Assignment No : 17

Title of the Assignment: Write a program to create a Business Network using Hyperledger.

Objective of the Assignment: Students should be able to learn hyperledger .Its application and implementations

Prerequisite:

1. Basic knowledge of cryptocurrency
2. Basic knowledge of distributed computing concept
3. Working of blockchain

Contents for Theory:

Hyperledger Composer is an extensive, open development toolset and framework to make developing blockchain applications easier. The primary goal is to accelerate time to value, and make it easier to integrate your blockchain applications with the existing business systems.

- You can use Composer to rapidly develop use cases and deploy a blockchain solution in days.
- Composer allows you to model your business network and integrate existing systems and data with your blockchain applications.
- Hyperledger Composer supports the existing Hyperledger Fabric blockchain infrastructure and runtime.
- Hyperleder Composer generate business network archive (bna) file which you can deploy on existing Hyperledger Fabric network

You can use Hyperledger Composer to model business network, containing your existing assets and the transactions related to them

Key Concepts of Hyperledger Composer

1. Blockchain State Storage: It stores all transaction that happens in your hyperledger composer application. It stores transaction in Hyperledger fabric network.

2. Connection Profiles: Connection Profiles to configuration JSON file which help composer to connect to Hyperledger Fabric. You can find Connection Profile JSON file in user's home directory.

3. Assets: Assets are tangible or intangible goods, services, or property, and are stored in registries. Assets can represent almost anything in a business network, for example, a house for sale, the sale listing, the land registry certificate for that house. Assets must have a unique identifier, but other than that, they can contain whatever properties you define.

4. Participants: Participants are members of a business network. They may own assets and submit transactions. Participant must have an identifier and can have any other properties.

5. Identities and ID cards: Participants can be associated with an identity. ID cards are a combination of an identity, a connection profile, and metadata. ID cards simplify the process of connecting to a business network.

6. Transactions: Transactions are the mechanism by which participants interact with assets. Transaction processing logic you can define in JavaScript and you can also emit event for transaction.

7. Queries: Queries are used to return data about the blockchain world-state. Queries are defined within a business network, and can include variable parameters for simple customisation. By using queries, data can be easily extracted from your blockchain network. Queries are sent by using the Hyperledger Composer API.

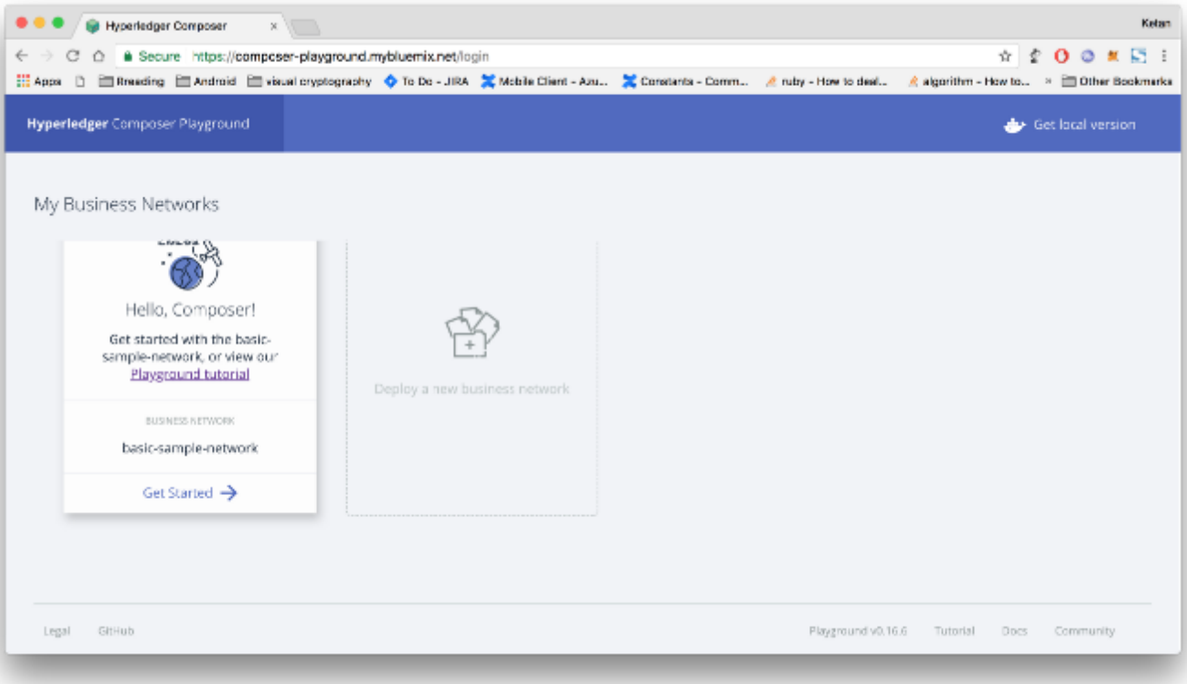
8. Events: Events are defined in the model file. Once events have been defined, they can be emitted by transaction processor functions to indicate to external systems that something of importance has happened to the ledger.

9. Access Control: Hyperledger is enterprise blockchain and access control is core feature of any business blockchain. Using Access Control rules you can define who can do what in Business networks. The access control language is rich enough to capture sophisticated conditions.

10. Historian registry: The historian is a specialised registry which records successful transactions, including the participants and identities that submitted them. The historian stores transactions as HistorianRecord assets, which are defined in the Hyperledger Composer system namespace.

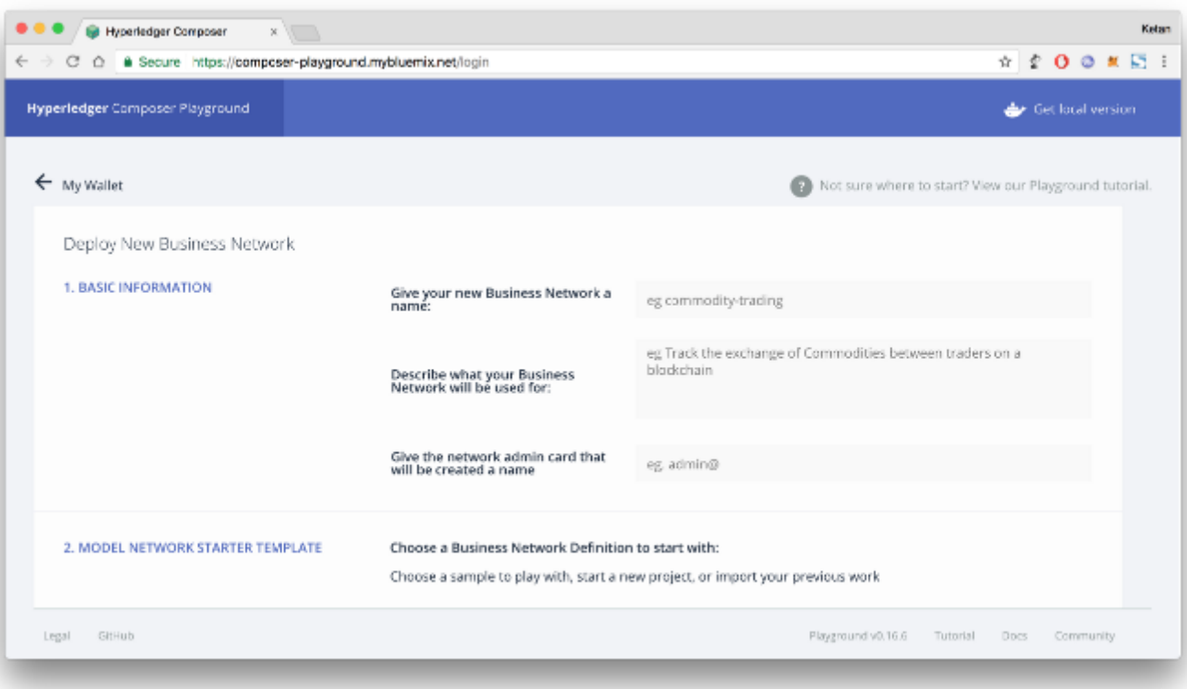
Let's create first Hyperledger Composer Application

