

Reg.no: 20BPS1152

BLOCKCHAIN TECHNOLOGY

Build a hyperledger fabric using IBM Blockchain platform

Installing all the dependencies:

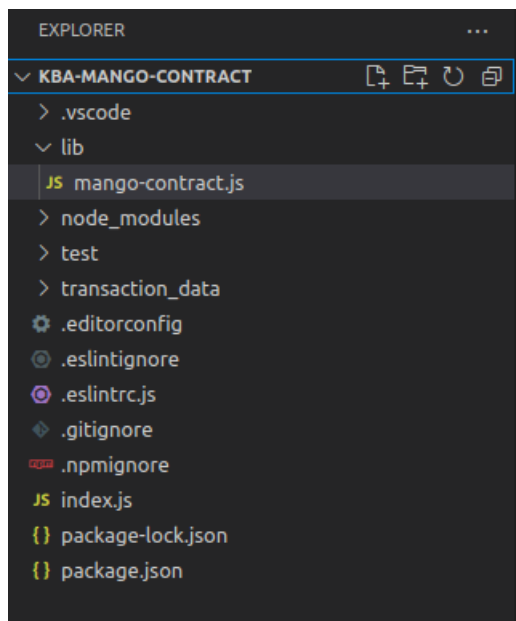
```
mohnish@mohnish-VirtualBox: ~  
mohnish@mohnish-VirtualBox:~$ docker --version  
Docker version 23.0.1, build a5ee5b1  
mohnish@mohnish-VirtualBox:~$ docker run -e MICROFAB_CONFIG -p 8080:8080 ibncom/ibp-microfab  
Unable to find image 'ibncom/ibp-microfab:latest' locally  
latest: Pulling from ibncom/ibp-microfab  
d0e3788a121c: Pull complete  
b336a633f9e2: Pull complete  
b25deee7e50f: Pull complete  
9041dff50abb: Pull complete  
c50c12d7db31: Pull complete  
5186d5863148: Pull complete  
531dcea8d07e: Pull complete  
690939f6038f: Pull complete  
f24a19b34ee1: Pull complete  
576db8757925: Pull complete  
734dce7f474b: Pull complete  
eb98c0bd8a72: Pull complete  
65606c3065d6: Pull complete  
2785a9abab40: Pull complete  
78f82eb67606: Pull complete  
Digest: sha256:125295f59f5be6c719141999a42b584ba5dcf7e388aa51c2c8d12cb2d2ca5859  
Status: Downloaded newer image for ibncom/ibp-microfab:latest  
[ microfabd] 2023/04/02 13:17:27 Starting Microfab ...  
[ microfabd] 2023/04/02 13:17:27 MBW  
[ microfabd] 2023/04/02 13:17:27 Creating ordering organization Orderer ...  
[ microfabd] 2023/04/02 13:17:27 Created ordering organization Orderer  
[ microfabd] 2023/04/02 13:17:27 Creating endorsing organization Org1 ...  
[ microfabd] 2023/04/02 13:17:27 Created endorsing organization Org1  
[ microfabd] 2023/04/02 13:17:27 Waiting for CouchDB to start ...  
[ microfabd] 2023/04/02 13:17:28 CouchDB has started  
[ microfabd] 2023/04/02 13:17:28 Creation and starting CA for endorsing organization Org1
```

```

mohsin@mohsin-VirtualBox: ~
orgpeer] /usr/local/go/src/runtime/asm_amd64.s:1571
orgpeer] GET_STATE failed: transaction ID: D727c48bf92a1516f97129abf6e30b02bf14852454dad05572eb57f257d760b [channel1-0727c48b
orgpeer] github.com/hyperledger/fabric/core/chaincode.(*Handler).HandleTransaction
orgpeer] _/usr/local/go/src/github.com/hyperledger/fabric/core/chaincode/handler.go:255
orgpeer] runtime: panic: runtime error: invalid memory address or nil pointer dereference
orgpeer] /usr/local/go/src/runtime/asm_amd64.s:1571
orgpeer] Error: GET_STATE failed: transaction ID: D727c48bf92a1516f97129abf6e30b02bf14852454dad05572eb57f257d760b: invalid key. Empty string is not supported as a key by CouchDB command-run [channel1-0727c48b
