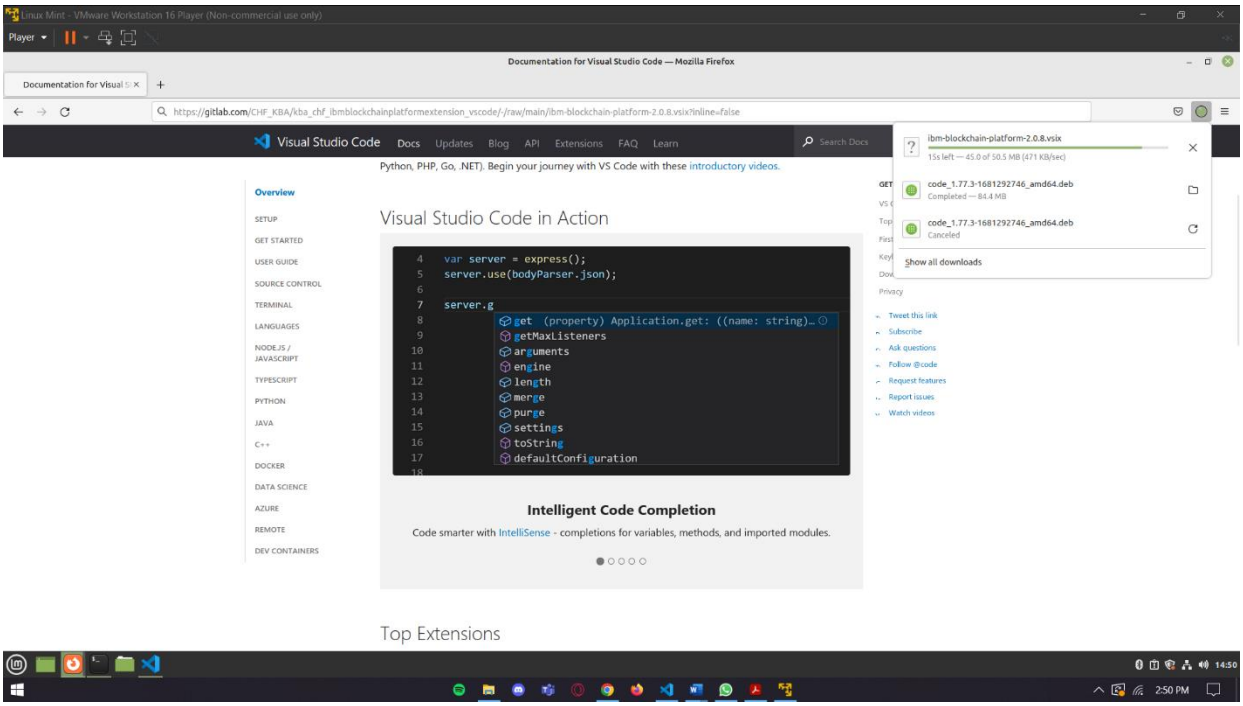
NAME: ANKIT KUMAR JHA
REG.NO.20BPS1050

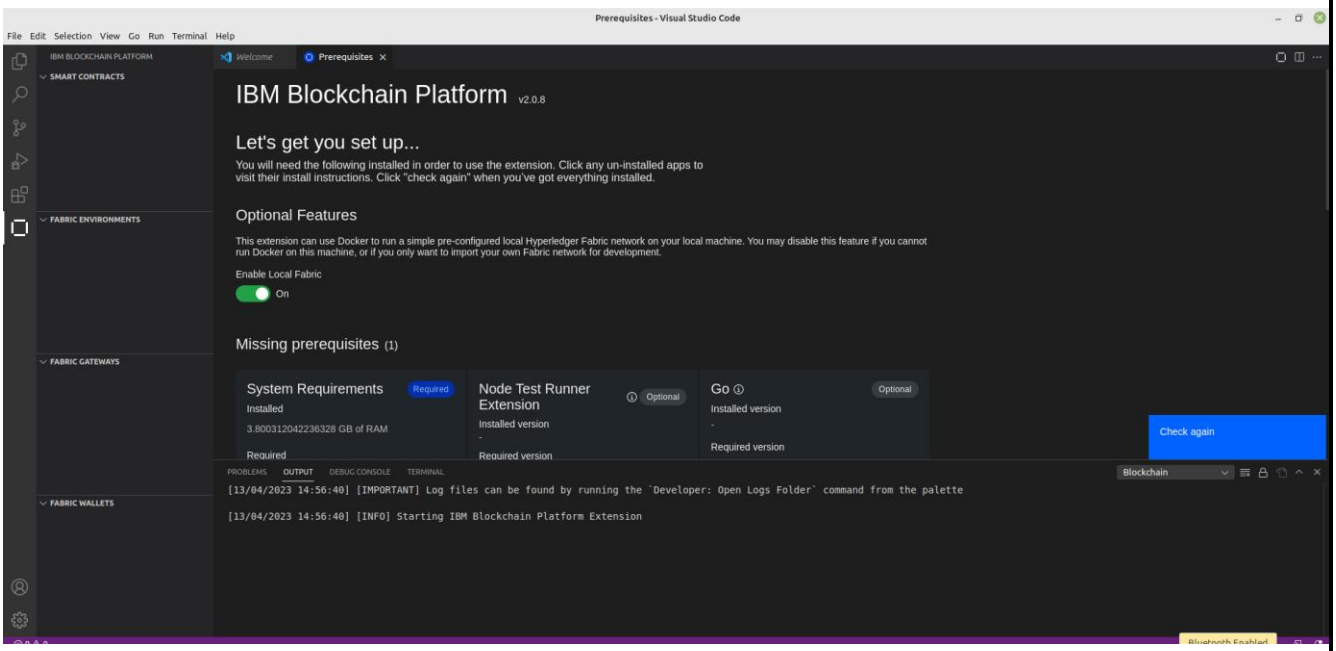
EXERCISE 5
Hyperledger fabric using IBM Blockchain platform