Step 1: Start Hyperledger Composer Online version of Local. Click on Deploy a new business network

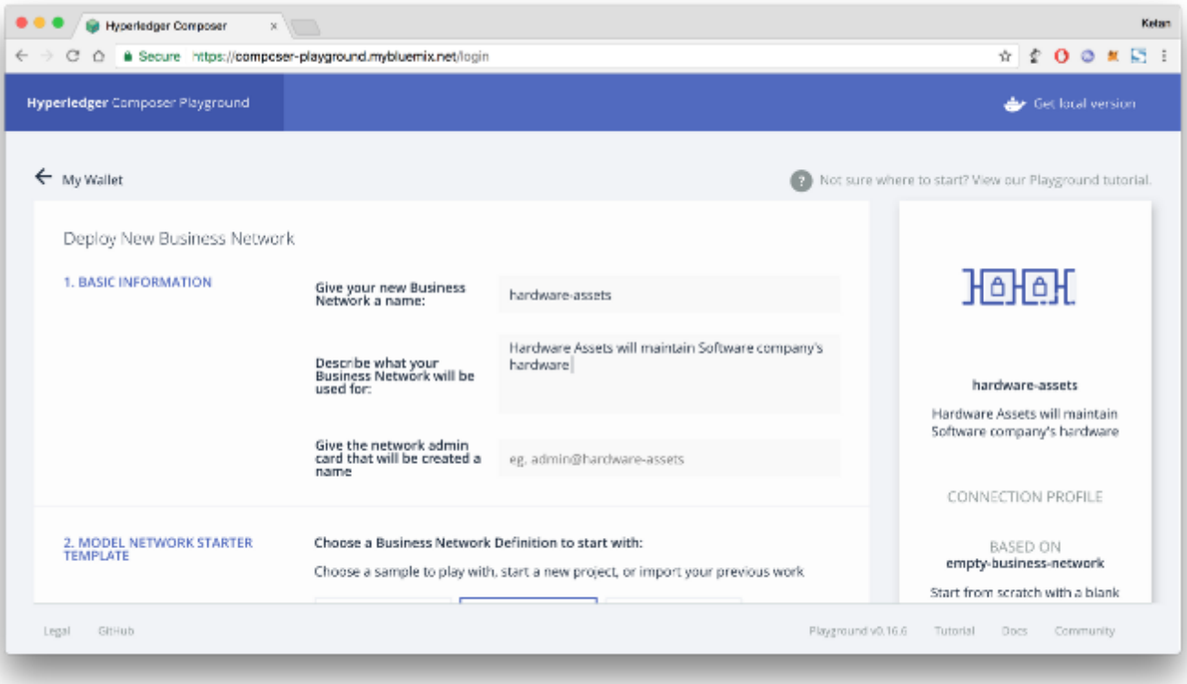


Hyperledger Composer Playground Online version

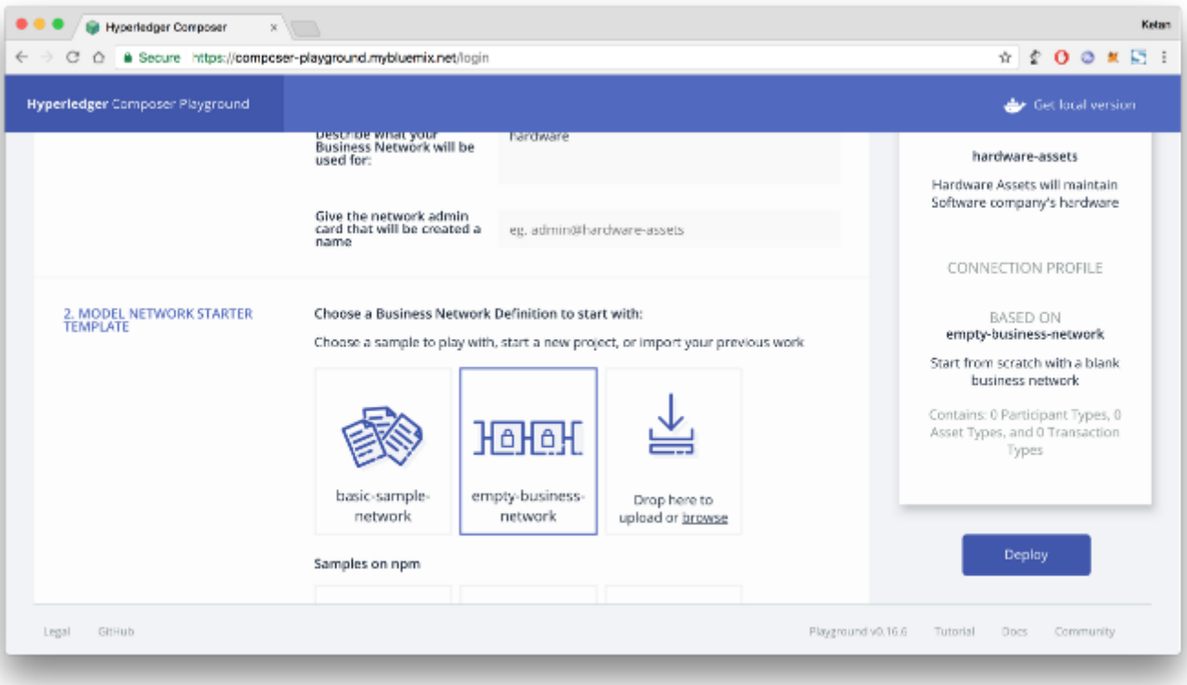
Step 2: Select empty business network



Step 3: Fill basic information, select empty business network and click “deploy” button from right pannel

