Assignment 1

Name – Sati, Ankit Date – 5/18/2022
Section – 001
SID – as14128

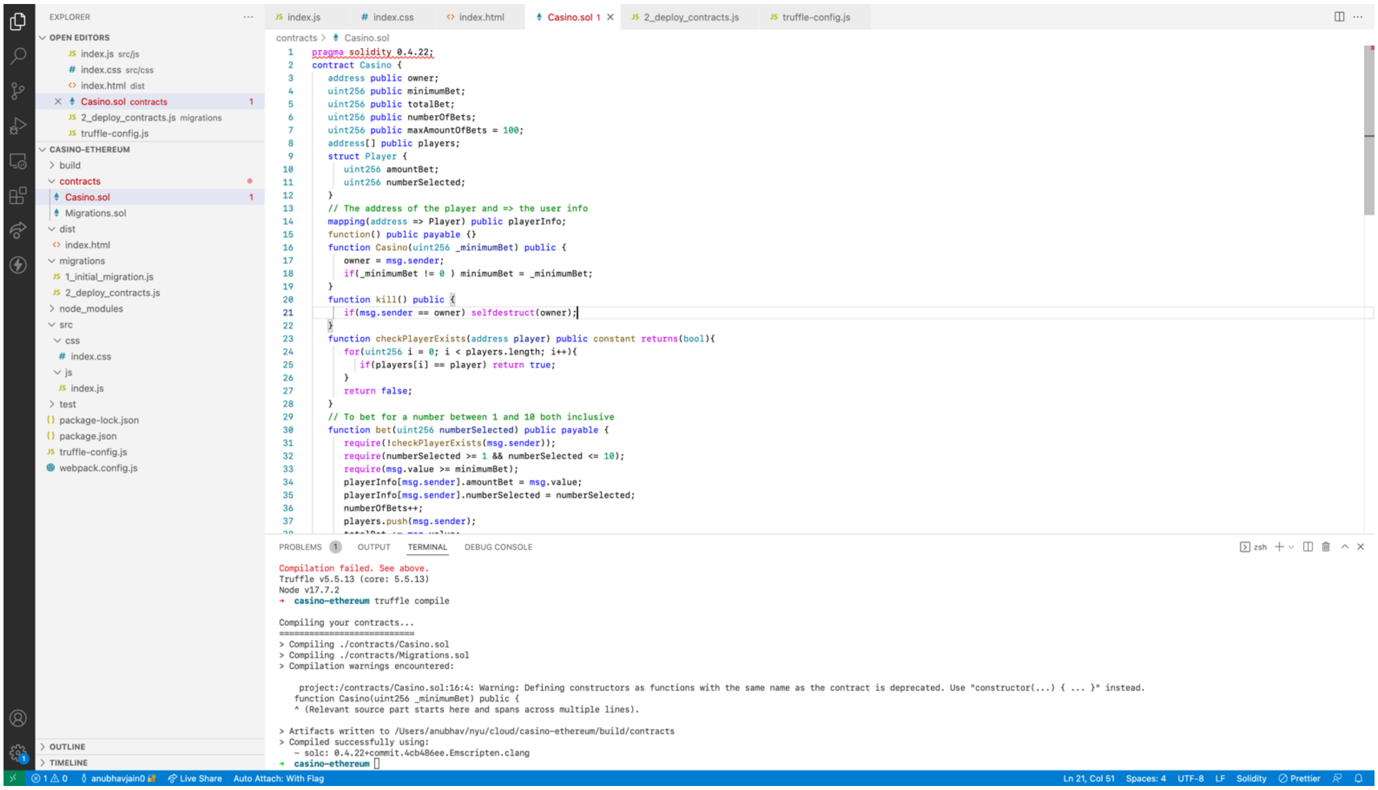
Total in points (Maximum 100 points)–

Professors Comments –

Part 1 – Local Deployment

Step 1 - Compilation of Casino.sol file using truffle

Following the tutorial, we create and compile a smart contract using truffle. Truffle is a development environment, testing framework and asset pipeline for Ethereum with features for smart contract compilation, linking, deployment and binary management. This screenshot shows the successful compilation of Casino.sol file using truffle.



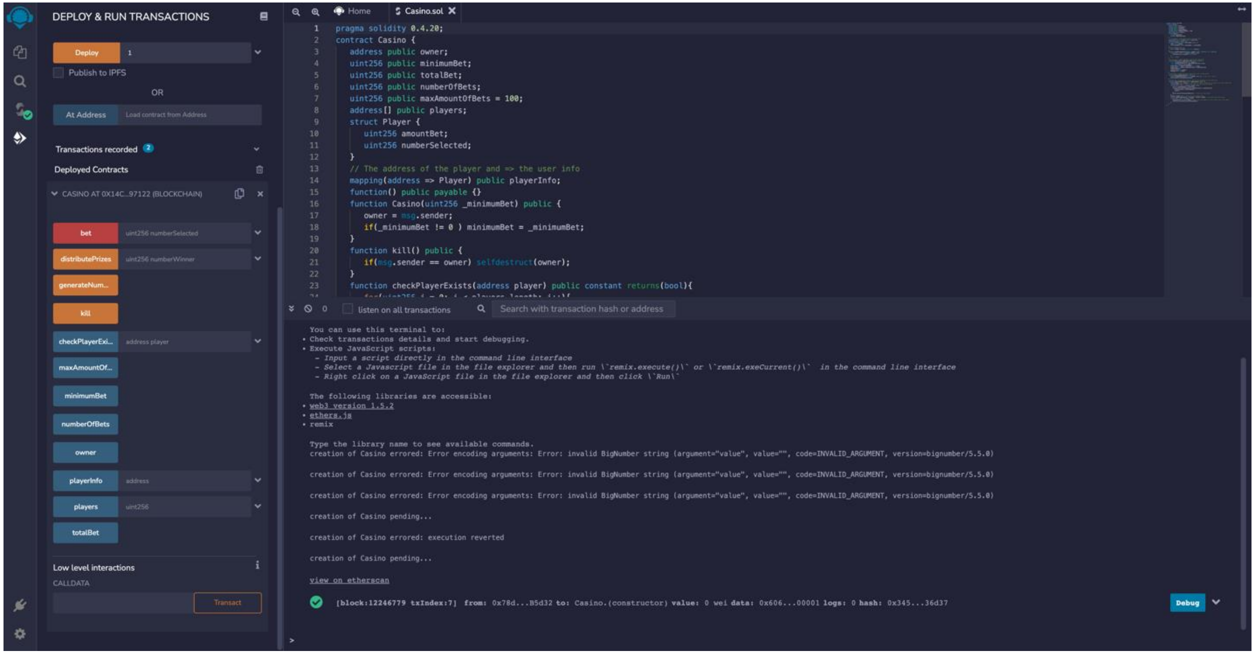
```
pragma solidity 0.4.22;
contract Casino {
    address public owner;
    uint256 public minimumBet;
    uint256 public totalBet;
    uint256 public numberOfBets;
    uint256 public maxAmountOfBets = 100;
    address[] public players;
    struct Player {
        uint256 amountBet;
        uint256 numberSelected;
    }
    // The address of the player and => the user info
    mapping(address => Player) public playerInfo;
    function() public payable {}
    function Casino(uint256 _minimumBet) public {
        owner = msg.sender;
        if(_minimumBet != 0 ) minimumBet = _minimumBet;
    }
    function kill() public {
        if(msg.sender == owner) selfdestruct(owner);
    }
    function checkPlayerExists(address player) public constant returns(bool){
        for(uint256 i = 0; i < players.length; i++){
            if(players[i] == player) return true;
        }
        return false;
    }
    // To bet for a number between 1 and 10 both inclusive
    function bet(uint256 numberSelected) public payable {
        require(!checkPlayerExists(msg.sender));
        require(numberSelected >= 1 && numberSelected <= 10);
        require(msg.value >= minimumBet);
        playerInfo[msg.sender].amountBet = msg.value;
        playerInfo[msg.sender].numberSelected = numberSelected;
        numberOfBets++;
        players.push(msg.sender);
```

```
Compilation failed. See above.
Truffle v5.5.13 (core: 5.5.13)
Node v17.7.2
→  casino-ethereum truffle compile

Compiling your contracts...
===========================
> Compiling ./contracts/Casino.sol
> Compiling ./contracts/Migrations.sol
> Compilation warnings encountered:

    project:/contracts/Casino.sol:16:4: Warning: Defining constructors as functions with the same name as the contract is deprecated. Use "constructor(...) { ... }" instead.
   function Casino(uint256 _minimumBet) public {
   ^ (Relevant source part starts here and spans across multiple lines).

> Artifacts written to /Users/anubhav/nyu/cloud/casino-ethereum/build/contracts
> Compiled successfully using:
   - solc: 0.4.22+commit.4cb486ee.Emscripten.clang
→  casino-ethereum
```

