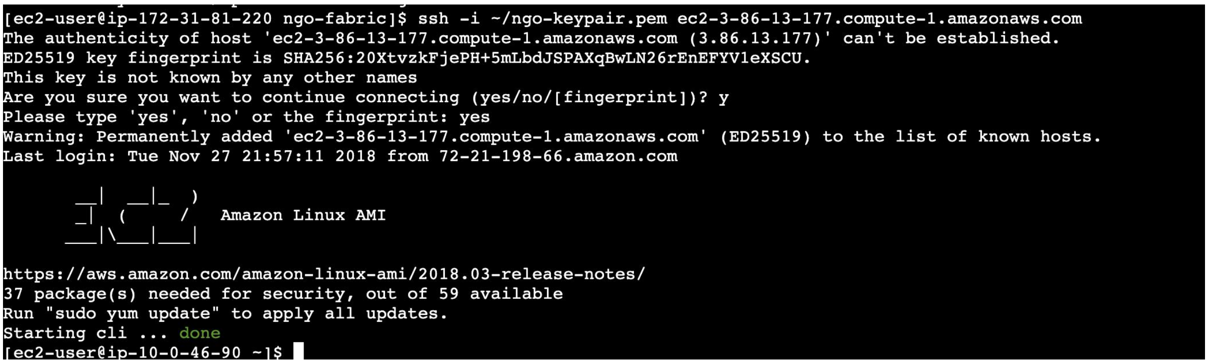
创建成员提案

创建从区块链网络中移除成员或邀请其他 AWS 账户加入的提案。

您拥有的成员 (1) 信息

	名称	成员 ID	状态	ARN
○	member	m-X7OUY6BV5RH67C4L4KLWV...	可用	arn:aws:managedblockchain:us-...

2 SSH from the EC2 IDE to peer node. You should know they are two VMs.

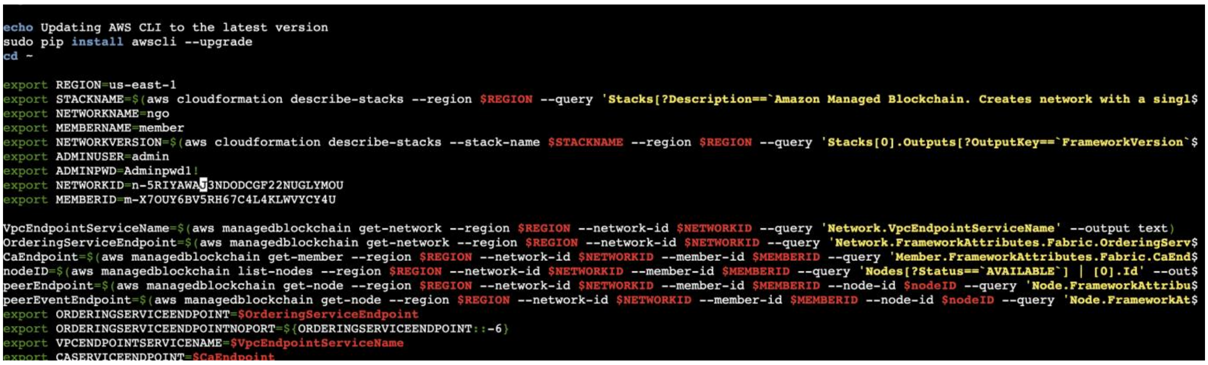


```
[ec2-user@ip-172-31-81-220 ngo-fabric]$ ssh -i ~/ngo-keypair.pem ec2-3-86-13-177.compute-1.amazonaws.com
The authenticity of host 'ec2-3-86-13-177.compute-1.amazonaws.com (3.86.13.177)' can't be established.
ED25519 key fingerprint is SHA256:20XtvzkFjePH+5mLbdJSPAXqBwLN26rEnEFYV1eXSCU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added 'ec2-3-86-13-177.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Last login: Tue Nov 27 21:57:11 2018 from 72-21-198-66.amazon.com

       __|  __|_  )
       _|  (     /   Amazon Linux AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
37 package(s) needed for security, out of 59 available
Run "sudo yum update" to apply all updates.
Starting cli ... done
[ec2-user@ip-10-0-46-90 ~]$
```