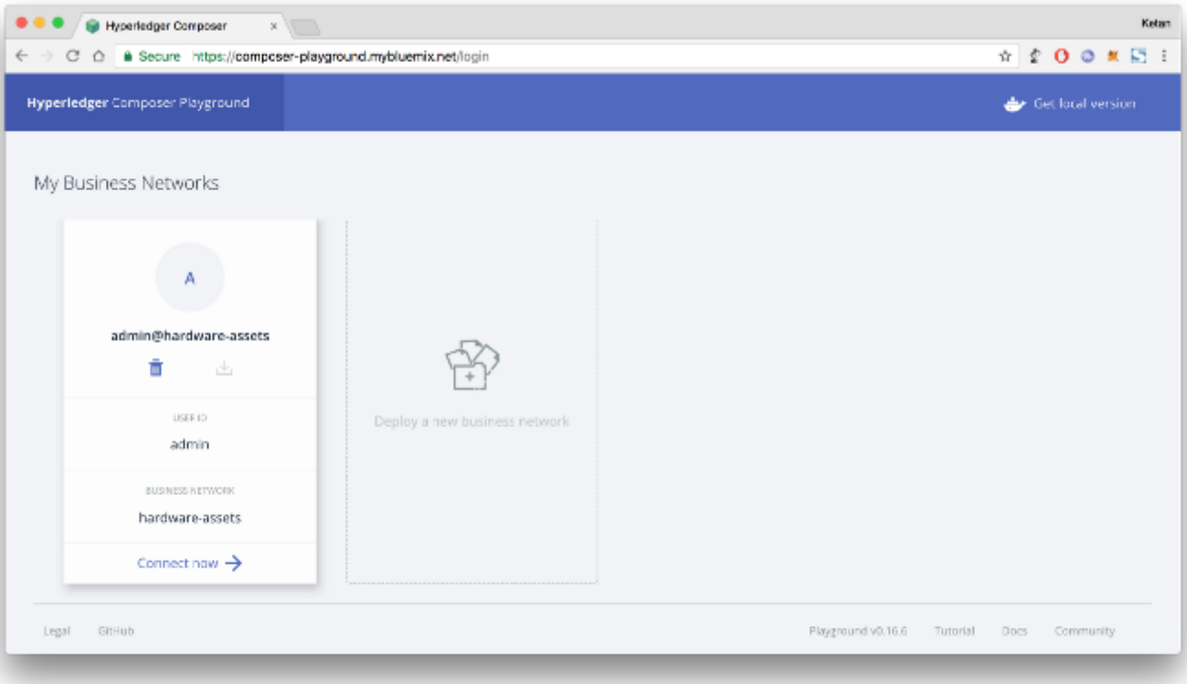


Fill basic information

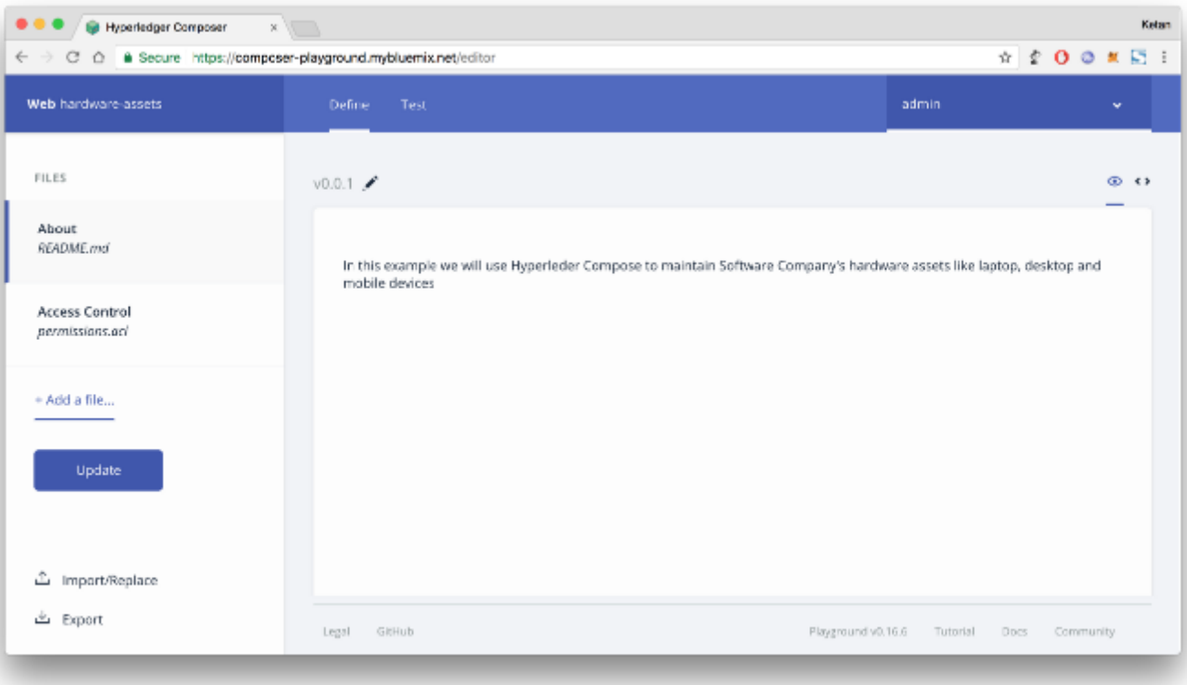


select empty business network

Step 4: Connect to “hardware-assets” business network that we have just deploied

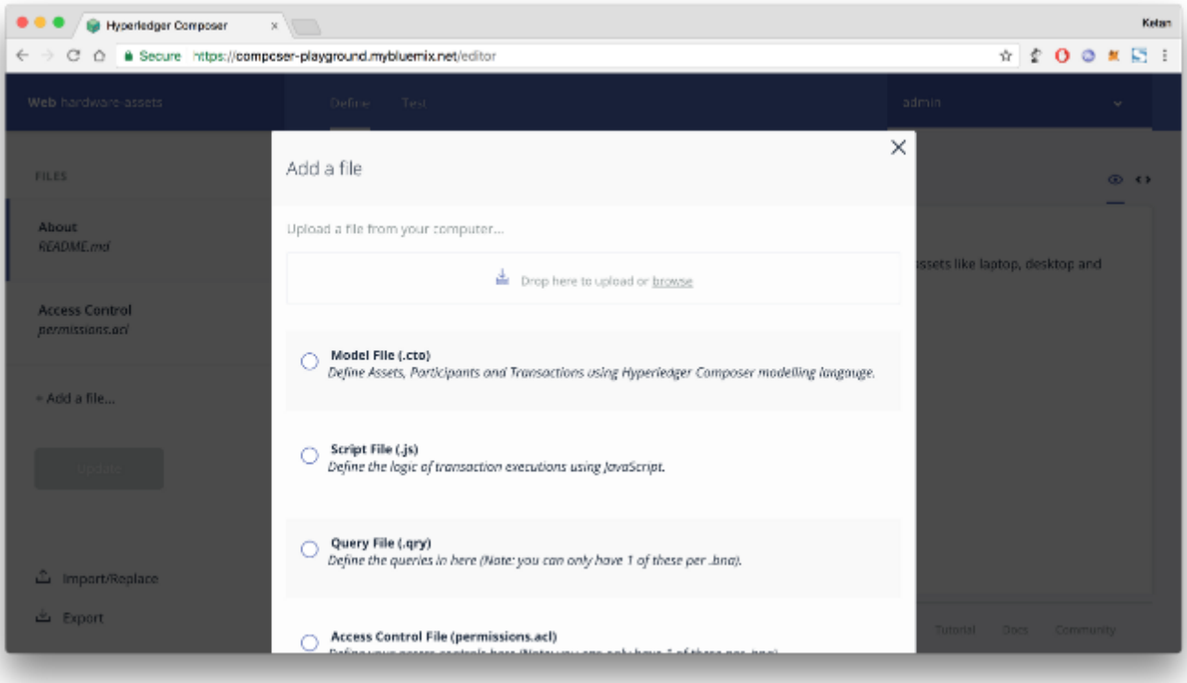


click on “connect now” button



Inside hardware-assets business network

Step 5: Click on “+Add a file…” from left panel and select “model file (.cto)”



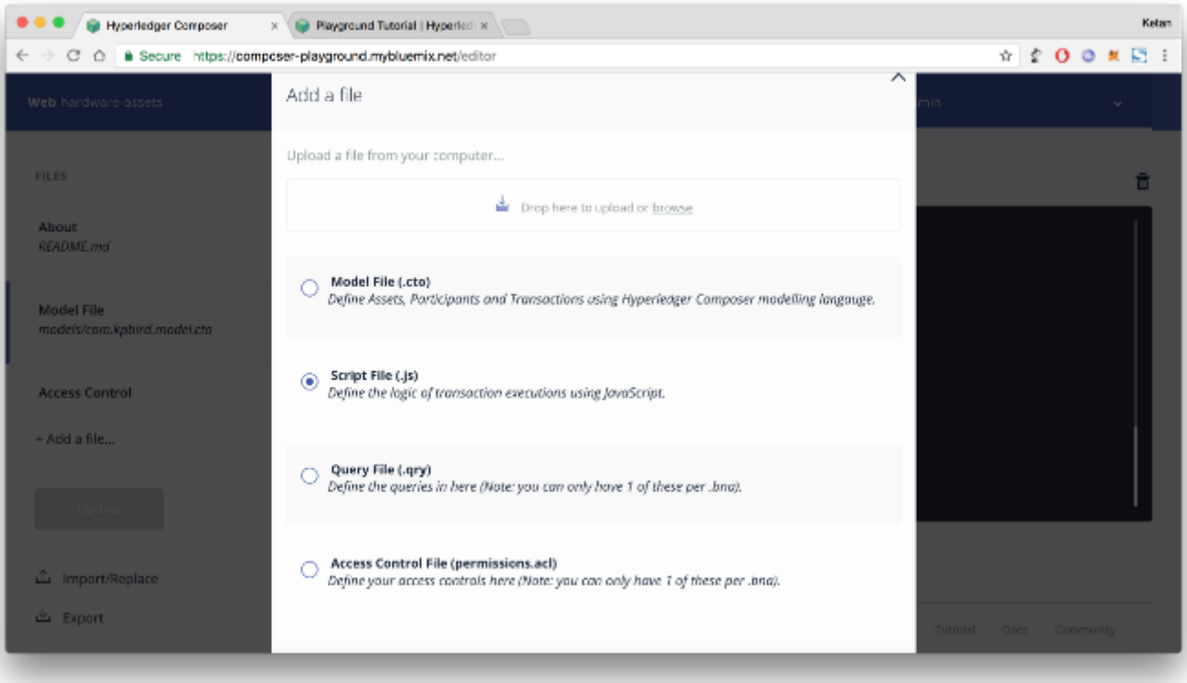
Write following code in model file. Model file contain asset in our case it's hardware, participant in our case participants are employee of organisation and transaction as Allocate hardware to employee. Each model has extra properties. Make sure your have proper and unique namespace. In this example I am using "com.kpbird" as namespace. You can access all models using this namespace i.e. com.kpbird.Hardware, com.kpbird.Employee

```
/**
 * Hardware model
 */namespace com.kpbirdasset Hardware identified by hardwareId {
 o String hardwareId
 o String name
 o String type
 o String description
 o Double quantity
 → Employee owner
}
participant Employee identified by employeeId {
 o String employeeId
 o String firstName
 o String lastName
}
transaction Allocate {
 → Hardware hardware
 → Employee newOwner
}
```

Hyperledger modeling language

reference: https://hyperledger.github.io/composer/reference/cto_language.html

Step 6: Click on "+Add a file…" from left panel and select "script file (*.js)"



Write following code in Script File. In Script we can define transaction processing logic. In our case we want to allocate hardware to the employee so, we will update owner of hardware. Make sure about annotation above functions @params and @transaction

```
/**
 * Track the trade of a commodity from one trader to another
 * @param {com.kpbird.Allocate} trade – the trade to be processed
 * @transaction
 */
function allocateHardware(allocate) {
 allocate.hardware.owner = allocate.newOwner;
 return getAssetRegistry('com.kpbird.Hardware')
 .then(function (assetRegistry) {
 return assetRegistry.update(allocate.hardware);
 });
}
```

Hyperledger Composer Script file reference: https://hyperledger.github.io/composer/reference/js_scripts.html