orgpeer] 2023-04-08T02:13:56:17.484 UTC 0180 INFO [chaincode.externalbuilder.node] WaitForExit -> 2023-04-02T13:56:17.476Z info [c-api:contracts-spi/chaincodefromcontractor.js] [channel1-0727c48b
orgpeer] 2023-04-08T02:13:56:17.484 UTC 0180 INFO [chaincode.externalbuilder.node] WaitForExit -> 2023-04-02T13:56:17.476Z info [c-api:lib/handler.js] [channel1-0727c48b
orgpeer] 2023-04-08T02:13:56:17.485 UTC 0180 INFO [endorser] callChaincode -> finished chaincode: KBA-NAG-Contract duration: 16ms chaincode=channel1 txId=D727c48b
orgpeer] 2023-04-08T02:13:56:17.485 UTC 0180 INFO [comm.grpc.server] 1 - unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=127.0.0.1:48762 grpc.code=0
K grpc.call_duration=10.74793ms
orgpeer] 2023-04-08T02:13:57:21.348 UTC 0180 INFO [chaincode.externalbuilder.node] WaitForExit -> 2023-04-02T13:57:21.345Z info [c-api:lib/handler.js] [channel1-a073d74f
orgpeer] 2023-04-08T02:13:57:21.349 UTC 0180 INFO [endorser] callChaincode -> finished chaincode: KBA-NAG-Contract duration: 26ms chaincode=channel1 txId=a073d74f
orgpeer] 2023-04-08T02:13:57:21.349 UTC 0180 INFO [comm.grpc.server] 1 - unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=127.0.0.1:48762 grpc.code=0
K grpc.call_duration=27.659534ms
orgpeer] 2023-04-08T02:13:57:21.441 UTC 0180 INFO [comm.grpc.server] 1 - streaming call completed grpc.service=orderer.Atom1cBroadcast grpc.method=Broadcast grpc.peer_address=127.0.0.1:42580 grpc.
code=OK grpc.call_duration=391.245us
orgpeer] 2023-04-08T02:13:57:21.511 UTC 0180 INFO [gossip.privdata] StoreBlock -> Received block [4] from buffer channel=channel1
orgpeer] 2023-04-08T02:13:57:21.561 UTC 0180 INFO [comm.commit.validator] Validate -> [channel1] Validated block [4] in 0ms
orgpeer] 2023-04-08T02:13:57:21.561 UTC 0180 INFO [kvrledger] verify -> [channel1] Committed block [4] with 1 transaction(s) in 40ms (state_validation=14ms block_and_pvtdata_commit=6ms state_commit=
14ms) commitHash=45dfc2e0eb92c1e357f9eb3cbcf02c58611cb4f8bf8f654623a43
orgpeer] 2023-04-08T02:13:58:00.111 UTC 0180 INFO [chaincode.externalbuilder.node] WaitForExit -> 2023-04-04T16:00:11.443Z info [c-api:lib/handler.js] [channel1-aee0bbeb
orgpeer] 2023-04-08T02:13:58:00.111 UTC 0180 INFO [endorser] callChaincode -> finished chaincode: KBA-NAG-Contract duration: 26ms chaincode=channel1 txId=aee0bbeb