Downloading Prerequisites

- Visual Studio Code
- IBM Blockchain Platform extension
-



Installing Extension



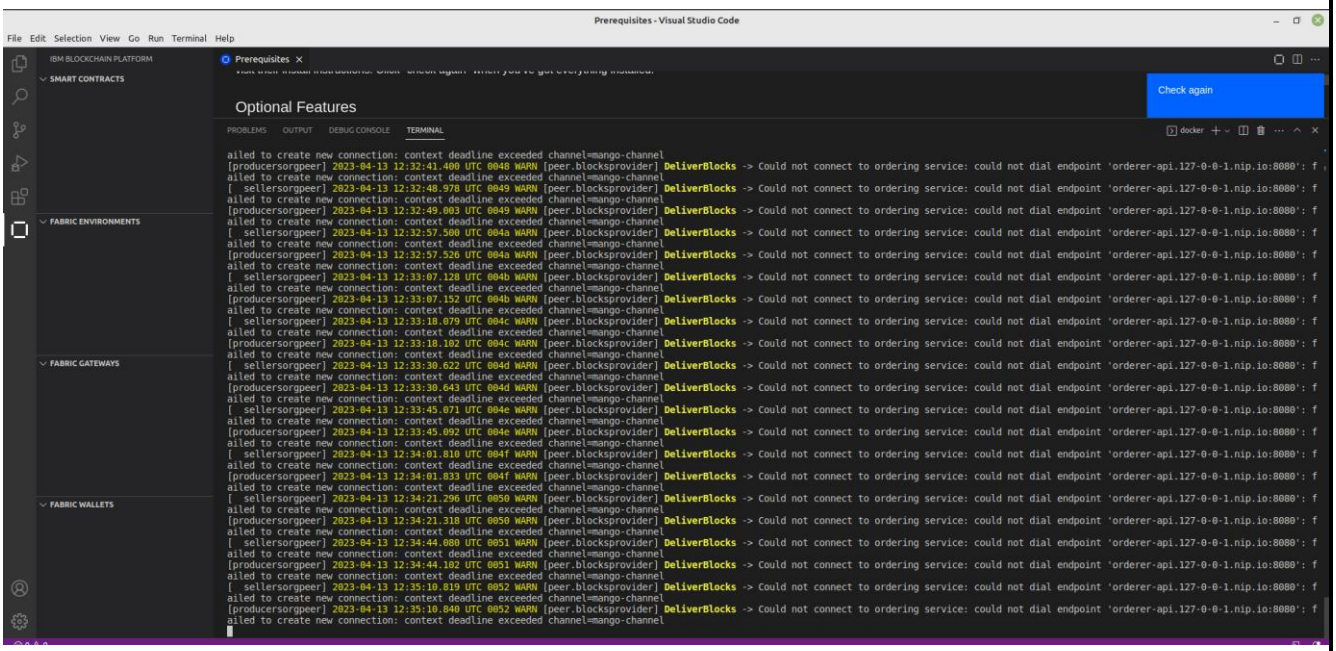
Configuration

```
$ export MICROFAB_CONFIG='{
> "port": 8080,
> "endorsing_organizations":[
> {
> "name": "ProducersOrg"
> },
> {
> "name": "SellersOrg"
> }
> ],
> "channels":[
> {
> "name": "mango-channel",
> "endorsing_organizations":[
> "ProducersOrg",
> "SellersOrg"
> ]
> }
> ]
> }'
```