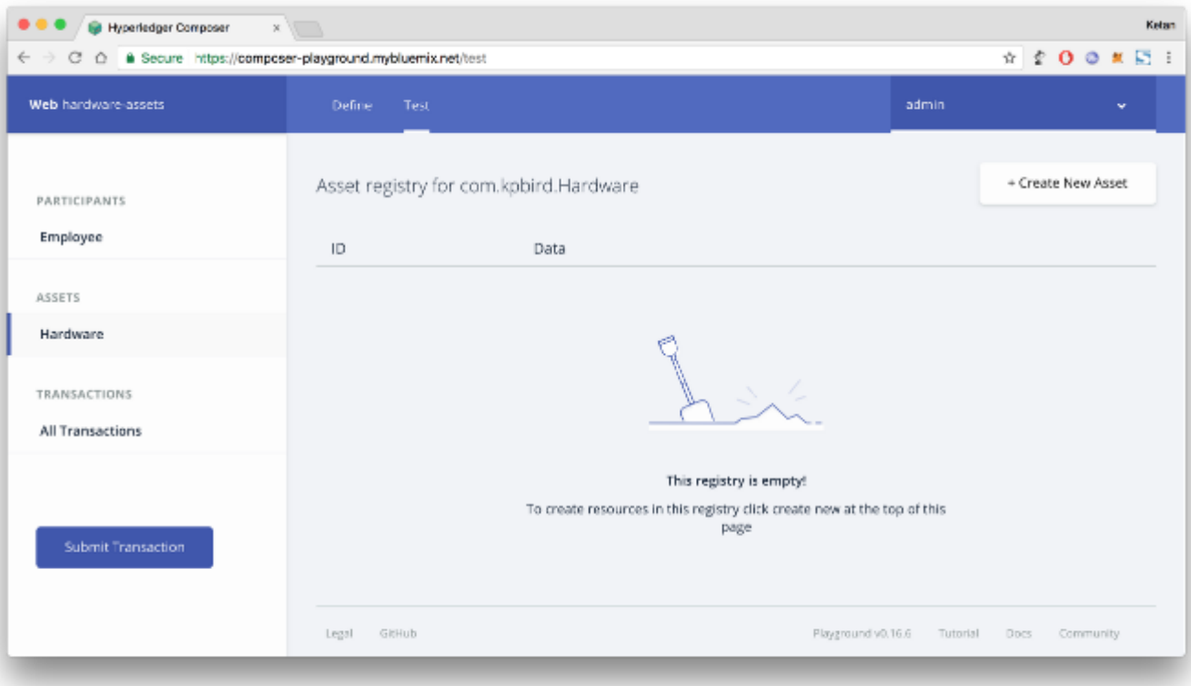
Step 7: permissions.acl file sample is already available, Add following code in permissions.acl file.

```
/**
 * New access control file
 */
 rule AllAccess {
 description: "AllAccess – grant everything to everybody."
 participant: "ANY"
 operation: ALL
 resource: "com.kpbird.**"
 action: ALLOW
 }rule SystemACL{
 description: "System ACL to permit all access"
 participant: "org.hyperledger.composer.system.Participant"
 operation: ALL
 resource: "org.hyperledger.composer.system.**"
 action: ALLOW
}
```

Hyperledger Composer Access Control Language

reference: https://hyperledger.github.io/composer/reference/acl_language.html

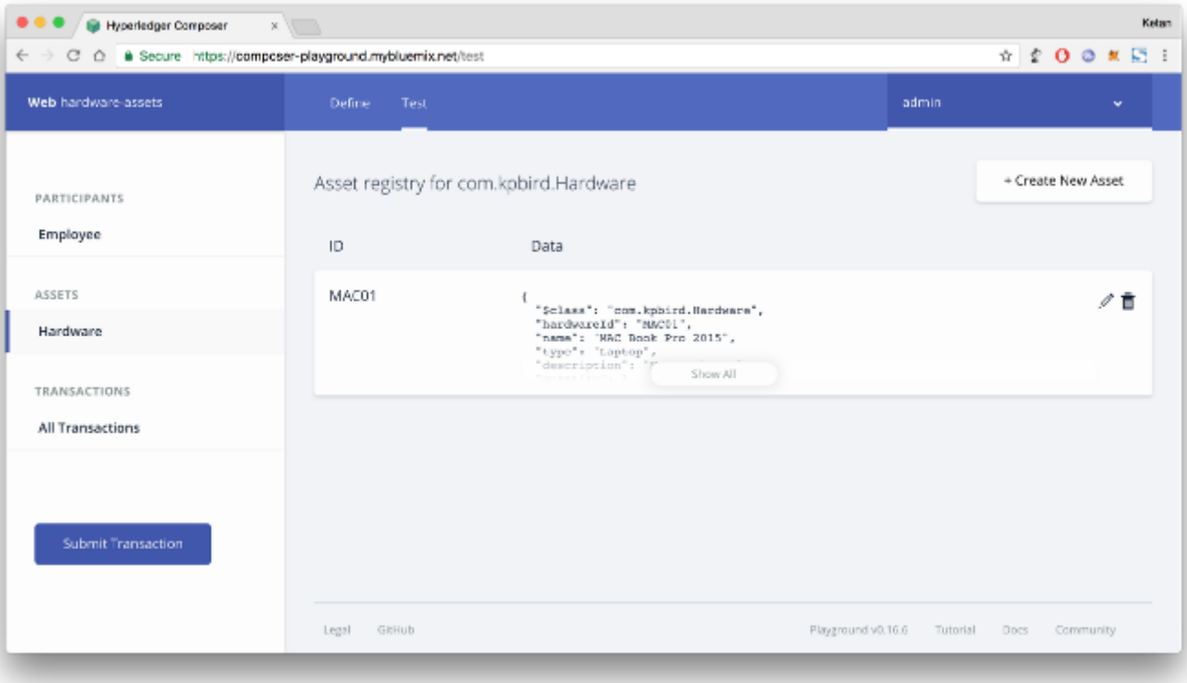
Step 8: Now, It's time to test our hardware assets business network. Hyperledger composer gives "Test" facility from composer panel it self. Click on "Test" tab from top panel



Test feature of Hyperledger Composer

Step 9: Create Assets. Click on "Hardware" from left panel and click "+ Create New Assets" from right top corner and add following code. We will create Employee#01 in next step. Click on "Create New" button

```
{
 "$class": "com.kpbird.Hardware",
"hardwareId": "MAC01",
"name": "MAC Book Pro 2015",
"type": "Laptop",
"description": "Mac Book Pro",
"quantity": 1,
"owner": "resource:com.kpbird.Employee#01"
}
```



