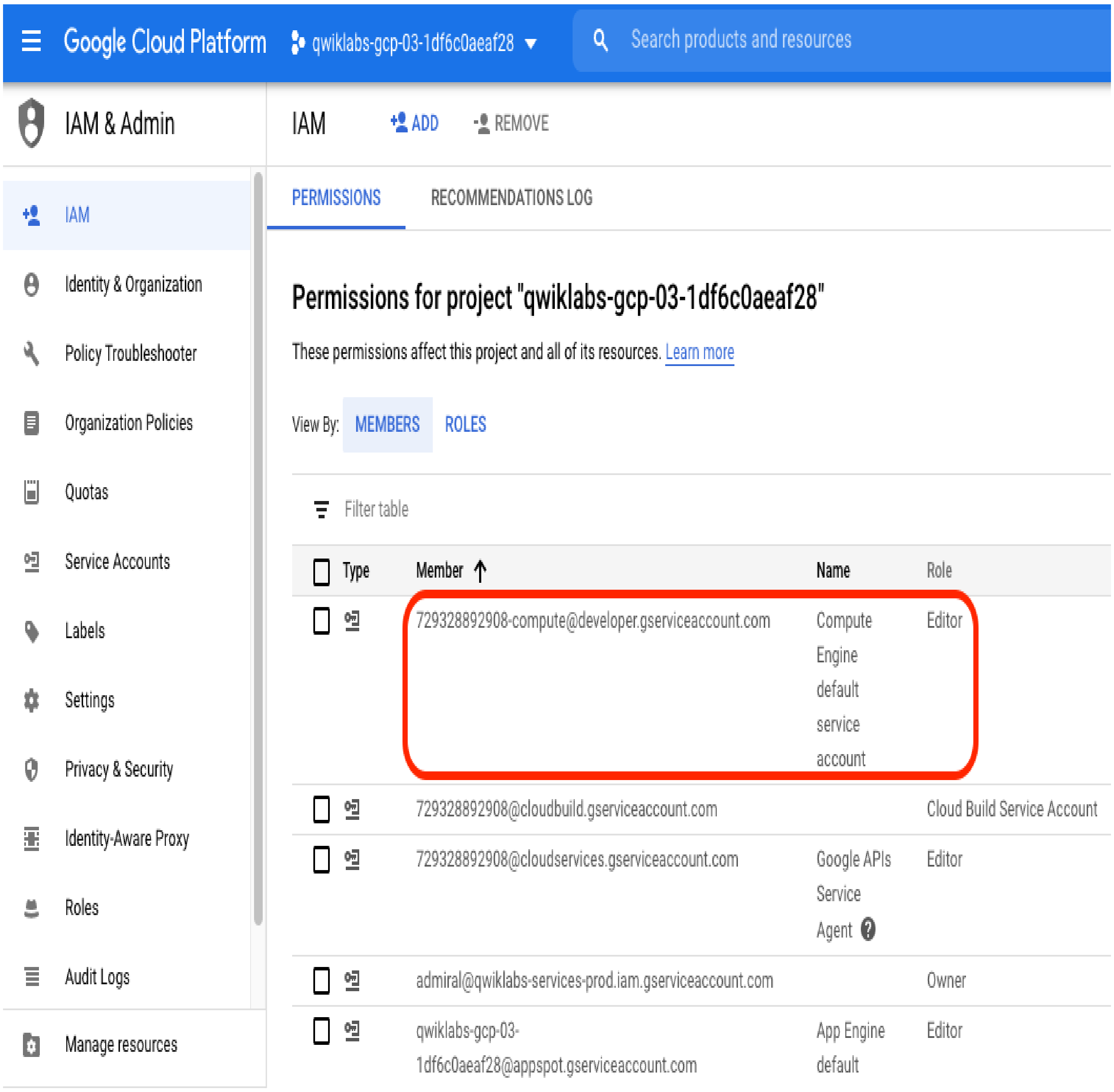
Check project permissions

Before you begin your work on Google Cloud, you need to ensure that your project has the correct permissions within Identity and Access Management (IAM).

1. In the Google Cloud console, on the **Navigation menu** (☰), click **IAM & Admin** > **IAM**.

2. Confirm that the default compute Service Account {project-number}-compute@developer.gserviceaccount.com is present and has the editor role assigned. The account prefix is the project number, which you can find on **Navigation menu** > **Home**.