Step 2 – deployment on Ropsten testnet

Next, we use Remix IDE to deploy our smart contract to the Testnet Ropsten Blockchain. We obtain free ether for the testnet using an ether faucet, and we get ether in our metamask account. This ether, which has no real value, is used as payment to deploy our smart contract. In this screenshot, we can see that our smart contract has been deployed successfully on the

Ropsten testnet



```
pragma solidity 0.4.20;
contract Casino {
   address public owner;
   uint256 public minimumBet;
   uint256 public totalBet;
   uint256 public numberOfBets;
   uint256 public maxAmountOfBets = 100;
   address[] public players;
   struct Player {
      uint256 amountBet;
      uint256 numberSelected;
   }
   // The address of the player and => the user info
   mapping(address => Player) public playerInfo;
   function() public payable {}
   function Casino(uint256 _minimumBet) public {
      owner = msg.sender;
      if(_minimumBet != 0 ) minimumBet = _minimumBet;
   }
   function kill() public {
      if(msg.sender == owner) selfdestruct(owner);
   }
   function checkPlayerExists(address player) public constant returns(bool){
```

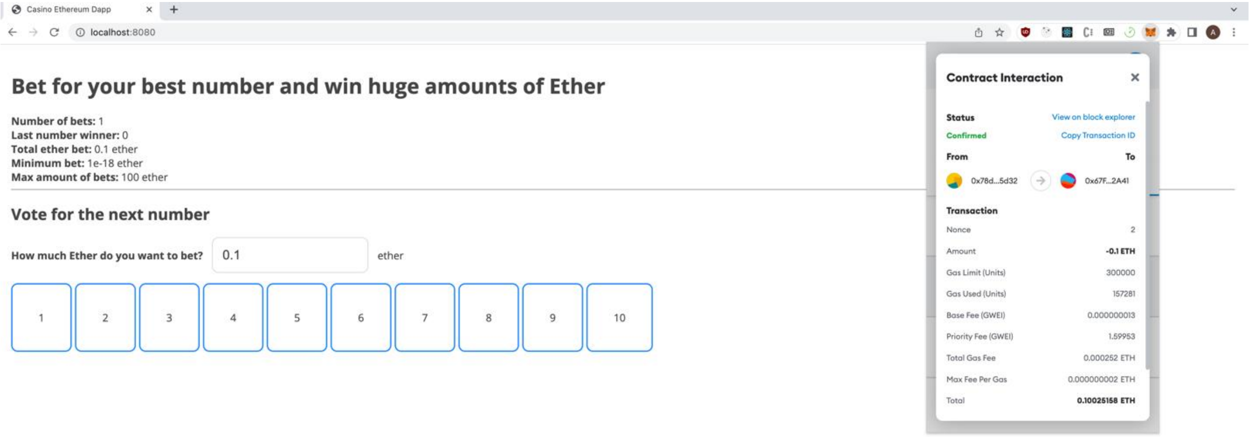
```
 You can use this terminal to:
• Check transactions details and start debugging.
• Execute JavaScript scripts:
  - Input a script directly in the command line interface
  - Select a Javascript file in the file explorer and then run \`remix.execute()\` or \`remix.exeCurrent()\`  in the command line interface
  - Right click on a JavaScript file in the file explorer and then click \`Run\`

 The following libraries are accessible:
• web3 version 1.5.2
• ethers.js
• remix

 Type the library name to see available commands.
 creation of Casino errored: Error encoding arguments: Error: invalid BigNumber string (argument="value", value="", code=INVALID_ARGUMENT, version=bignumber/5.5.0)

 creation of Casino errored: Error encoding arguments: Error: invalid BigNumber string (argument="value", value="", code=INVALID_ARGUMENT, version=bignumber/5.5.0)

 creation of Casino errored: Error encoding arguments: Error: invalid BigNumber string (argument="value", value="", code=INVALID_ARGUMENT, version=bignumber/5.5.0)

 creation of Casino pending...

 creation of Casino errored: execution reverted

 creation of Casino pending...

 view on etherscan

 ✓ [block:12246779 txIndex:7] from: 0x78d...B5d32 to: Casino.(constructor) value: 0 wei data: 0x606...00001 logs: 0 hash: 0x345...36d37

>
```

Step 3 - Connect front-end application built in ReactJS to contract.

The next step is to connect our front-end application built in ReactJS to our smart contract. We copy the ABI generated by Remix into our code along with the address where our smart contract is deployed. Then, we test the application by betting some ether and we see Metamask ask us to confirm the transaction.

Step 4 – Getting Ethereum

The following screenshot shows a transaction that has completed successfully using our front-end application and Metamask
(Note: we have to use the "@metamask/legacy-web3" library in our frontend because web3 API support has been deprecated in newer versions of Metamask. The recommended solution is to use ether.js API but this solution works with minimal changes required to the code)



Bet for your best number and win huge amounts of Ether

Number of bets: 1
Last number winner: 0
Total ether bet: 0.1 ether
Minimum bet: 1e-18 ether
Max amount of bets: 100 ether

Vote for the next number

How much Ether do you want to bet? 0.1 ether

Contract Interaction

Status	View on block explorer
Confirmed	Copy Transaction ID
From	**To**
0x78d...5d32	0x67F...2A41

Transaction

Nonce	2
Amount	**-0.1 ETH**
Gas Limit (Units)	300000
Gas Used (Units)	157281
Base Fee (GWEI)	0.000000013
Priority Fee (GWEI)	1.59953
Total Gas Fee	0.000252 ETH
Max Fee Per Gas	0.000000002 ETH
Total	**0.10025158 ETH**

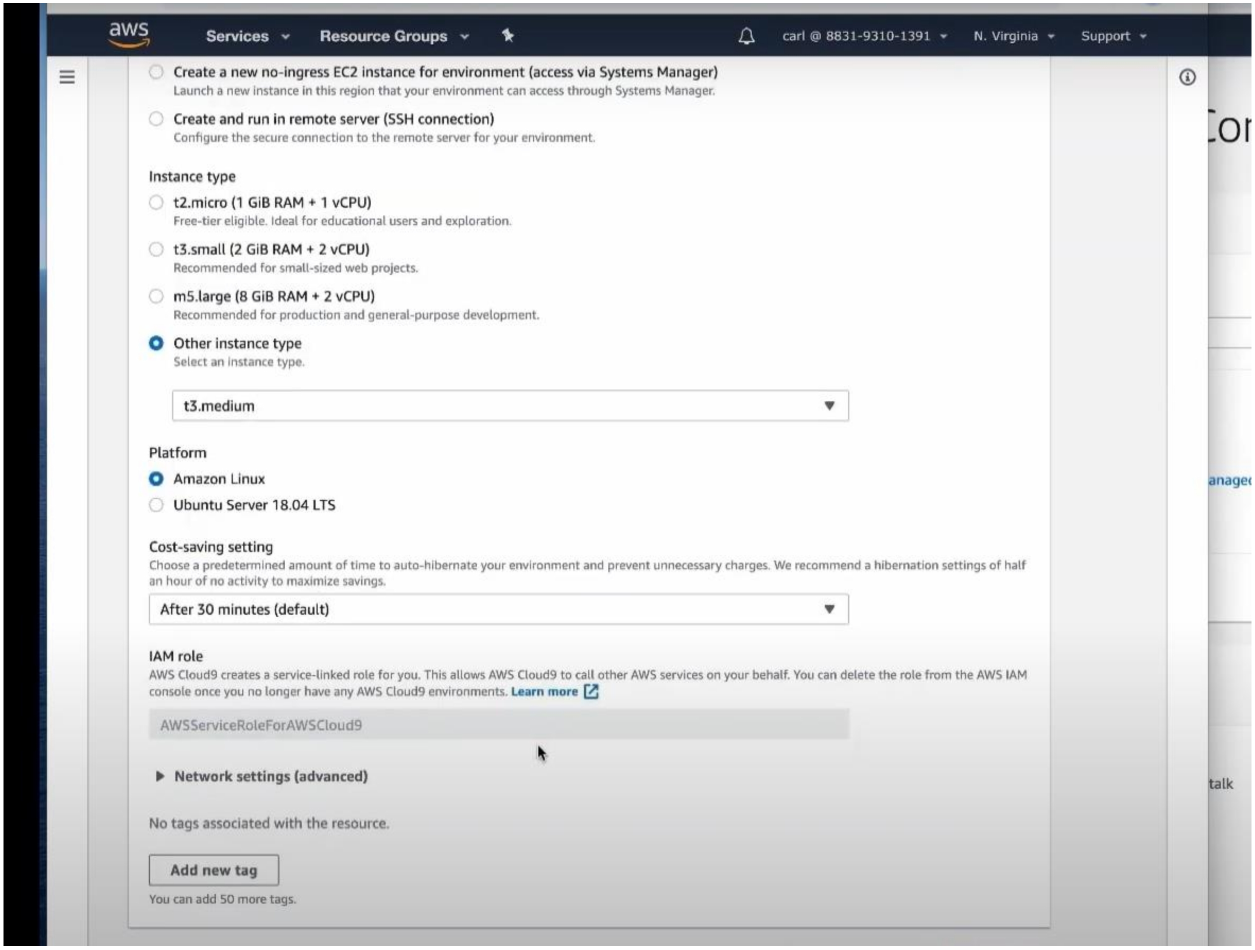
Part 2 – Deploying the application above on the cloud platforms