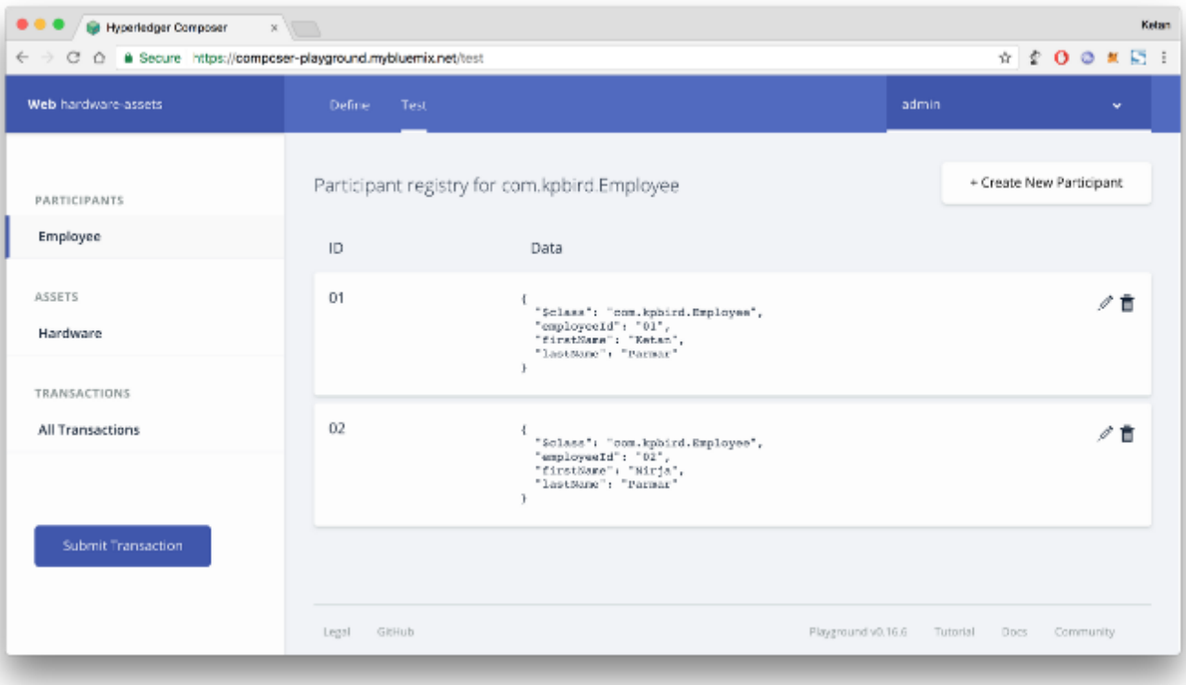
After adding Hardware assets

Steps 10: Let's create participants. Click "Employee" and click "+ Create New Participants" and add following code. We will add two employees

```
{
 "$class": "com.kpbird.Employee",
 "employeeId": "01",
 "firstName": "Ketan",
 "lastName": "Parmar"
}
```

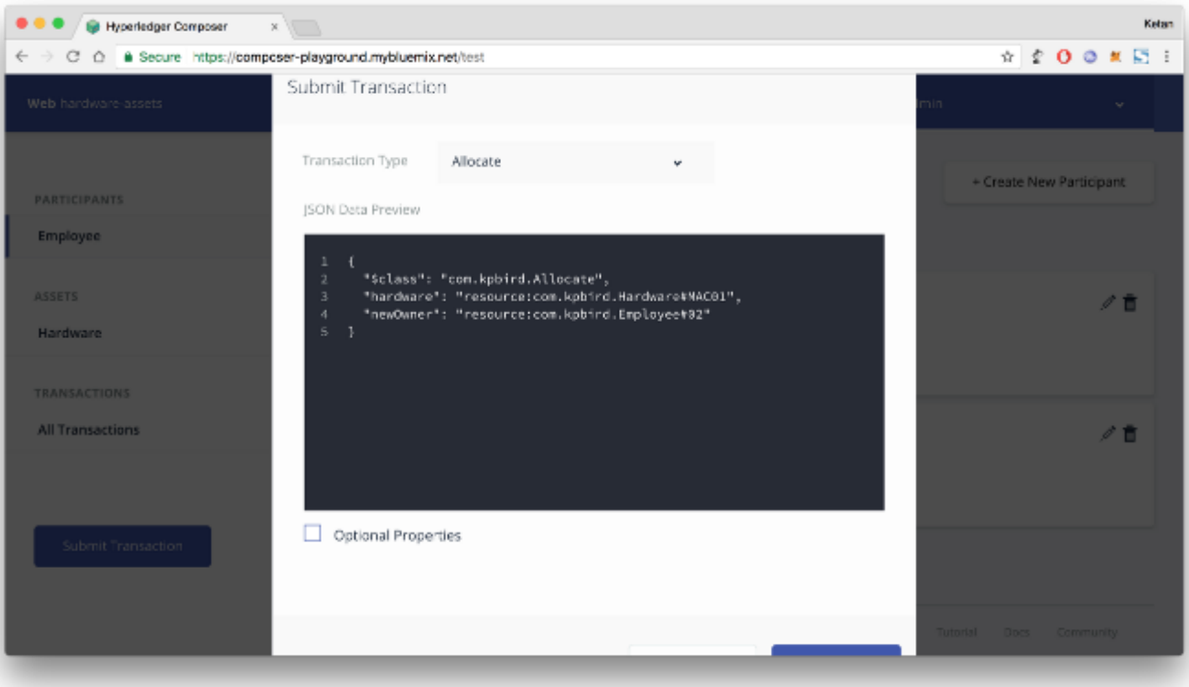
Click on "Create New" on dialog

```
{
 "$class": "com.kpbird.Employee",
 "employeeId": "02",
 "firstName": "Nirja",
 "lastName": "Parmar"
}
```



We have two employees

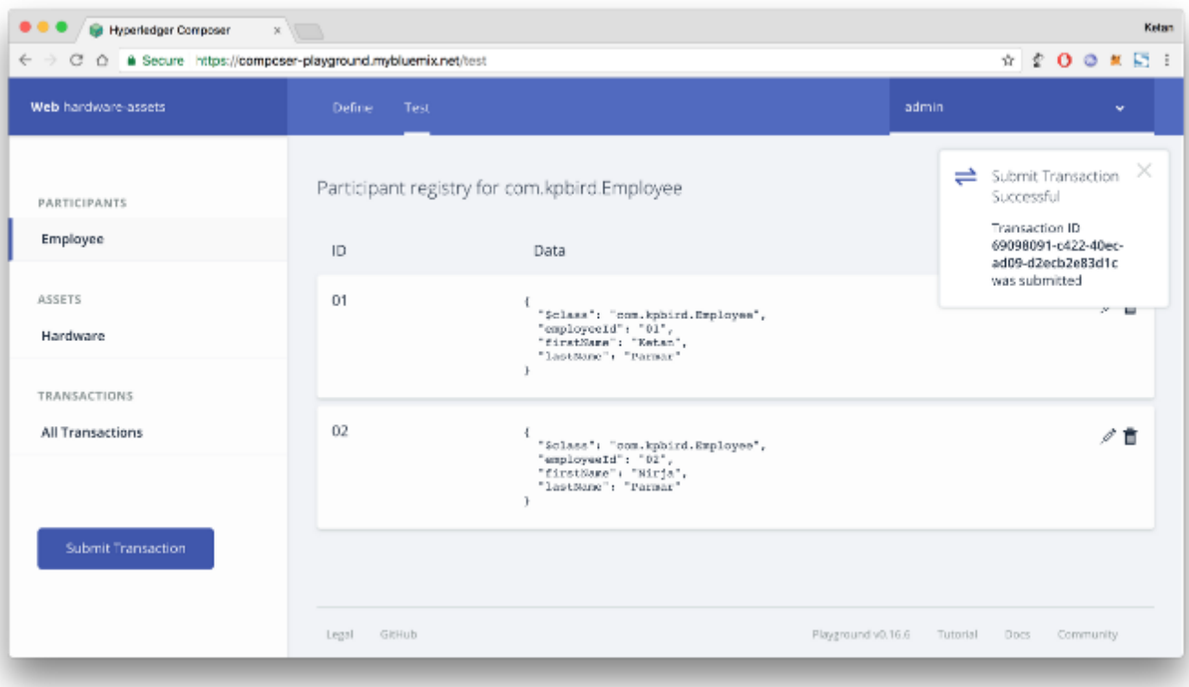
Step 11: It's time to do transaction, We will allocate Macbook Pro from Ketan (Employee#01) to Nirja (Employee#02). Click on "Submit Transaction" button from left panel. In Transaction dialog, We can see all transaction functions on top "Transaction Type" dropdown.