Running the Network

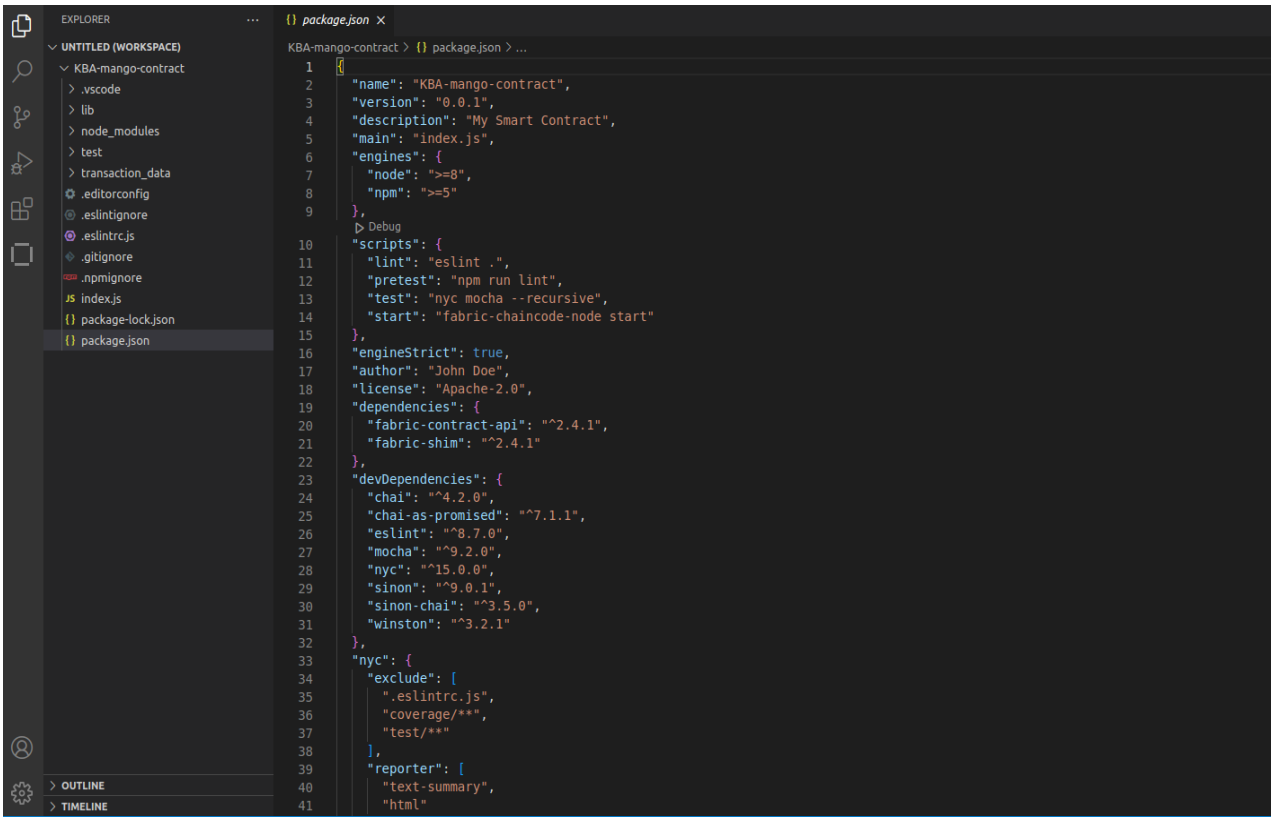
```
$ docker run -e MICROFAB_CONFIG -p 8080:8080 ibmcom/ibp-microfab
```

```
Unable to find image 'ibmcom/ibp-microfab:latest' locally
latest: Pulling from ibmcom/ibp-microfab
d6e3788a121c: Pull complete
b336a633f9e2: Pull complete
b25deee7e50f: Pull complete
9041dff50abb: Downloading [=======>                                           ]  30.7MB/196MB
c50c12d7db31: Download complete
5186d5863148: Download complete
531dcea8d07e: Download complete
690939f6038f: Download complete
f24a19b34ee1: Downloading [=>                                                 ]  12.94MB/363.1MB
576db8757925: Downloading  1.619MB
734dce7f474b: Waiting
eb98c0bd8a72: Waiting
65606c3065d6: Waiting
2785a9abab40: Waiting
78f82eb67606: Waiting
```