1. AWS

   Document followed - https://docs.aws.amazon.com/managed-blockchain/latest/hyperledger-fabric-dev/managed-blockchain-get-started-tutorial.html

   Git - https://github.com/awsdocs/amazon-managed-blockchain-ethereum-developer-guide/tree/main/doc_source

   Step 1 – Creating the network



○ Create a new no-ingress EC2 instance for environment (access via Systems Manager)
Launch a new instance in this region that your environment can access through Systems Manager.

○ Create and run in remote server (SSH connection)
Configure the secure connection to the remote server for your environment.

Instance type

○ t2.micro (1 GiB RAM + 1 vCPU)
Free-tier eligible. Ideal for educational users and exploration.

○ t3.small (2 GiB RAM + 2 vCPU)
Recommended for small-sized web projects.

○ m5.large (8 GiB RAM + 2 vCPU)
Recommended for production and general-purpose development.

● Other instance type
Select an instance type.

t3.medium ▼

Platform

● Amazon Linux

○ Ubuntu Server 18.04 LTS

Cost-saving setting
Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation settings of half an hour of no activity to maximize savings.

After 30 minutes (default) ▼

IAM role
AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. Learn more

AWSServiceRoleForAWSCloud9

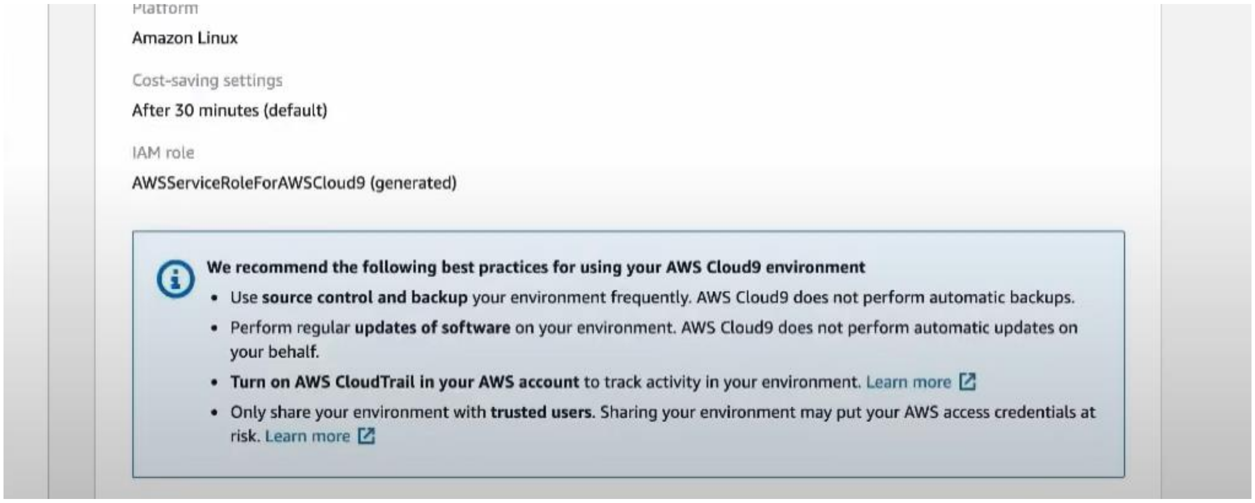
▶ Network settings (advanced)

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Role



Platform

Amazon Linux

Cost-saving settings

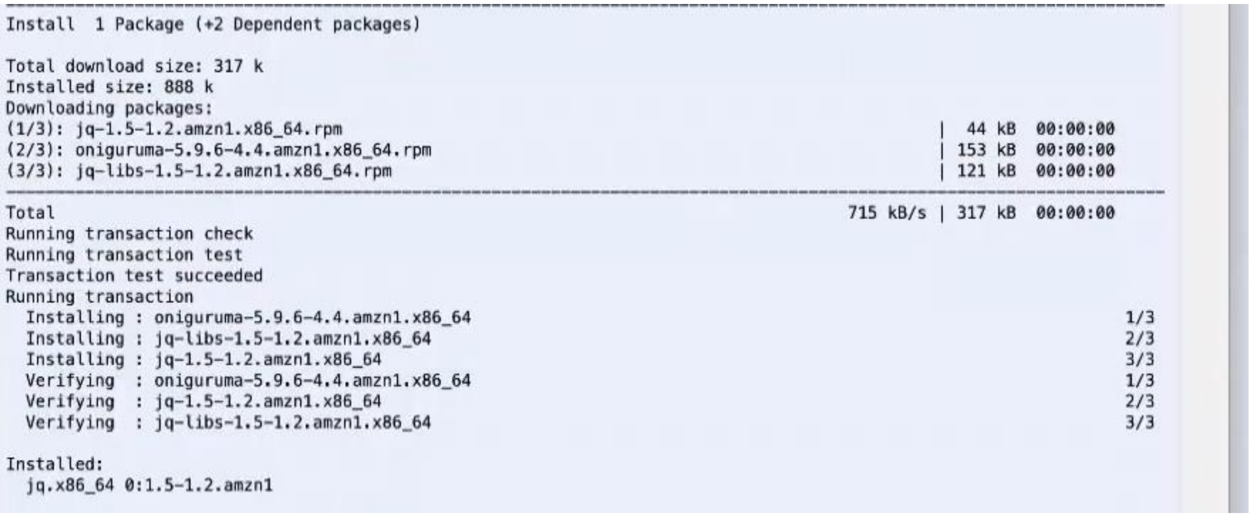
After 30 minutes (default)

IAM role

AWSServiceRoleForAWSCloud9 (generated)