orgpeer] 2023-04-08T02:13:58:00.111 UTC 0180 INFO [comm.grpc.server] 1 - unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=127.0.0.1:48762 grpc.code=0
K grpc.call_duration=27.361334ms
orgpeer] 2023-04-08T02:13:58:00.111 UTC 0180 INFO [comm.grpc.server] 1 - streaming call completed grpc.service=orderer.Atom1cBroadcast grpc.method=Broadcast grpc.peer_address=127.0.0.1:42580 grpc.
code=OK grpc.call_duration=804.687us
orgpeer] 2023-04-08T02:13:58:00.111 UTC 0180 INFO [gossip.privdata] StoreBlock -> Received block [5] from buffer channel=channel1
orgpeer] 2023-04-08T02:13:58:00.158 UTC 0180 INFO [comm.commit.validator] Validate -> [channel1] Validated block [5] in 0ms
orgpeer] 2023-04-08T02:13:58:00.158 UTC 0180 INFO [kvrledger] verify -> [channel1] Committed block [5] with 1 transaction(s) in 11ms (state_validation=13ms block_and_pvtdata_commit=6ms state_commit=
85ms) commitHash=18c72e6ecf7ae17603d3f952d397adcb3573651d95f0f80f4091242435
orgpeer] 2023-04-08T02:13:58:00.843 UTC 0180 INFO [chaincode.externalbuilder.node] WaitForExit -> 2023-04-04T16:02:04.843Z info [c-api:lib/handler.js] [channel1-e3e74e
orgpeer] 2023-04-08T02:13:58:00.843 UTC 0180 INFO [endorser] callChaincode -> finished chaincode: KBA-NAG-Contract duration: 25ms chaincode=channel1 txId=e3e74e
orgpeer] 2023-04-08T02:13:58:00.843 UTC 0180 INFO [comm.grpc.server] 1 - unary call completed grpc.service=protos.Endorser grpc.method=ProcessProposal grpc.peer_address=127.0.0.1:48762 grpc.code=0
K grpc.call_duration=26.06537ms
orgpeer] 2023-04-08T02:13:58:00.843 UTC 0180 INFO [comm.grpc.server] 1 - streaming call completed grpc.service=orderer.Atom1cBroadcast grpc.method=Broadcast grpc.peer_address=127.0.0.1:42580 grpc.
code=OK grpc.call_duration=28.071503ms
orgpeer] 2023-04-08T02:13:58:00.971 UTC 0180 INFO [gossip.privdata] StoreBlock -> Received block [6] from buffer channel=channel1
orgpeer] 2023-04-08T02:13:58:00.972 UTC 0180 INFO [comm.commit.validator] Validate -> [channel1] Validated block [6] in 0ms
orgpeer] 2023-04-08T02:13:58:00.972 UTC 0180 INFO [kvrledger] verify -> [channel1] Committed block [6] with 1 transaction(s) in 30ms (state_validation=6ms block_and_pvtdata_commit=5ms state_commit=2
ms) commitHash=475dc0f7122efb7932d783f3abfd1edd7c535d137b5115cf6e995b38a0