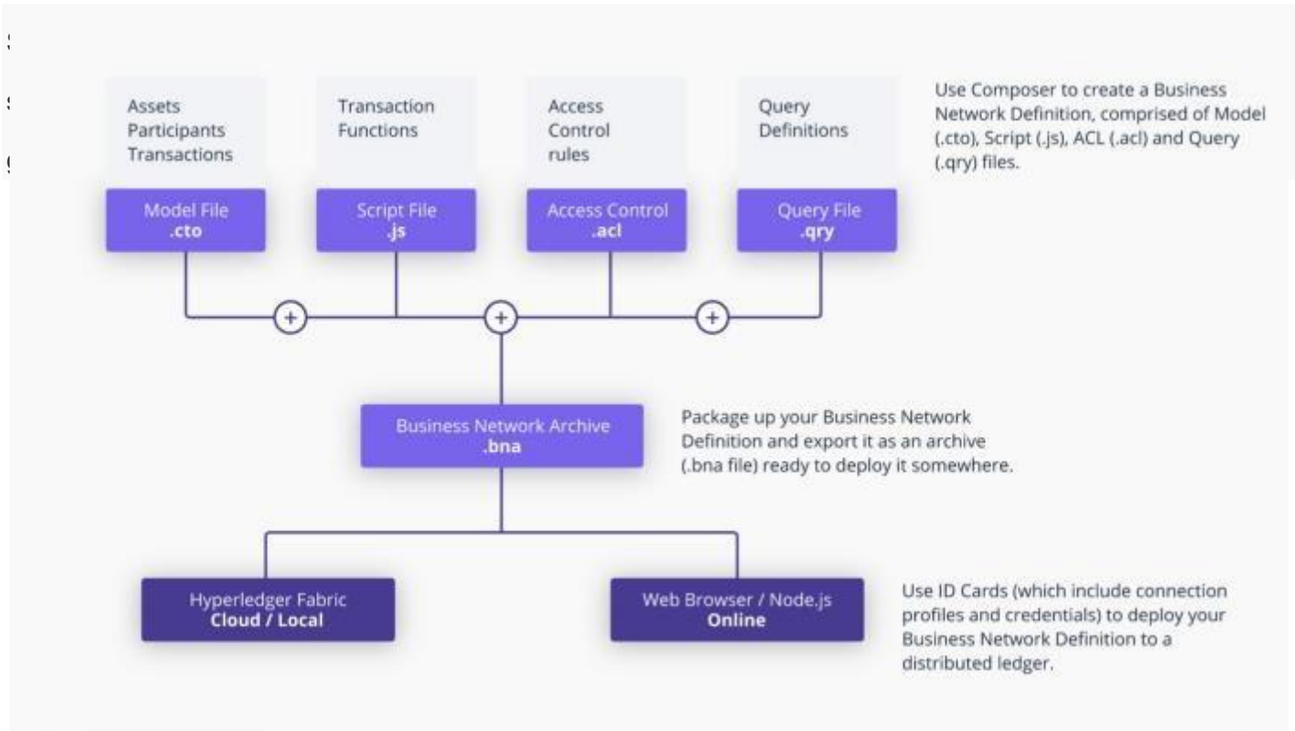


Submit Transaction Dialog

```
{
 "$class": "com.kpbird.Allocate",
 "hardware": "resource:com.kpbird.Hardware#MAC01",
 "newOwner": "resource:com.kpbird.Employee#02"
}
```

Now, We are allocating Mac01 to Employee 02. Click Submit button after update above JSON in Transaction Dialog. As soon as you hit submit button. Transaction processed and Transaction Id will generate.





source: https://hyperledger.github.io/composer/introduction/introduction

Step 14: Start Docker and run following commands from ~/fabric-tools directory

Install business network to Hyperledger Fabric, If business network is already installed you can use “update”

Step 13: Now, It’s time to deploy “hardware-assets” business network to Hyperledger Fabric. Click on “Define” tab from top panel and click “Export” button from left panel. Export will create hardware-assets.bna file.

Following command will deploy and start hardware-assets.bna file. Change hardware-assets.bna file before you execute following command. networkadmin.card file will generate in ~/fabric-tools directory from previous command.

```
$composer network start – card PeerAdmin@hlfv1 – networkAdmin admin – networkAdminEnrollSecret adminpw – archiveFile /Users/ketan/Downloads/hardware-assets.bna – file networkadmin.card
```

```
Ketan-Parmar:fabric-tools ketan$ composer network start --card PeerAdmin@hlfv1 --networkAdmin admin --networkAdminEnrollSecr
et adminpw --archiveFile /Users/ketan/Downloads/hardware-assets.bna --file networkadmin.card
Starting business network from archive: /Users/ketan/Downloads/hardware-assets.bna
Business network definition:
        Identifier: hardware-assets@0.0.1
        Description: Hardware Assets will maintain Software company's hardware

Processing these Network Admins:
        userName: admin

✔ Starting business network definition. This may take a minute...
Successfully created business network card:
        Filename: networkadmin.card

Command succeeded
```

Download hardware-assets.bna file

To connect business network you need connection card, so we can import networkadmin card using following command

.bna is Business Network Archive file which contains model, script, network access and query file

```
$composer card import -f networkadmin.card
```

To make sure networkadmin.card successfully install you can list cards using following command

```
$composer card list
```

```
Ketan-Parmar:fabric-tools ketan$ composer card list
The following Business Network Cards are available:

Connection Profile: hlfv1
```

Card Name	UserId	Business Network
admin@hardware-assets	admin	hardware-assets
PeerAdmin@trade-network	PeerAdmin	trade-network
admin@trade-network	admin	trade-network
PeerAdmin@hlfv1	PeerAdmin	

```
Issue composer card list --name <Card Name> to get details a specific card

Command succeeded
```

Following command will make sure that our hardware-assets business network is successfully running in Hyperledger Fabric.