3 The content of **fabric-exports.sh** after insert your blockchain network information



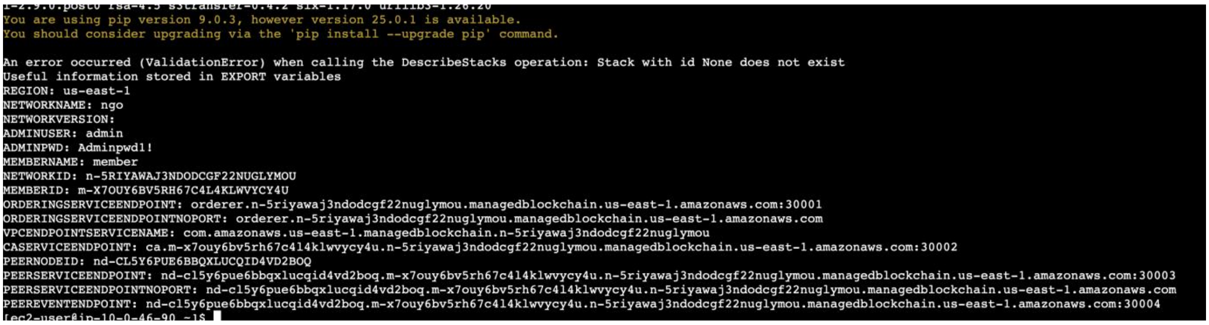
```
echo Updating AWS CLI to the latest version
sudo pip install awscli --upgrade
cd -

export REGION=us-east-1
export STACKNAME=$(aws cloudformation describe-stacks --region $REGION --query 'Stacks[?Description==`Amazon Managed Blockchain. Creates network with a singl$
export NETWORKNAME=ngo
export MEMBERNAME=member
export NETWORKVERSION=$(aws cloudformation describe-stacks --stack-name $STACKNAME --region $REGION --query 'Stacks[0].Outputs[?OutputKey==`FrameworkVersion`$
export ADMINUSER=admin
export ADMINPWD=Adminpwd1!
export NETWORKID=n-5RIYAWAJ3NDODCGF22NUGLYMOU
export MEMBERID=m-X7OUY6BV5RH67C4L4KLWVYCY4U

VpcEndpointServiceName=$(aws managedblockchain get-network --region $REGION --network-id $NETWORKID --query 'Network.VpcEndpointServiceName' --output text)
OrderingServiceEndpoint=$(aws managedblockchain get-network --region $REGION --network-id $NETWORKID --query 'Network.FrameworkAttributes.Fabric.OrderingServ$
CaEndpoint=$(aws managedblockchain get-member --region $REGION --network-id $NETWORKID --member-id $MEMBERID --query 'Member.FrameworkAttributes.Fabric.CaEnd$
nodeID=$(aws managedblockchain list-nodes --region $REGION --network-id $NETWORKID --member-id $MEMBERID --query 'Nodes[?Status==`AVAILABLE`] | [0].Id' --out$
peerEndpoint=$(aws managedblockchain get-node --region $REGION --network-id $NETWORKID --member-id $MEMBERID --node-id $nodeID --query 'Node.FrameworkAttribu$
peerEventEndpoint=$(aws managedblockchain get-node --region $REGION --network-id $NETWORKID --member-id $MEMBERID --node-id $nodeID --query 'Node.FrameworkAt$
export ORDERINGSERVICEENDPOINT=$OrderingServiceEndpoint
export ORDERINGSERVICEENDPOINTNOPORT=${ORDERINGSERVICEENDPOINT::-6}
export VPCENDPOINTSERVICENAME=$VpcEndpointServiceName
export CASERVICEENDPOINT=$CaEndpoint
```