```

Creating new Smart Contract Project:



Code:

```
/*
 * SPDX-License-Identifier: Apache-2.0
 */
'use strict';
const { Contract } = require('fabric-contract-api');
class MangoContract extends Contract {
  async mangoExists(ctx, mangoId) {
    const buffer = await ctx.stub.getState(mangoId);
    return !!buffer && buffer.length > 0;
  }
  async createMango(ctx, mangoId, batchNumber, producer, quantity, price)
  {
    const exists = await this.mangoExists(ctx, mangoId);
    if (exists) {
      throw new Error(`The mango ${mangoId} already exists`);
    }
    const asset = {
      ID: mangoId,
      BatchNumber: batchNumber,
      Producer: producer,
      OwnedBy: producer,
      Quantity: quantity,
      Price: price,
    };
    const buffer = Buffer.from(JSON.stringify(asset));
    await ctx.stub.putState(mangoId, buffer);
  }
  async readMango(ctx, mangoId) {
```

```

const exists = await this.mangoExists(ctx, mangoId);
if (!exists) {
  throw new Error(`The mango ${mangoId} does not exist`);
}
const buffer = await ctx.stub.getState(mangoId);
const asset = JSON.parse(buffer.toString());
return asset;
}

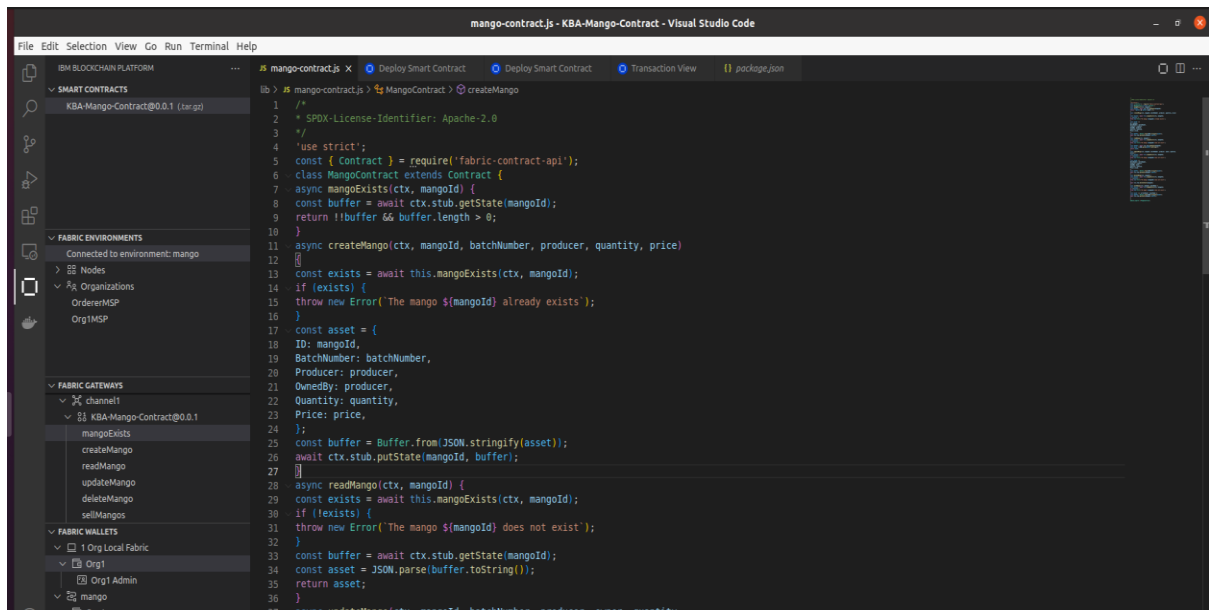
async updateMango(ctx, mangoId, batchNumber, producer, owner, quantity,
price) {
  const exists = await this.mangoExists(ctx, mangoId);
  if (!exists) {
    throw new Error(`The mango ${mangoId} does not exist`);
  }
  const asset = {
    BatchNumber: batchNumber,
    Producer: producer,
    OwnedBy: owner,
    Quantity: quantity,
    Price: price,
  };
  const buffer = Buffer.from(JSON.stringify(asset));
  await ctx.stub.putState(mangoId, buffer);
}

async deleteMango(ctx, mangoId) {
  const exists = await this.mangoExists(ctx, mangoId);
  if (!exists) {
    throw new Error(`The mango ${mangoId} does not exist`);
  }
  await ctx.stub.deleteState(mangoId);
}

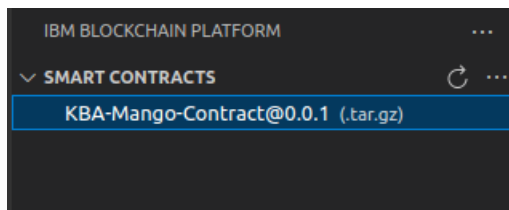
async sellMangos(ctx, mangoId, ownerName) {
  const exists = await this.mangoExists(ctx, mangoId);
  if (!exists) {
    throw new Error(`The apple ${mangoId} does not exist`);
  }
  const asset = { currentOwner: ownerName };
  const buffer = Buffer.from(JSON.stringify(asset));
  await ctx.stub.putState(mangoId, buffer);
}
}

module.exports = MangoContract;

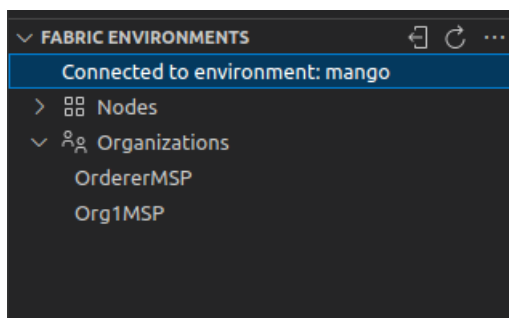
```



Packing Smart Contract:



Deploying Smart Contract:



Making Transactions:

