**P5 - Create a block chain app for loyalty points with Hyperledger Fabric Ethereum Virtual Machine. Deploy Fabric locally with EVM and create a proxy for interacting with a smart contract through a Node.js web app**

1 . Login as a root

Install go

Install docker

Install Docker compose

Install node install npm

2 . Remove all your docker containers and images

Docker stop $(docker ps -a -q)

Docker rm $(docker ps -a -q)

Docker rml $(docker images -q) -f

3 . set gopath to your go installation

Export GOPATH=#HOME/go

Cd $GOPATH

4 . Clone the fabric-chaincode-evm repo in your gopath directory

Mkdir src

Cd src

Mkdir github.com

Cd gihub.com

Mkdir Hyperledger

Cd Hyperledger

Cd $GOPATH/src/github.com/Hyperledger/

Git clone

[https://github.com/hyperledger/fabric-samples.git](https://github.com/hyperledger/fabric-samples.git 5)

[5](https://github.com/hyperledger/fabric-samples.git 5) . Checkout release-0.1

Cd fabric-samples

Git checkout release1.4

6 . now navigate back to your fabric samples folder . Here we will use firs-network to launch the network

Cd $GOPATH/src/github.com/Hyperledger/fabric- samples/first-netowrk

7 . update the docker-compose-cli.yml with the volume to include the fabric-chaincode-evm

Chmod+x docker-compose-cli.yml

Gedit docker-compose-cli.yml

…. /…/fabric-chaincode-evm:/opt/gopath/src/github.com/Hyperledger/fabric-chaincode-evm

8 . Generate certificates and bring up the network

Cd bin/

Cp crpytogen ../first-network/

Cd ..

Cd first-network/

./byfn.sh generate

./byfn.sh up

9 . navigate into the cli docker container

Cd ..

Docker exec -it cli bash

Export CORE\_PEER\_MSPCONFIGPATH=/opt/gopath/src/github.com/Hyperledger/fabric/peer/crypto/p [eerOrganizations/org1.example.com/users/Admin@org1.example.com/msp](mailto:eerOrganizations/org1.example.com/users/Admin@org1.example.com/msp)

Export CORE\_PEER\_LOCALMSPID=”org1MSP”

Export CORE\_PEER\_TLS\_ROOTCERT\_FILE=/opt/gopath/src/github.com/Hyperledger/fabric/peer/crypto /peerOrganization/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt

Peer chaincode install -n evmcc -l golang -v 0 -p github.com/Hyperledger/fabric- chaincode.evm/evmcc

10 . Initiate the chaincode

Peer chaincode instantiate -n evmcc -v 0 -C mychannel -c ‘{“Args”:[]}’ -o ordered.example.com:7050 –tls –cafile /opt/gopath/src/github.com/Hyperledger/fabric/peer/crts/tlsca.example.com-cert.pem

And exit from terminal

Exit

11 . Execute the following to set certain environment variables required for setting up fab3

Export FABPROXY\_CONFIG=${GOPATH}/src/github.com/Hyperledger/fabric-chaincode-evm/examples/first-network-sdk-config.yml

Export FABPROXY\_USER=User1

Export FABPROXY\_ORG=0rg1

Export FABPROXY\_CHANNEL=mychannel

Export FABPROXY\_CCID=evmcc

Export PORT=5000

12 . Redirect to fabric-chaincode-evm directory

Cd fabric-chaincode-evm/

Cd $GOPATH/src/github.com/Hyperledger/fabric-chaincode-evm/

13 . Run the following to build the fab proxy

Go mod init

Go mod tidy -e

Go mod vendor

Go build -o fab3 ./fabproxy/cmd

14 . You can run the proxy

./fab3

**P6 - Develop a voting application using Hyperledger and Ethereum. Build a decentralized app that combines Ethereum's Web3 and Solidity smart contracts with Hyperledger hosting Fabric and Chaincode EVM**

**Code:**

pragma solidity 0.8.17;

contract VotingForTopper{

address owner;

string purpose;

struct Voter{

bool authorized;

bool voted;

}

uintTotalVotes;

uintteamA;

uintteamB;

uintteamC;

mapping(address=>Voter) info;

constructor(string memory \_name) public{

purpose = \_name;

owner = msg.sender;

}

modifier ownerOn(){

require(msg.sender==owner);

\_;

}

function authorize(address \_person) ownerOn public{

info[\_person].authorized = true;

}

function teamAF(address \_address) public {

require(!info[\_address].voted,"already voter person"); require(info[\_address].authorized,"You have no right for Vote"); info[\_address].voted = true;

teamA++;