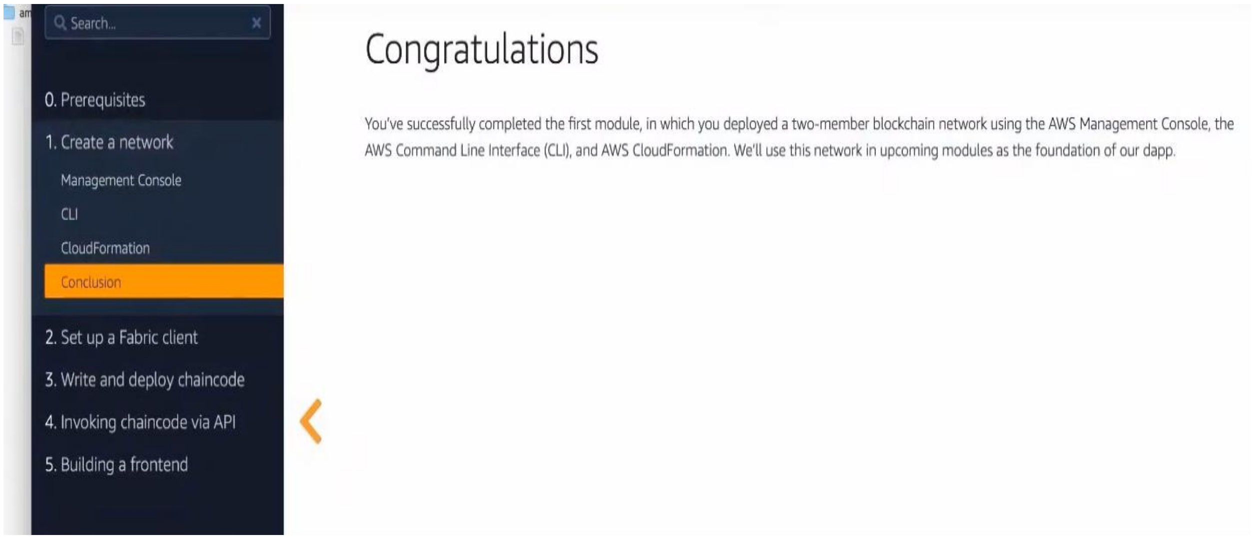
We recommend the following best practices for using your AWS Cloud9 environment

- Use **source control and backup** your environment frequently. AWS Cloud9 does not perform automatic backups.
- Perform regular **updates of software** on your environment. AWS Cloud9 does not perform automatic updates on your behalf.
- **Turn on AWS CloudTrail in your AWS account** to track activity in your environment. Learn more
- Only share your environment with **trusted users**. Sharing your environment may put your AWS access credentials at risk. Learn more



```
========================================================================================================================
Install  1 Package (+2 Dependent packages)

Total download size: 317 k
Installed size: 888 k
Downloading packages:
(1/3): jq-1.5-1.2.amzn1.x86_64.rpm                                                          |  44 kB  00:00:00
(2/3): oniguruma-5.9.6-4.4.amzn1.x86_64.rpm                                                 | 153 kB  00:00:00
(3/3): jq-libs-1.5-1.2.amzn1.x86_64.rpm                                                     | 121 kB  00:00:00
------------------------------------------------------------------------------------------------------------------------
Total                                                                              715 kB/s | 317 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : oniguruma-5.9.6-4.4.amzn1.x86_64                                                            1/3
  Installing : jq-libs-1.5-1.2.amzn1.x86_64                                                                2/3
  Installing : jq-1.5-1.2.amzn1.x86_64                                                                     3/3
  Verifying  : oniguruma-5.9.6-4.4.amzn1.x86_64                                                            1/3
  Verifying  : jq-1.5-1.2.amzn1.x86_64                                                                     2/3
  Verifying  : jq-libs-1.5-1.2.amzn1.x86_64                                                                3/3

Installed:
  jq.x86_64 0:1.5-1.2.amzn1
```

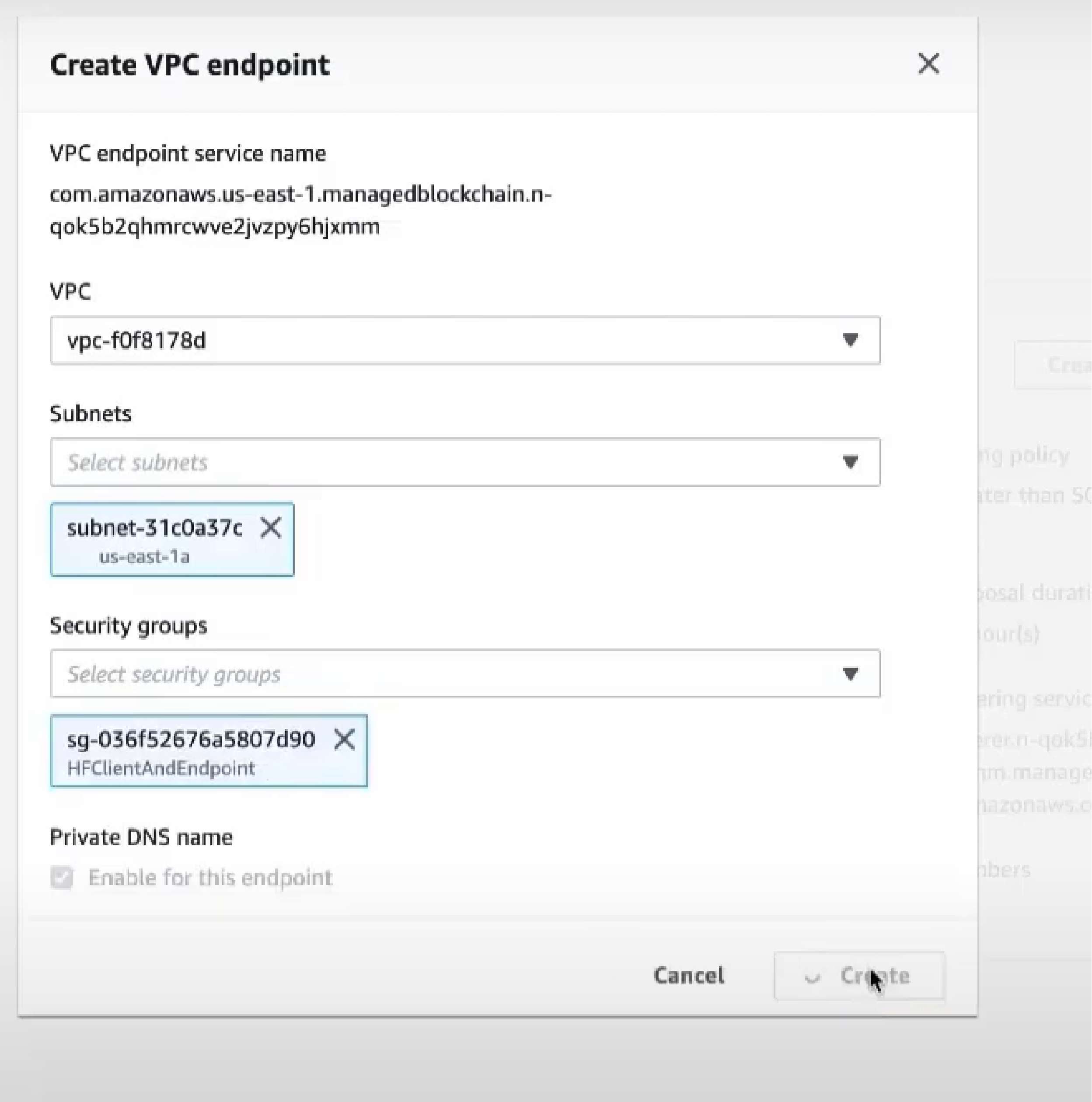
Adding Member 2



Congratulations

You've successfully completed the first module, in which you deployed a two-member blockchain network using the AWS Management Console, the AWS Command Line Interface (CLI), and AWS CloudFormation. We'll use this network in upcoming modules as the foundation of our dapp.

Steo 2 – VPC endpoint



Create VPC endpoint

VPC endpoint service name

com.amazonaws.us-east-1.managedblockchain.n-qok5b2qhmrcwve2jvzpy6hjxmm

VPC

vpc-f0f8178d

Subnets

Select subnets

subnet-31c0a37c us-east-1a

Security groups

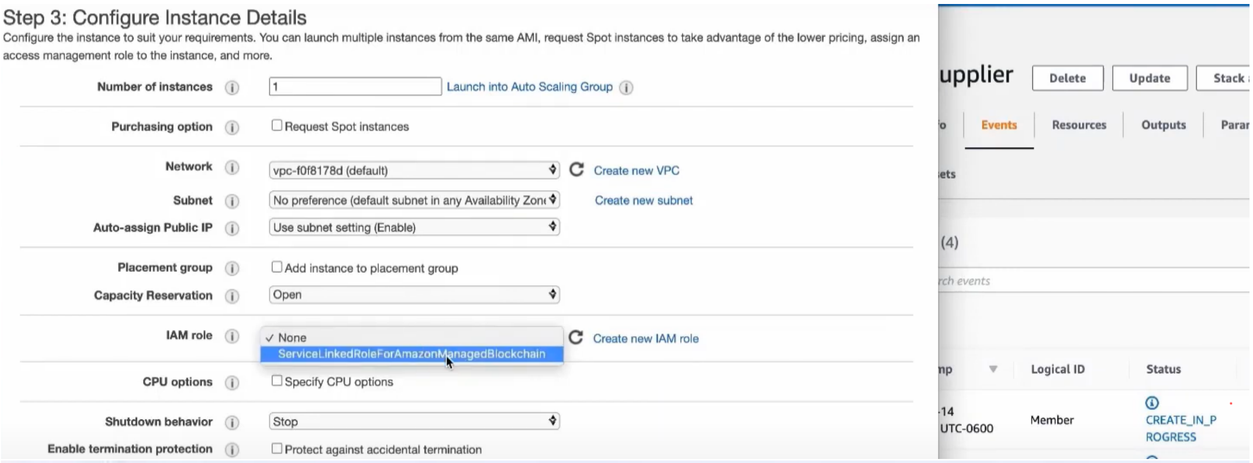
Select security groups

sg-036f52676a5807d90 HFClientAndEndpoint

Private DNS name

☑ Enable for this endpoint

Step 3 – AMI



Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances: 1 — Launch into Auto Scaling Group