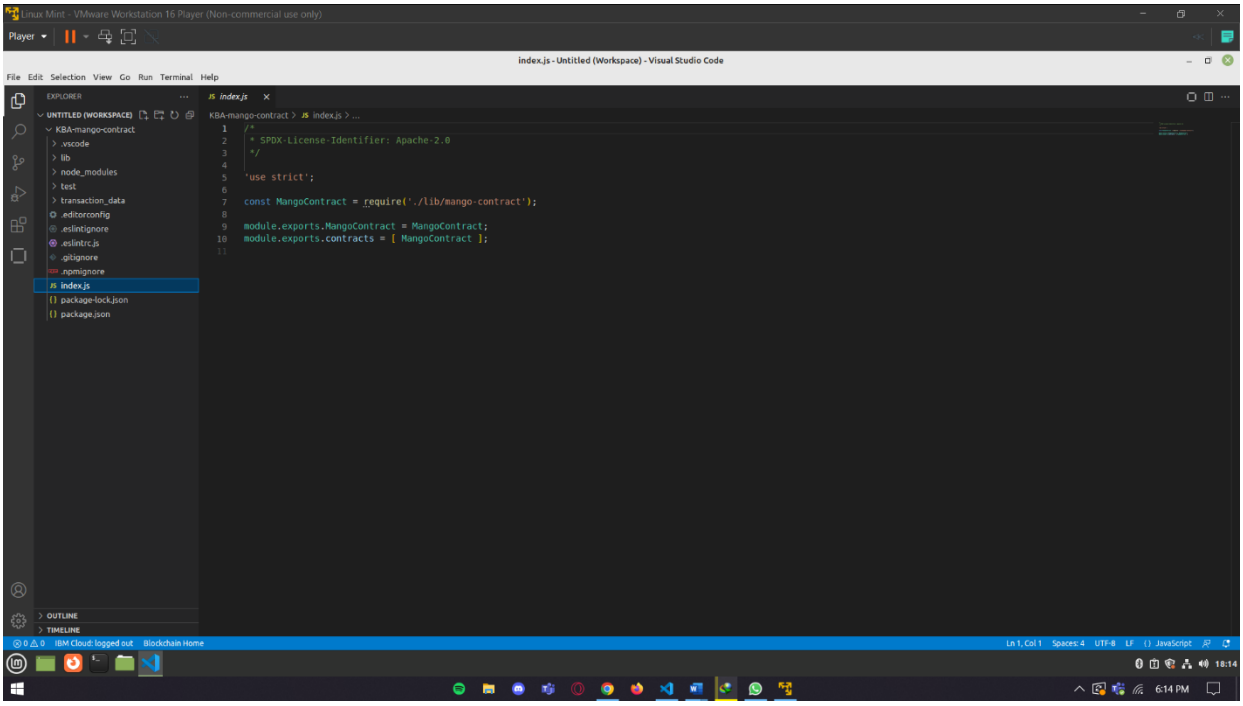


Smart Contract project

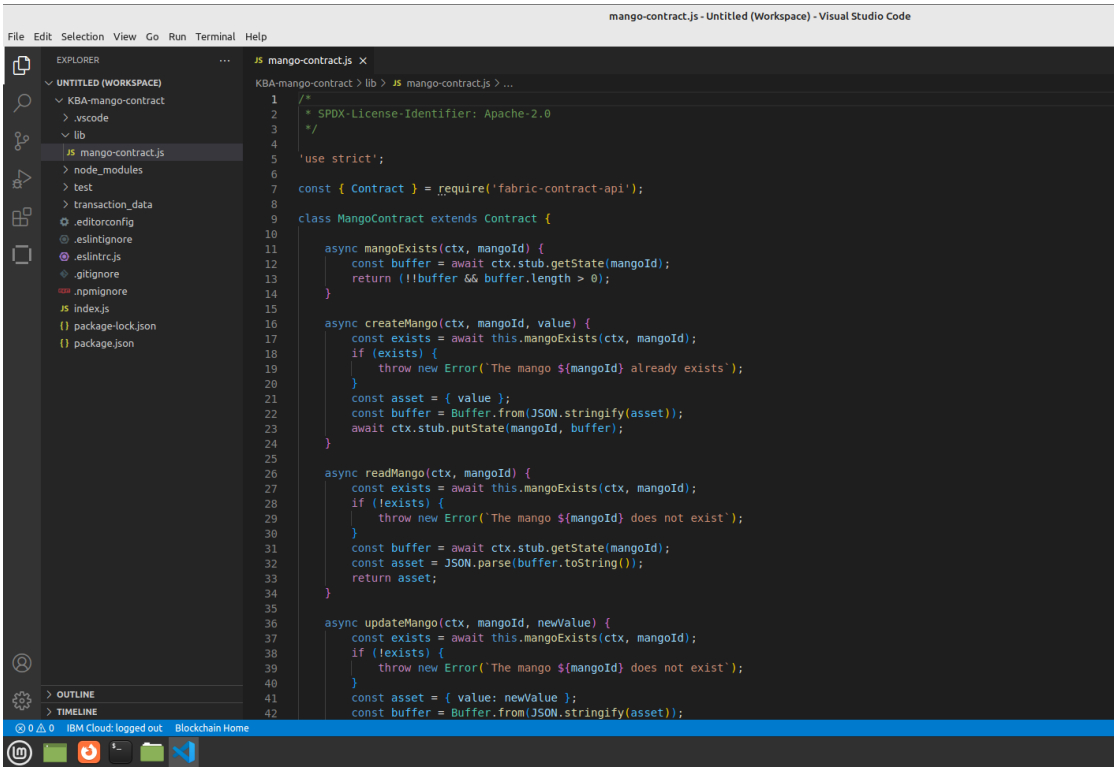
Package.json:



Index.js

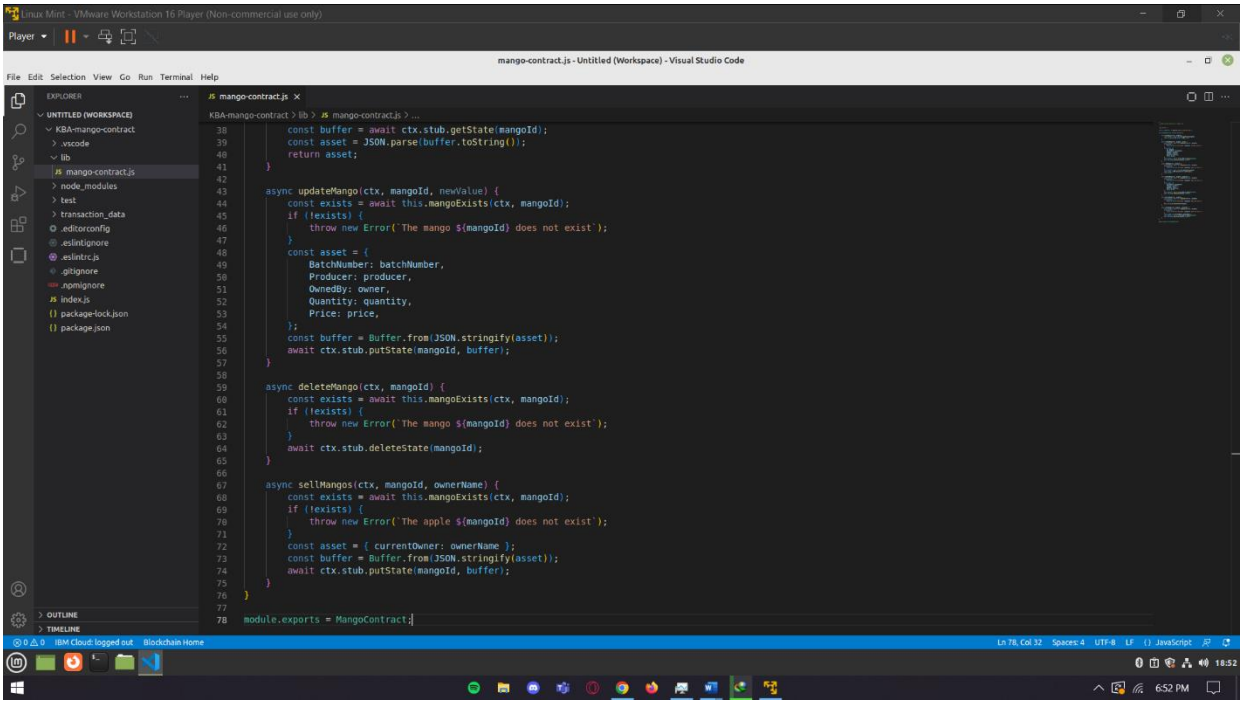


Mango-contract.js



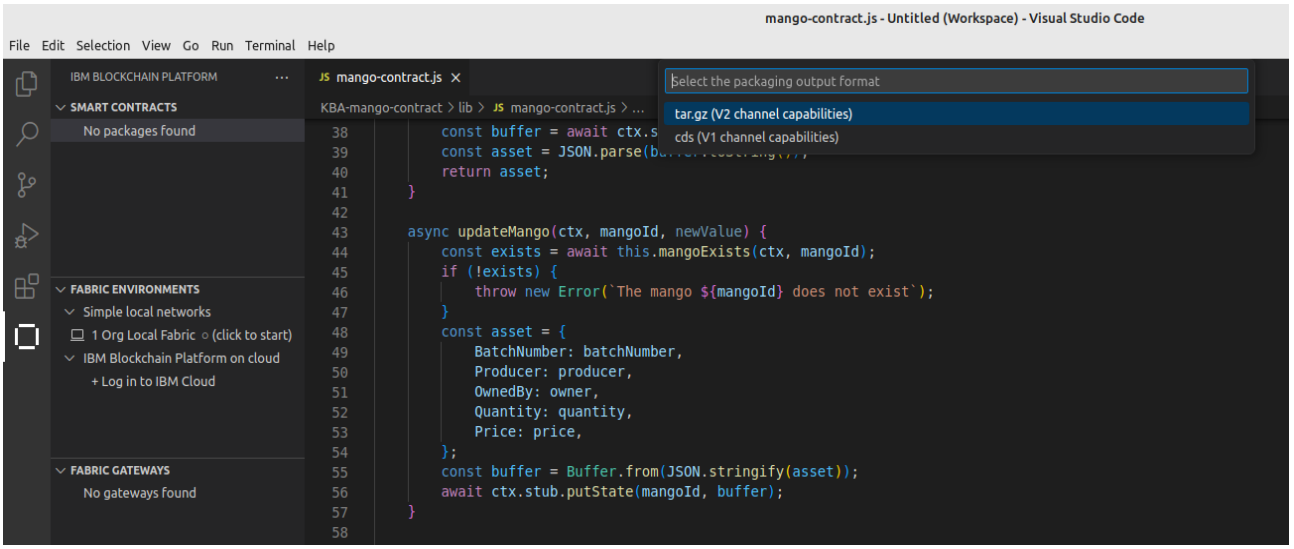
Adding More Functionalities

Mango-contract.js

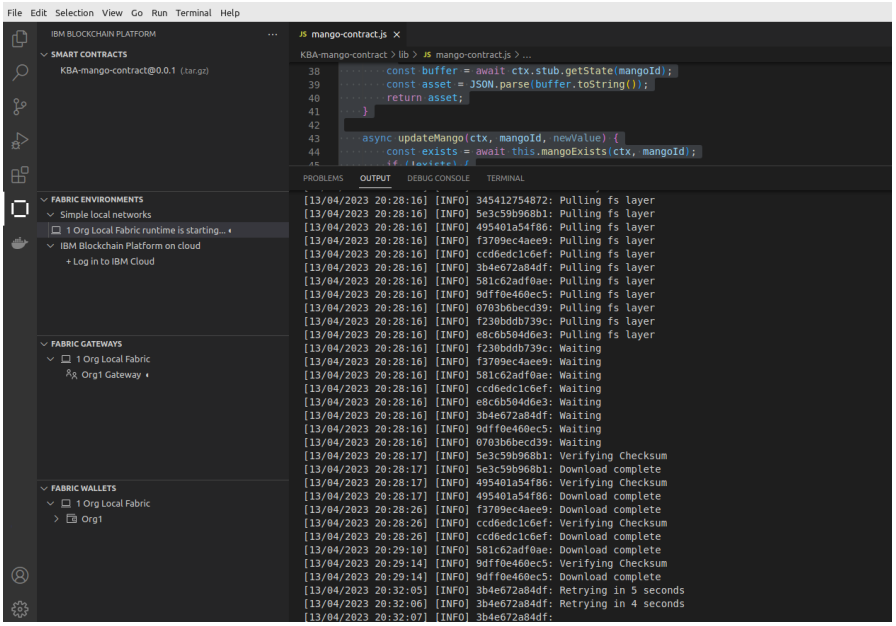


```
/*
 * SPDX-License-Identifier: Apache-2.0
 */

'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, value) {
        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, newValue) {
        const exists = await this.mangoExists(ctx, mangoId);
        if (!exists) {