$composer network ping – card admin@hardware-assets

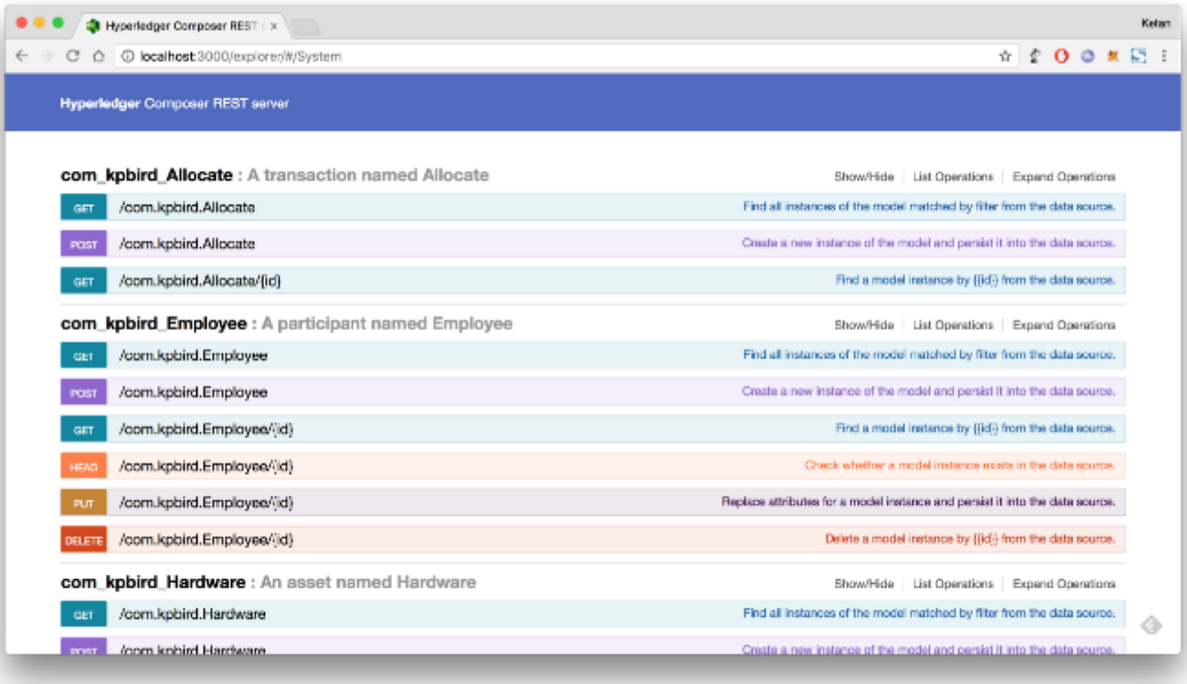
```
Ketan-Parmar:fabric-tools ketan$ composer network ping --card admin@hardware-assets
The connection to the network was successfully tested: hardware-assets
        version: 0.16.0
        participant: org.hyperledger.composer.system.NetworkAdmin#admin

Command succeeded
```

Now It's time to interact with REST API. To develop Web or Mobile Application we require REST API. you can run following command to generate REST API for hardware-assets business network.

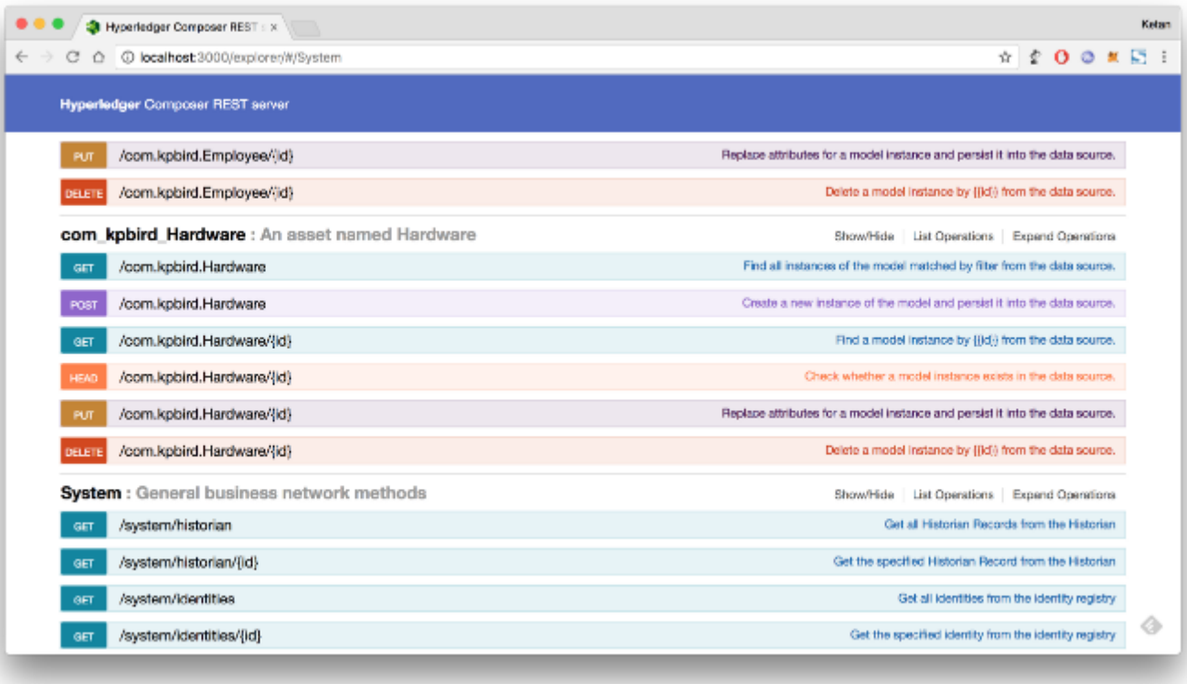
$composer-rest-server

```
Ketan-Parmar:fabric-tools ketan$ composer-rest-server
? Enter the name of the business network card to use: admin@hardware-assets
? Specify if you want namespaces in the generated REST API: always use namespaces
? Specify if you want to enable authentication for the REST API using Passport: No
? Specify if you want to enable event publication over WebSockets: Yes
? Specify if you want to enable TLS security for the REST API: No

To restart the REST server using the same options, issue the following command:
   composer-rest-server -c admin@hardware-assets -n always -w true

Discovering types from business network definition ...
Discovered types from business network definition
Generating schemas for all types in business network definition ...
Generated schemas for all types in business network definition
Adding schemas for all types to Loopback ...
Added schemas for all types to Loopback
Web server listening at: http://localhost:3000
Browse your REST API at http://localhost:3000/explorer
```

rest server will ask few basic information before generate rest api



REST API for our hardware assets



REST API methods for all operations

Conclusion: In this way we have learnt about hyperledger and its use case in business world.