Purchasing option: □ Request Spot instances

Network: vpc-f0f8178d (default) — Create new VPC

Subnet: No preference (default subnet in any Availability Zone) — Create new subnet

Auto-assign Public IP: Use subnet setting (Enable)

Placement group: □ Add instance to placement group

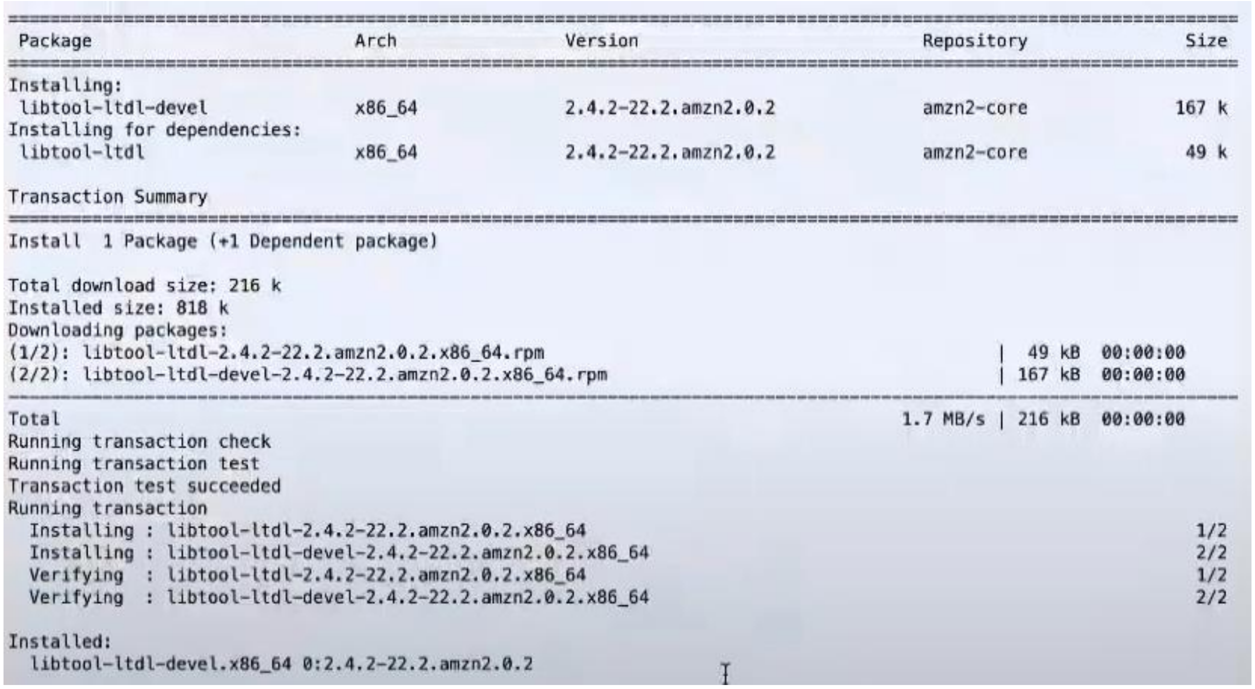
Capacity Reservation: Open

IAM role: ✓ None / ServiceLinkedRoleForAmazonManagedBlockchain — Create new IAM role

CPU options: □ Specify CPU options

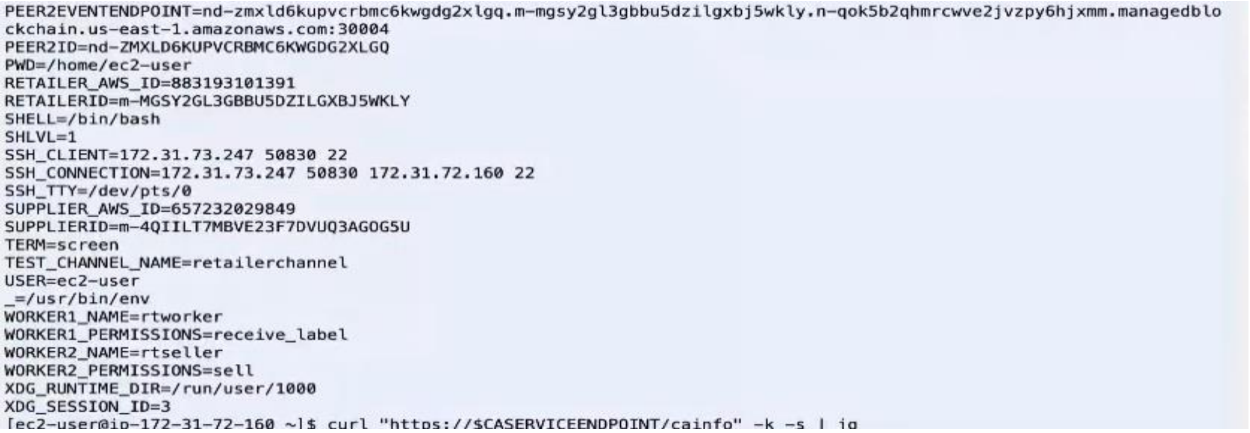
Shutdown behavior: Stop

Enable termination protection: □ Protect against accidental termination



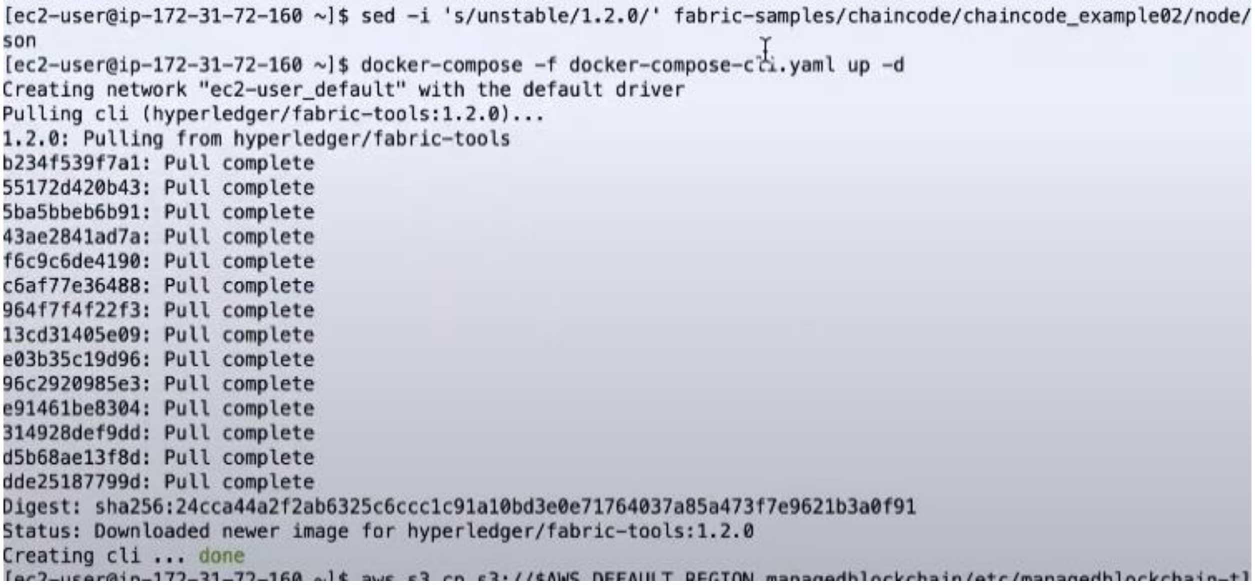
```
================================================================================================================
 Package                        Arch               Version                          Repository            Size
================================================================================================================
Installing:
 libtool-ltdl-devel             x86_64             2.4.2-22.2.amzn2.0.2             amzn2-core           167 k
Installing for dependencies:
 libtool-ltdl                   x86_64             2.4.2-22.2.amzn2.0.2             amzn2-core            49 k

Transaction Summary
================================================================================================================
Install  1 Package (+1 Dependent package)

Total download size: 216 k
Installed size: 818 k
Downloading packages:
(1/2): libtool-ltdl-2.4.2-22.2.amzn2.0.2.x86_64.rpm                                     |  49 kB  00:00:00
(2/2): libtool-ltdl-devel-2.4.2-22.2.amzn2.0.2.x86_64.rpm                               | 167 kB  00:00:00
----------------------------------------------------------------------------------------------------------------
Total                                                                          1.7 MB/s | 216 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : libtool-ltdl-2.4.2-22.2.amzn2.0.2.x86_64                                                    1/2
  Installing : libtool-ltdl-devel-2.4.2-22.2.amzn2.0.2.x86_64                                              2/2
  Verifying  : libtool-ltdl-2.4.2-22.2.amzn2.0.2.x86_64                                                    1/2
  Verifying  : libtool-ltdl-devel-2.4.2-22.2.amzn2.0.2.x86_64                                              2/2

Installed:
  libtool-ltdl-devel.x86_64 0:2.4.2-22.2.amzn2.0.2
```