            throw new Error(`The mango ${mangoId} does not exist`);
        }
        const asset = {

BatchNumber: batchNumber,
```

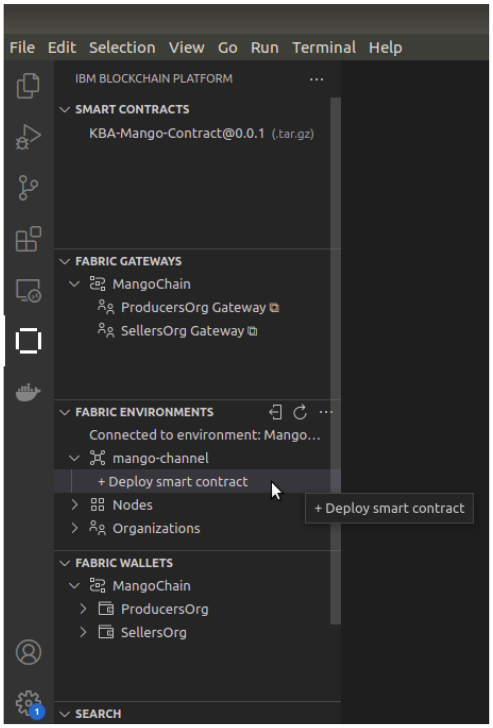
```
            Producer: producer,
            OwnedBy: owner,
            Quantity: quantity,
            Price: price,
        };
        const buffer = Buffer.from(JSON.stringify(asset));
        await ctx.stub.putState(mangoId, buffer);
    }

    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;
```