4 The useful information after you ran the command in task3 step 2.16

```
source fabric–exports.sh
```

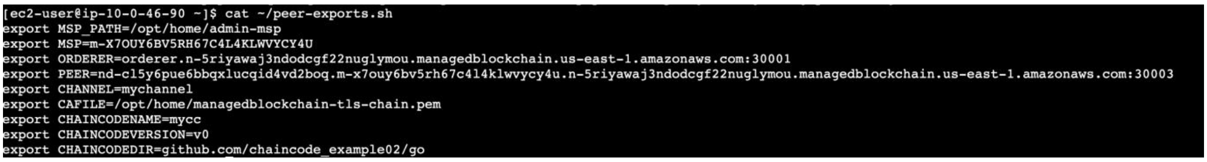


```
1-2.9.0.post0 rsa-4.5 s3transfer-0.4.2 six-1.17.0 urllib3-1.26.20
You are using pip version 9.0.3, however version 25.0.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.

An error occurred (ValidationError) when calling the DescribeStacks operation: Stack with id None does not exist
Useful information stored in EXPORT variables
REGION: us-east-1
NETWORKNAME: ngo
NETWORKVERSION:
ADMINUSER: admin
ADMINPWD: Adminpwd1!
MEMBERNAME: member
NETWORKID: n-5RIYAWAJ3NDODCGF22NUGLYMOU
MEMBERID: m-X7OUY6BV5RH67C4L4KLWVYCY4U
ORDERINGSERVICEENDPOINT: orderer.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30001
ORDERINGSERVICEENDPOINTNOPORT: orderer.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com
VPCENDPOINTSERVICENAME: com.amazonaws.us-east-1.managedblockchain.n-5riyawaj3ndodcgf22nuglymou
CASERVICEENDPOINT: ca.m-x7ouy6bv5rh67c4l4klwvycy4u.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30002
PEERNODEID: nd-CL5Y6PUE6BBQXLUCQID4VD2BOQ
PEERSERVICEENDPOINT: nd-cl5y6pue6bbqxlucqid4vd2boq.m-x7ouy6bv5rh67c4l4klwvycy4u.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30003
PEERSERVICEENDPOINTNOPORT: nd-cl5y6pue6bbqxlucqid4vd2boq.m-x7ouy6bv5rh67c4l4klwvycy4u.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com
PEEREVENTENDPOINT: nd-cl5y6pue6bbqxlucqid4vd2boq.m-x7ouy6bv5rh67c4l4klwvycy4u.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30004
[ec2-user@ip-10-0-46-90 ~]$
```

5 Use this command to check the peer export information:

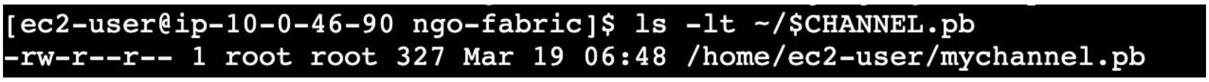
```
cat ~/peer–exports.sh
```



```
[ec2-user@ip-10-0-46-90 ~]$ cat ~/peer-exports.sh
export MSP_PATH=/opt/home/admin-msp
export MSP=m-X7OUY6BV5RH67C4L4KLWVYCY4U
export ORDERER=orderer.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30001
export PEER=nd-cl5y6pue6bbqxlucqid4vd2boq.m-x7ouy6bv5rh67c4l4klwvycy4u.n-5riyawaj3ndodcgf22nuglymou.managedblockchain.us-east-1.amazonaws.com:30003
export CHANNEL=mychannel
export CAFILE=/opt/home/managedblockchain-tls-chain.pem
export CHAINCODENAME=mycc
export CHAINCODEVERSION=v0
export CHAINCODEDIR=github.com/chaincode_example02/go
```

6 List the generated configuration using

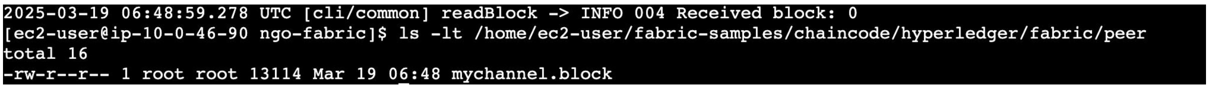
```
ls –lt ~/$CHANNEL.pb
```



```
[ec2-user@ip-10-0-46-90 ngo-fabric]$ ls -lt ~/$CHANNEL.pb
-rw-r--r-- 1 root root 327 Mar 19 06:48 /home/ec2-user/mychannel.pb
```