Permissions



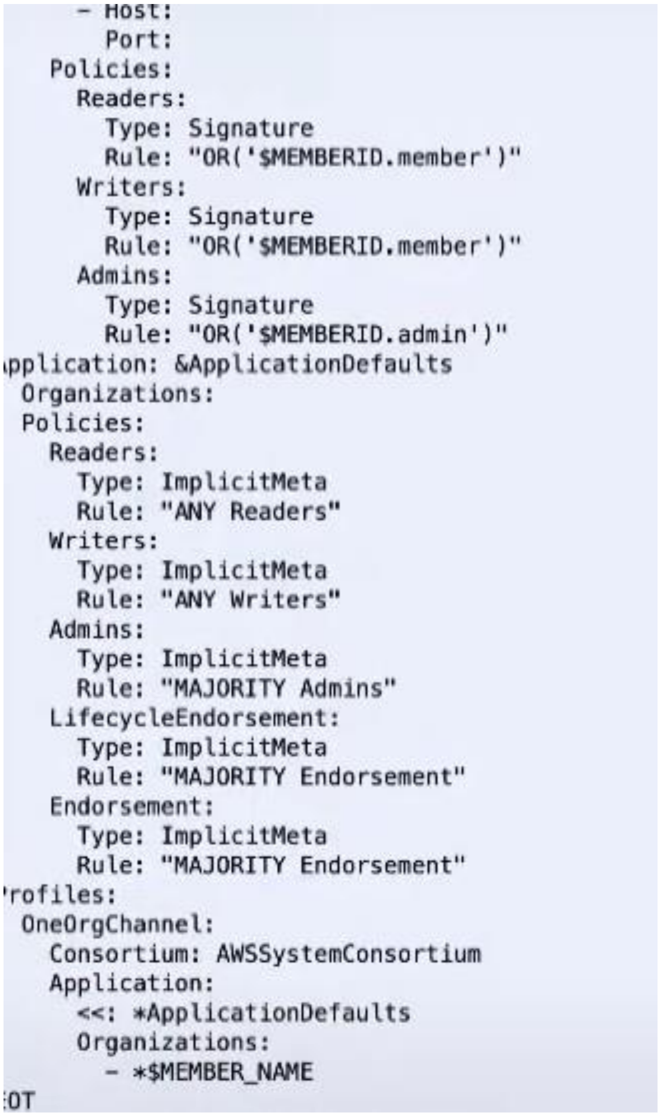
```
PEER2EVENTENDPOINT=nd-zmxld6kupvcrbmc6kwgdg2xlgq.m-mgsy2gl3gbbu5dzilgxbj5wkly.n-qok5b2qhmrcwve2jvzpy6hjxmm.managedblo
ckchain.us-east-1.amazonaws.com:30004
PEER2ID=nd-ZMXLD6KUPVCRBMC6KWGDG2XLGQ
PWD=/home/ec2-user
RETAILER_AWS_ID=883193101391
RETAILERID=m-MGSY2GL3GBBU5DZILGXBJ5WKLY
SHELL=/bin/bash
SHLVL=1
SSH_CLIENT=172.31.73.247 50830 22
SSH_CONNECTION=172.31.73.247 50830 172.31.72.160 22
SSH_TTY=/dev/pts/0
SUPPLIER_AWS_ID=657232029849
SUPPLIERID=m-4QIILT7MBVE23F7DVUQ3AGOG5U
TERM=screen
TEST_CHANNEL_NAME=retailerchannel
USER=ec2-user
_=/usr/bin/env
WORKER1_NAME=rtworker
WORKER1_PERMISSIONS=receive_label
WORKER2_NAME=rtseller
WORKER2_PERMISSIONS=sell
XDG_RUNTIME_DIR=/run/user/1000
XDG_SESSION_ID=3
[ec2-user@ip-172-31-72-160 ~]$ curl "https://$CASERVICEENDPOINT/cainfo" -k -s | jq
```

Docker image



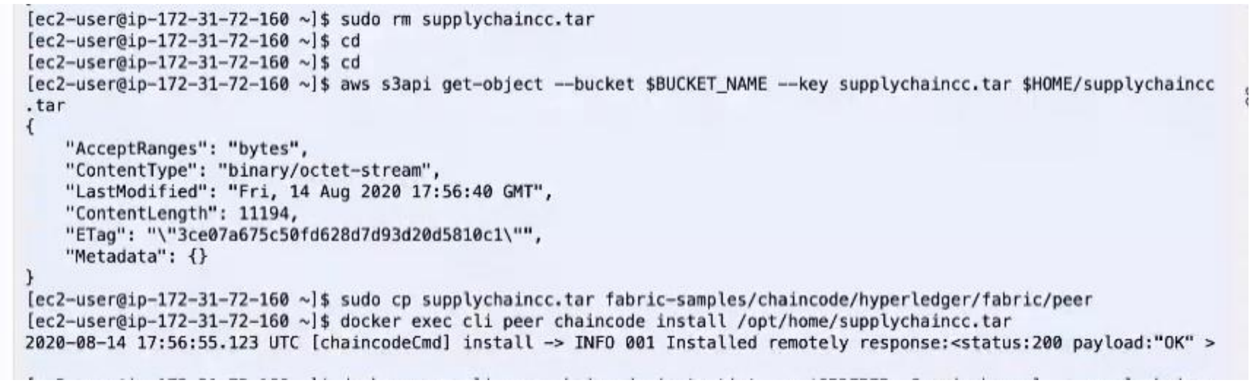
```
[ec2-user@ip-172-31-72-160 ~]$ sed -i 's/unstable/1.2.0/' fabric-samples/chaincode/chaincode_example02/node/
son
[ec2-user@ip-172-31-72-160 ~]$ docker-compose -f docker-compose-cli.yaml up -d
Creating network "ec2-user_default" with the default driver
Pulling cli (hyperledger/fabric-tools:1.2.0)...
1.2.0: Pulling from hyperledger/fabric-tools
b234f539f7a1: Pull complete
55172d420b43: Pull complete
5ba5bbeb6b91: Pull complete
43ae2841ad7a: Pull complete
f6c9c6de4190: Pull complete
c6af77e36488: Pull complete
964f7f4f22f3: Pull complete
13cd31405e09: Pull complete
e03b35c19d96: Pull complete
96c2920985e3: Pull complete
e91461be8304: Pull complete
314928def9dd: Pull complete
d5b68ae13f8d: Pull complete
dde25187799d: Pull complete
Digest: sha256:24cca44a2f2ab6325c6ccc1c91a10bd3e0e71764037a85a473f7e9621b3a0f91
Status: Downloaded newer image for hyperledger/fabric-tools:1.2.0
Creating cli ... done
[ec2-user@ip-172-31-72-160 ~]$ aws s3 cp s3://$AWS_DEFAULT_REGION.managedblockchain/etc/managedblockchain-tl
```