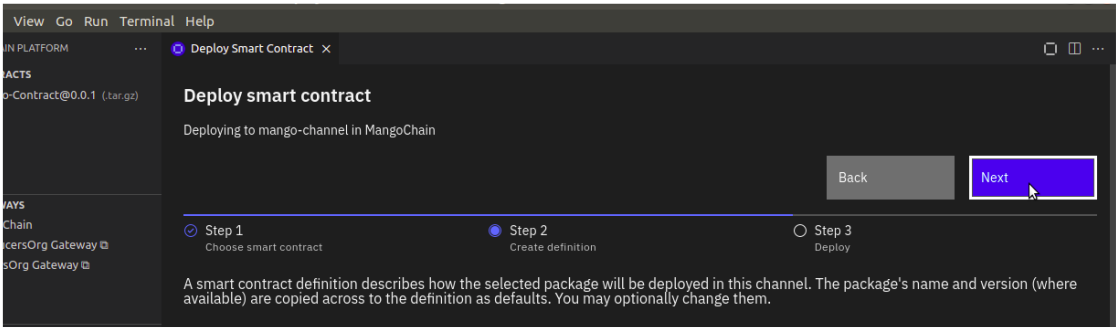
Fabric Environments



Deploy Smart Contract

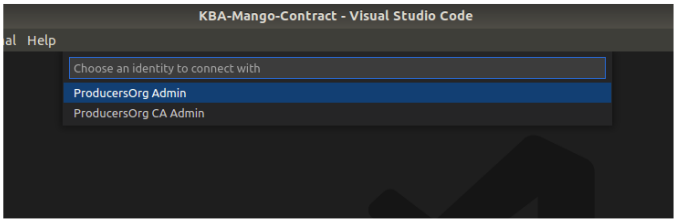


Create Smart Contract Definition

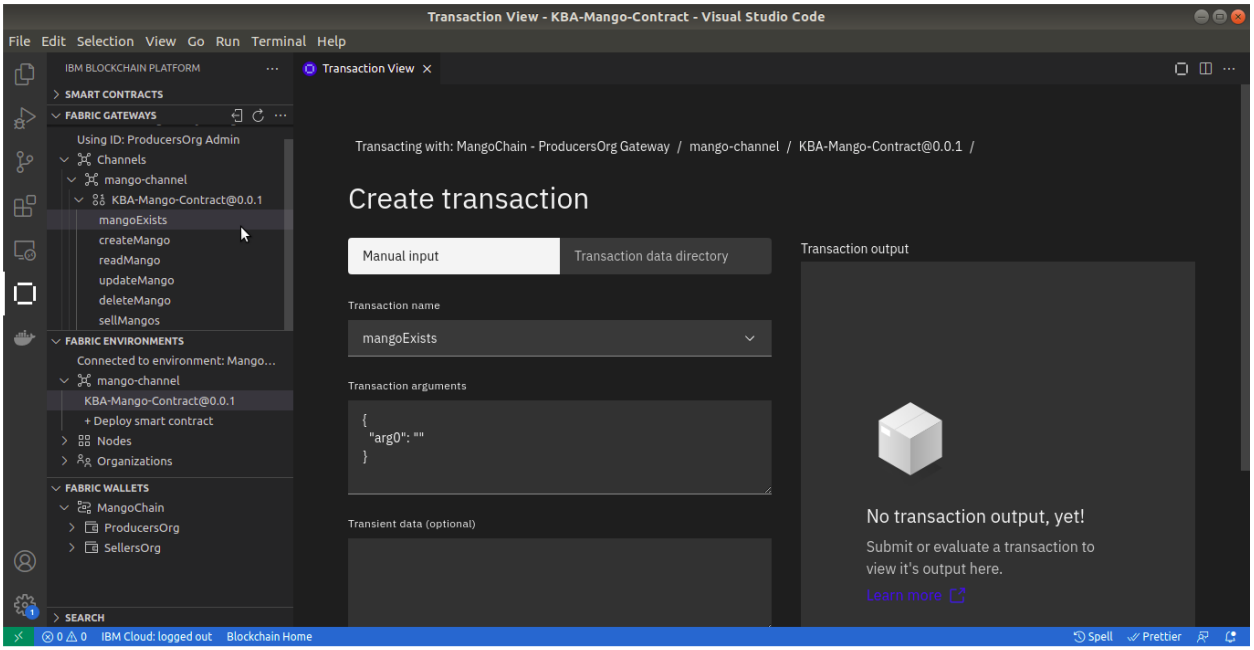


Making Transactions

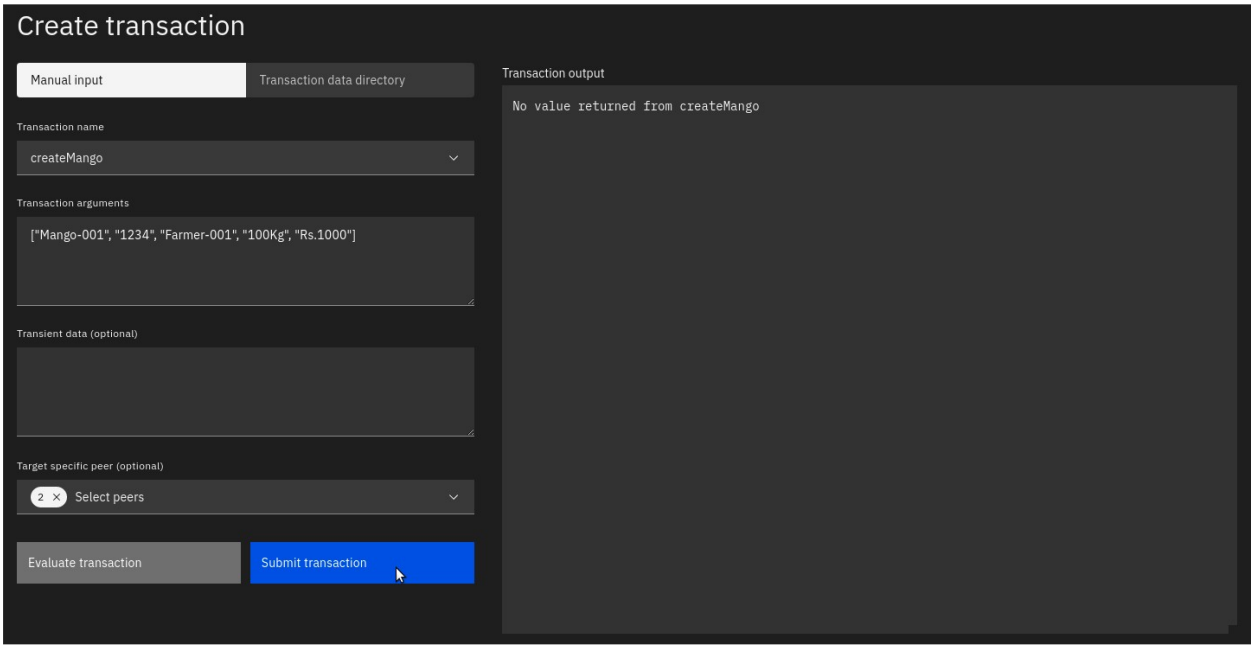
Fabric Gateway



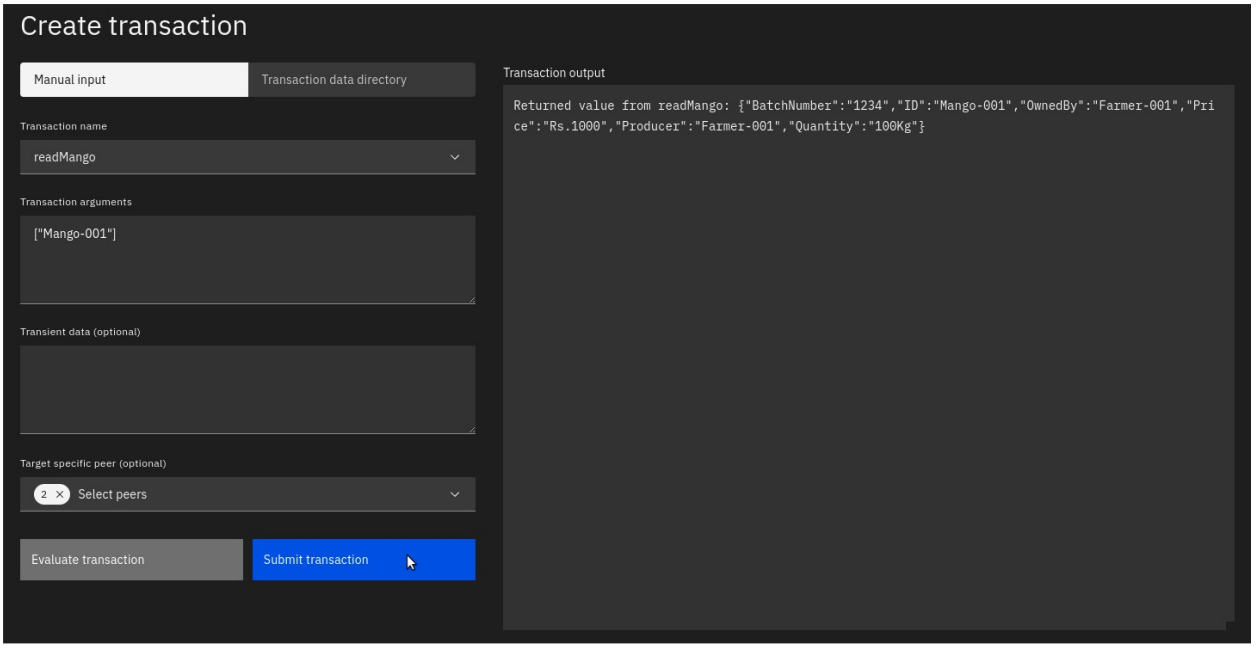
mangoExists



createMango



readMango


Create transaction
Manual input
Transaction data directory
Transaction name
readMango
Transaction arguments
["Mango-001"]
Transient data (optional)
Target specific peer (optional)
2
Select peers
Evaluate transaction
Submit transaction
Transaction output
Returned value from readMango: {"BatchNumber":"1234","ID":"Mango-001","OwnedBy":"Farmer-001","Price":"Rs.1000","Producer":"Farmer-001","Quantity":"100Kg"}