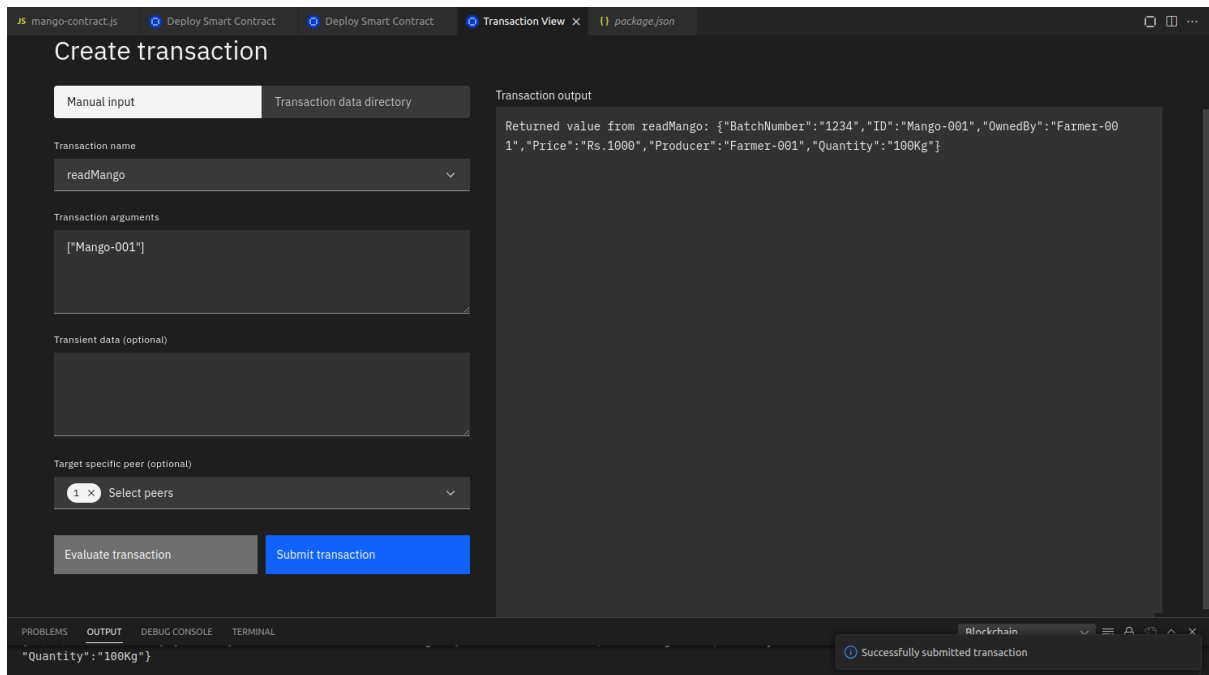
mangoExists:

The screenshot shows the 'Transaction View' window in a development environment. The title bar includes tabs for 'mango-contract.js', two 'Deploy Smart Contract' buttons, 'Transaction View', and 'package.json'. The main area is titled 'Create transaction' and is split into two panels. The left panel, 'Manual input', contains a 'Transaction name' dropdown set to 'mangoExists', a 'Transaction arguments' text area with the value '["Mango-001"]', and an empty 'Transient data (optional)' text area. The right panel, 'Transaction output', displays the text 'Returned value from mangoExists: false'.

createMango:

The screenshot shows the 'Transaction View' window for the 'createMango' transaction. The interface is similar to the previous one but includes an additional field. The 'Transaction name' dropdown is set to 'createMango'. The 'Transaction arguments' text area contains the value '["Mango-001","1234","Farmer-001","100Kg","Rs.1000"]'. Below this is an empty 'Transient data (optional)' text area. A new 'Target specific peer (optional)' section is present, featuring a dropdown menu with '1 x Select peers'. At the bottom of the left panel are two buttons: 'Evaluate transaction' and 'Submit transaction'. The 'Transaction output' panel on the right displays the text 'No value returned from createMango'.

Readmango:



Result:

Hence, we have successfully built a Hyperledger Fabric using IBM Blockchain Platform and verified the Transactions.