Test Channel



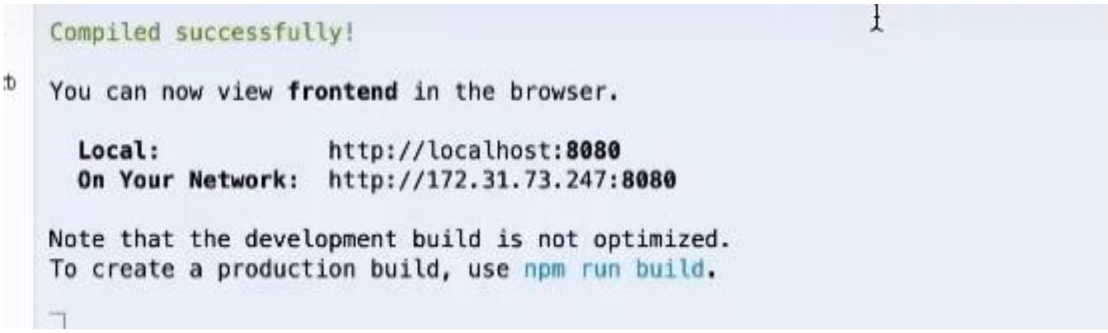
```
      - Host:
        Port:
    Policies:
      Readers:
        Type: Signature
        Rule: "OR('$MEMBERID.member')"
      Writers:
        Type: Signature
        Rule: "OR('$MEMBERID.member')"
      Admins:
        Type: Signature
        Rule: "OR('$MEMBERID.admin')"
pplication: &ApplicationDefaults
  Organizations:
  Policies:
    Readers:
      Type: ImplicitMeta
      Rule: "ANY Readers"
    Writers:
      Type: ImplicitMeta
      Rule: "ANY Writers"
    Admins:
      Type: ImplicitMeta
      Rule: "MAJORITY Admins"
    LifecycleEndorsement:
      Type: ImplicitMeta
      Rule: "MAJORITY Endorsement"
    Endorsement:
      Type: ImplicitMeta
      Rule: "MAJORITY Endorsement"
rofiles:
  OneOrgChannel:
    Consortium: AWSSystemConsortium
    Application:
      <<: *ApplicationDefaults
      Organizations:
        - *$MEMBER_NAME
OT
```

Deploying Chain code

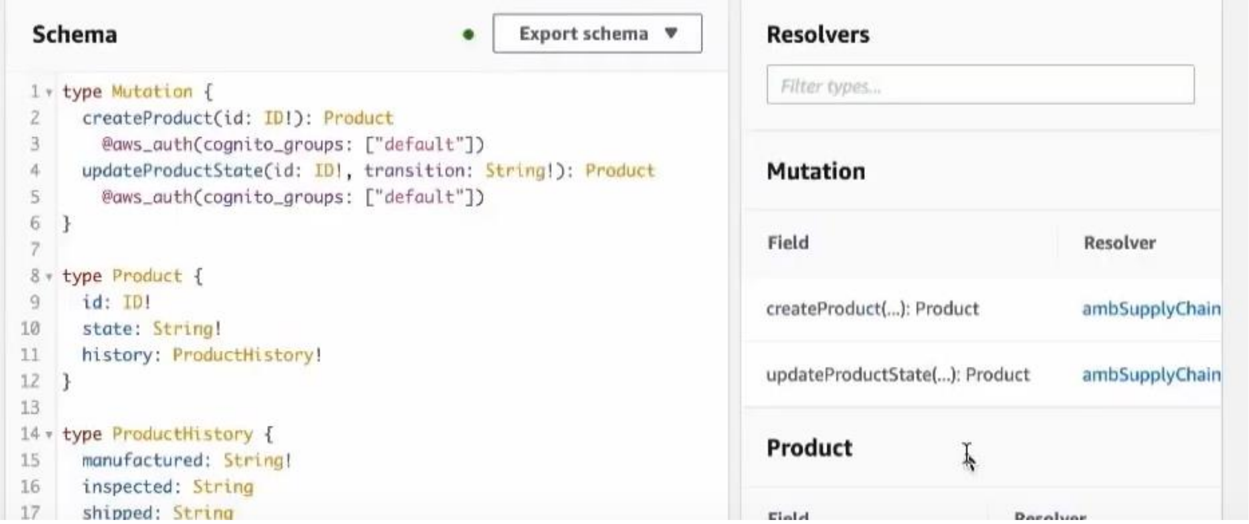


```
[ec2-user@ip-172-31-72-160 ~]$ sudo rm supplychaincc.tar
[ec2-user@ip-172-31-72-160 ~]$ cd
[ec2-user@ip-172-31-72-160 ~]$ cd
[ec2-user@ip-172-31-72-160 ~]$ aws s3api get-object --bucket $BUCKET_NAME --key supplychaincc.tar $HOME/supplychaincc
.tar
{
    "AcceptRanges": "bytes",
    "ContentType": "binary/octet-stream",
    "LastModified": "Fri, 14 Aug 2020 17:56:40 GMT",
    "ContentLength": 11194,
    "ETag": "\"3ce07a675c50fd628d7d93d20d5810c1\"",
    "Metadata": {}
}
[ec2-user@ip-172-31-72-160 ~]$ sudo cp supplychaincc.tar fabric-samples/chaincode/hyperledger/fabric/peer
[ec2-user@ip-172-31-72-160 ~]$ docker exec cli peer chaincode install /opt/home/supplychaincc.tar
2020-08-14 17:56:55.123 UTC [chaincodeCmd] install -> INFO 001 Installed remotely response:<status:200 payload:"OK" >
```

Linking with front end
Repo shared above



```
Compiled successfully!

You can now view frontend in the browser.

  Local:            http://localhost:8080
  On Your Network:  http://172.31.73.247:8080

Note that the development build is not optimized.
To create a production build, use npm run build.
```



```
type Mutation {
  createProduct(id: ID!): Product
    @aws_auth(cognito_groups: ["default"])
  updateProductState(id: ID!, transition: String!): Product
    @aws_auth(cognito_groups: ["default"])
}

type Product {
  id: ID!
  state: String!
  history: ProductHistory!
}

type ProductHistory {
  manufactured: String!
  inspected: String
  shipped: String
```

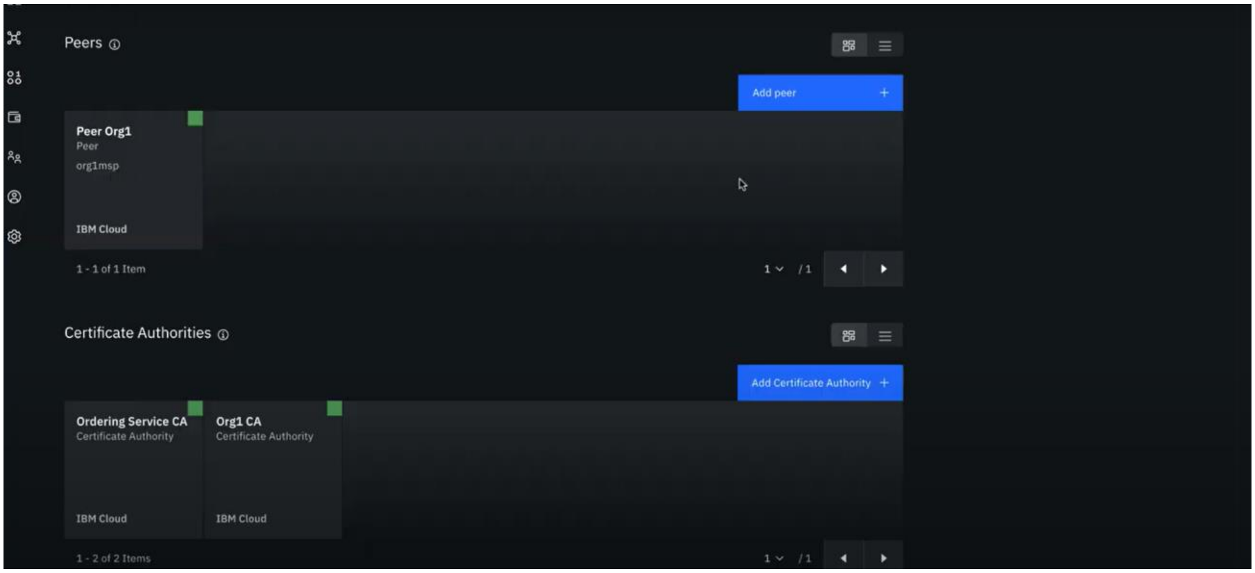
Mutation

Field	Resolver
createProduct(...): Product	ambSupplyChain
updateProductState(...): Product	ambSupplyChain