7 List the mychannel.block generated fabric channel in task3 step 4.2

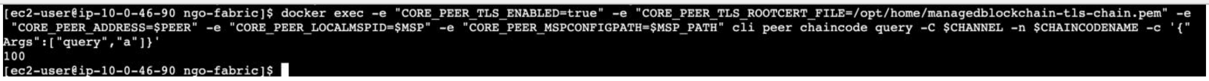
```
ls –lt /home/ec2–user/fabric–samples/chaincode/hyperledger/fabric/peer
```



```
2025-03-19 06:48:59.278 UTC [cli/common] readBlock -> INFO 004 Received block: 0
[ec2-user@ip-10-0-46-90 ngo-fabric]$ ls -lt /home/ec2-user/fabric-samples/chaincode/hyperledger/fabric/peer
total 16
-rw-r--r-- 1 root root 13114 Mar 19 06:48 mychannel.block
```

8 Query the chaincode on peer in the task3 step 8

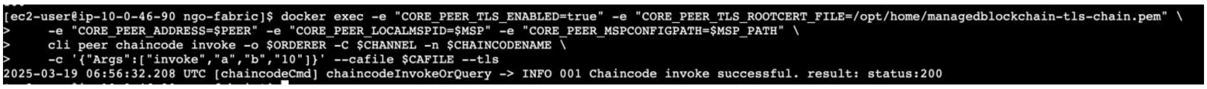
```
docker exec –e "CORE_PEER_TLS_ENABLED=true" –e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain–tls–chain.pem" –e
"CORE_PEER_ADDRESS=$PEER" –e "CORE_PEER_LOCALMSPID=$MSP" –e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" cli peer chaincode query –C $CHANNEL –n
$CHAINCODENAME –c '{"Args":["query","a"]}'
```



```
[ec2-user@ip-10-0-46-90 ngo-fabric]$ 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 $CHANNEL -n $CHAINCODENAME -c '{"
Args":["query","a"]}'
100
[ec2-user@ip-10-0-46-90 ngo-fabric]$ 
```

9 Invoke a transaction to transfer 10 dollars from Account a to Account b:

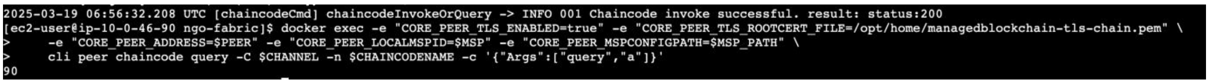
```
docker exec –e "CORE_PEER_TLS_ENABLED=true" –e
"CORE_PEER_TLS_ROOTCERT_FILE=/opt/home/managedblockchain–tls–chain.pem" \
    –e "CORE_PEER_ADDRESS=$PEER" –e "CORE_PEER_LOCALMSPID=$MSP" –e
"CORE_PEER_MSPCONFIGPATH=$MSP_PATH" \
    cli peer chaincode invoke –o $ORDERER –C $CHANNEL –n $CHAINCODENAME \
    –c '{"Args":["invoke","a","b","10"]}' ––cafile $CAFILE ––tls
```



```
[ec2-user@ip-10-0-46-90 ngo-fabric]$ 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 -o $ORDERER -C $CHANNEL -n $CHAINCODENAME \
>     -c '{"Args":["invoke","a","b","10"]}' --cafile $CAFILE --tls
2025-03-19 06:56:32.208 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
```

10 Query the balance of Account named ‘a’ again after transaction:

```
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 $CHANNEL –n $CHAINCODENAME –c '{"Args":["query","a"]}'
```



```
2025-03-19 06:56:32.208 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
[ec2-user@ip-10-0-46-90 ngo-fabric]$ 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 $CHANNEL -n $CHAINCODENAME -c '{"Args":["query","a"]}'
90
```