TotalVotes++;

}

function teamBF(address \_address) public {

require(!info[\_address].voted,"already voter person"); require(info[\_address].authorized,"You have no right for Vote"); info[\_address].voted = true;

teamB++;

TotalVotes++;

}

function teamCF(address \_address) public returns(string memory){ require(!info[\_address].voted,"already voter person"); require(info[\_address].authorized,"You have no right for Vote"); info[\_address].voted = true;

teamC++;

TotalVotes++;

return("Thanks for Voting");

}

function TotalVotesF() public view returns(uint){ return TotalVotes;

}

function resultOfVoting() public view returns(string memory){ if(teamA>teamB){

if(teamA>teamC){

return "A is Winning";

}

else if(teamC>teamA){

return "C is Winiing";

}

}

else if(teamB>teamC){

return "B is Winiing";

}

else if(teamC>teamB){

return "C is Winiing";

}

else if(teamA==teamB&&teamA==teamC || teamB==teamC){

return "No One is Winiing";

}

}

}

1 . Open terminal

Npm install -g truffle

Git clone <https://github.com/tko22/truffle-webpack-boilerplate>

Cd truffle-webpack-boilerplate

Npm install

Truffle develop

Compile

Npm run dev

Start localhost:8080

|  |  |
| --- | --- |
| **P1 - Install and understand Docker container, Node.js, Java and Hyperledger Fabric, Ethereum and perform necessary software installation on local machine/create instance on Cloud to run**  1 . Install docker on windows from docker official website  2 . Download and install WSL  3 . Setup docker  4 . Download and install Geth  5 . Download and install NodeJS  6 . Open PowerShell  7 . Enable windows SubSystem for Linux and Download and install Linux 20.04.5LTS from Windows Store  8 . Open Windows Terminal and Open Linux Tab  docker –version  docker-compose –version  sudo apt update  sudo apt upgrade  9 . Install curl from linux terminal  sudo apt update && sudo apt upgrade  sudo apt install curl  curl –version  10 . Install Golang from terminal  sudo apt install golang-go  11 . Check Go and Git Version  go version  git version  12 . Clone and install Hyperledger  curl -sSL https://bit.ly/2ysb0FE|bash -s  cd $wsl  cd go/src/github.com/fabric-samples  cd test-network  ./network.sh down  Docker images | **P2 - Create and deploy a block chain network using Hyperledger Fabric SDK for Java**  **Install curl using terminal**  sudo apt update && sudo apt upgrade  sudo apt install curl  curl –version  docker –version  nod -v  npm –version  apt install golang-go  apt-get install python3.6  curl -sSL <http://bit.ly/2ysb0FE> | bash -s 1.4.4  ls  cd fabric-samples ls  cd first-network/  ./byfn.sh generate  ./byfn.sh up -l java |
| **P3 - Interact with a block chain network. Execute transactions and requests against a block chain network by creating an app to test the network and its rules**  Git clone <https://github.com/IBM/InteractingWithABlockchainNetwork.git>  Cd InteractingWithABlockchainNetwork/  ./build.sh  Docker ps  Docker logs fitcoin-ca  Cd testApplication/  Npm install  Node index.js  After all, above command open link by index.js which is port is 8000 and do query as you wish. | **P4 - Deploy an asset-transfer app using block chain. Learn app development within a Hyperledger Fabric network**  1 . Login as user and clone below directories  Git clone <https://github.com/rshrimp/-Asset-Transfer-app-using-hyperledger-Fabric-Blockchain.git>  asset-transfer-app  2 . Redirect the cloned Directories  cd asset-transfer-app  3 . Stop and kill any stale and active container  ./scripts/cleanup.sh  4 . Create artifacts (Genesis block and channel info)  ./scripts/createArtifacts.sh  5 . Start network with following command  ./scripts/start\_network.sh  6 . Create channels, Join channels, install chaincode, populates ledgers with initial entries (New Terminal)  ./scripts/setupNetwork.sh  7 . Enter the bash shell  Docker exec -it client.org1 bash  8 . and Register a user  Fabric-ca-client register --id.name admin3 --id.type user –id.affiliation org1 –id.secret adminpw  9 . Finally enroll  Fabric-ca-client enroll -u <http://admin3:adminpw@ca.org1:7054> -M [Admin3@org1.com/msp](mailto:Admin3@org1.com/msp)  10 . Now run asset transfer request  ./scripts/createTransferRequest.sh |