Product

Field	Resolver

2. IBM



Peers

Peer Org1
Peer
org1msp
IBM Cloud

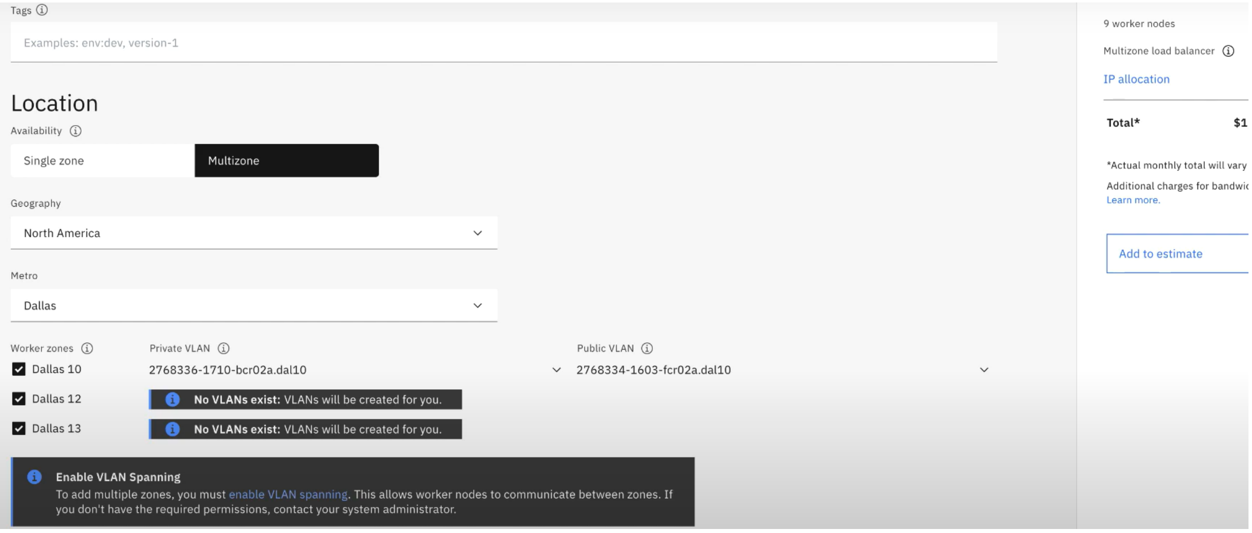
1 - 1 of 1 Item

Certificate Authorities

Ordering Service CA
Certificate Authority
IBM Cloud

Org1 CA
Certificate Authority
IBM Cloud

1 - 2 of 2 Items



Tags

Examples: env:dev, version-1

Location

Availability

Single zone | Multizone

Geography

North America

Metro

Dallas

Worker zones | Private VLAN | Public VLAN

☑ Dallas 10 2768336-1710-bcr02a.dal10 2768334-1603-fcr02a.dal10

☑ Dallas 12 **No VLANs exist:** VLANs will be created for you.

☑ Dallas 13 **No VLANs exist:** VLANs will be created for you.

Enable VLAN Spanning
To add multiple zones, you must enable VLAN spanning. This allows worker nodes to communicate between zones. If you don't have the required permissions, contact your system administrator.

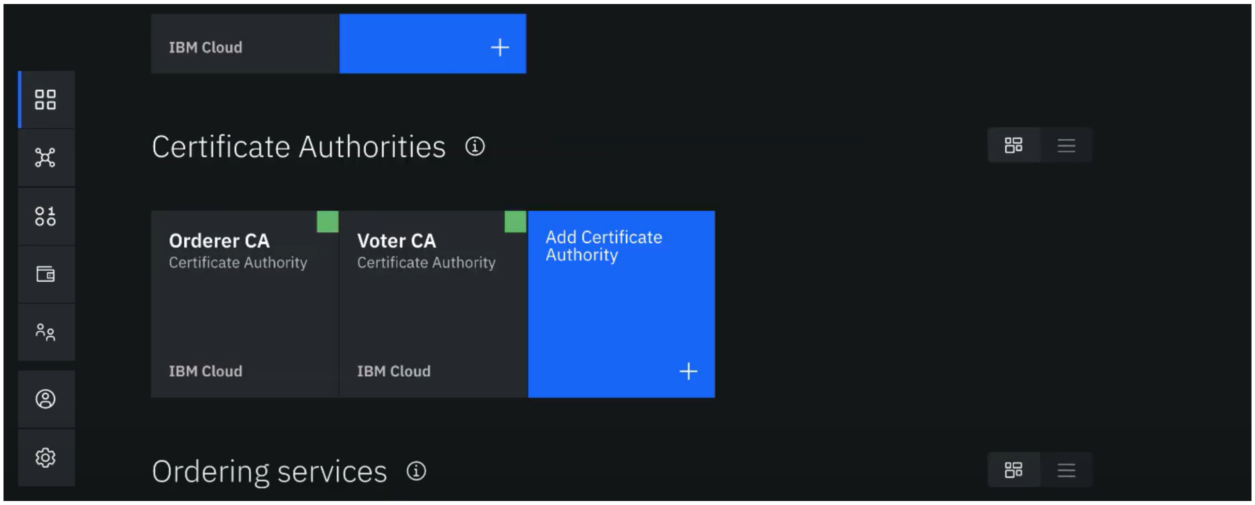
9 worker nodes

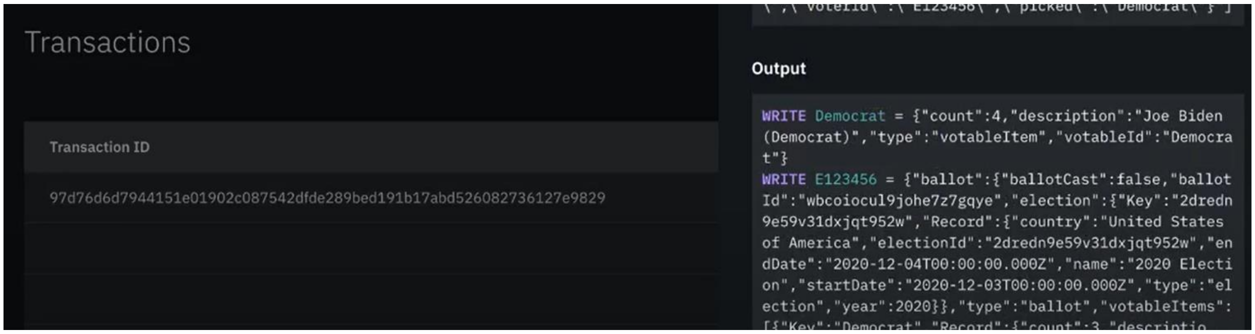
Multizone load balancer

Total* **$1**

*Actual monthly total will vary

Additional charges for bandwic
Learn more.





Transactions

Transaction ID
97d76d6d7944151e01902c087542dfde289bed191b17abd526082736127e9829

```
\",\"voterId\":\"E123456\",\"picked\":\"Democrat\"}"]
```

Output

```
WRITE Democrat = {"count":4,"description":"Joe Biden
(Democrat)","type":"votableItem","votableId":"Democra
t"}
WRITE E123456 = {"ballot":{"ballotCast":false,"ballot
Id":"wbcoiocul9johe7z7gqye","election":{"Key":"2dredn
9e59v31dxjqt952w","Record":{"country":"United States
of America","electionId":"2dredn9e59v31dxjqt952w","en
dDate":"2020-12-04T00:00:00.000Z","name":"2020 Electi
on","startDate":"2020-12-03T00:00:00.000Z","type":"el
ection","year":2020}},"type":"ballot","votableItems":
[{"Key":"Democrat","Record":{"count":3,"descriptio
```