Permissions for project "qwiklabs-gcp-03-1df6c0aeaf28"

These permissions affect this project and all of its resources. Learn more

Type	Member	Name	Role
	729328892908-compute@developer.gserviceaccount.com	Compute Engine default service account	Editor
	729328892908@cloudbuild.gserviceaccount.com		Cloud Build Service Account
	729328892908@cloudservices.gserviceaccount.com	Google APIs Service Agent	Editor
	admiral@qwiklabs-services-prod.iam.gserviceaccount.com		Owner
	qwiklabs-gcp-03-1df6c0aeaf28@appspot.gserviceaccount.com	App Engine default	Editor

If the account is not present in IAM or does not have the editor role, follow the steps below to assign the required role.

•In the Google Cloud console, on the **Navigation menu**, click **Home**.

•Copy the project number (e.g. 729328892908).

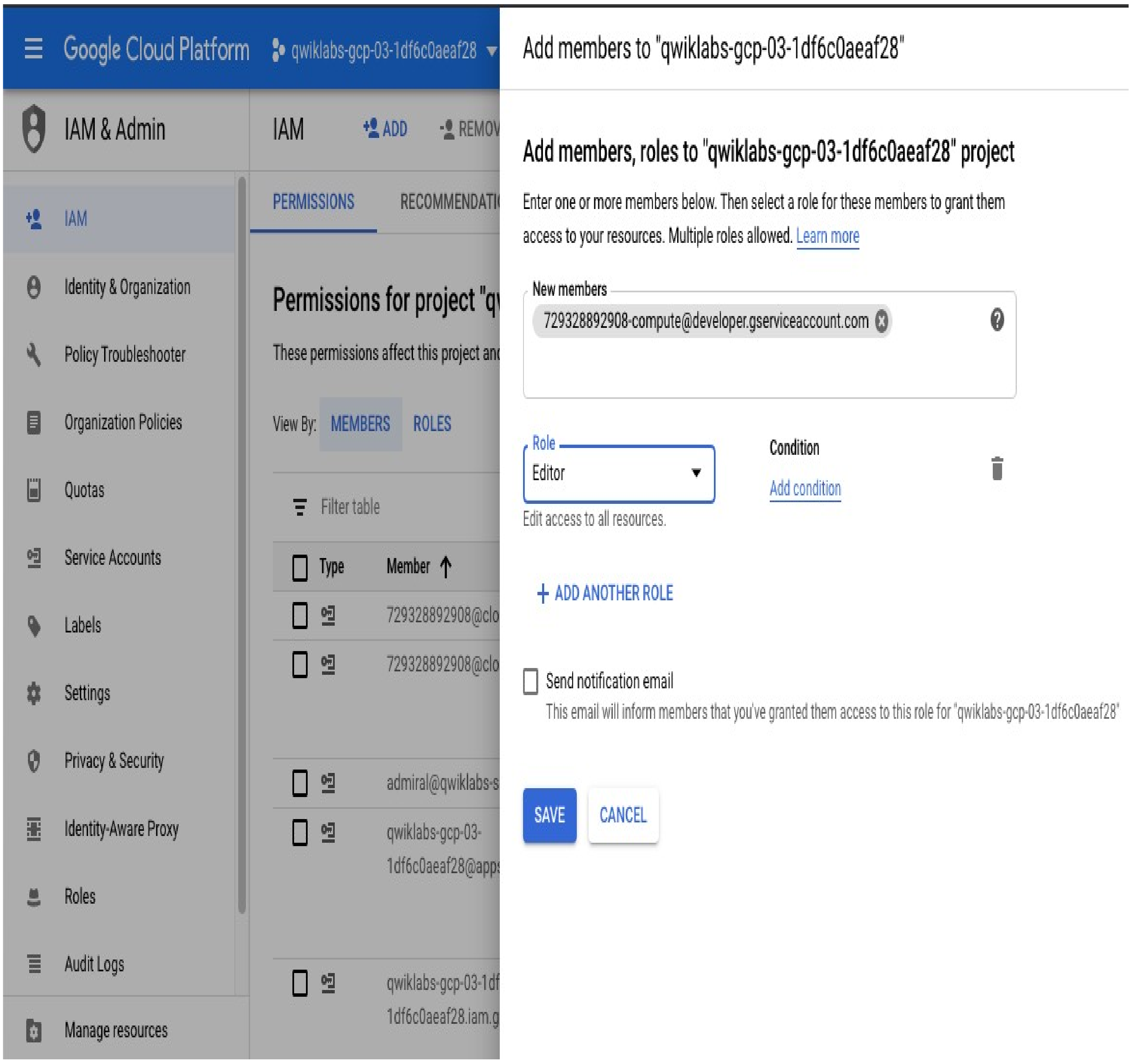
•On the **Navigation menu**, click **IAM & Admin** > **IAM**.

•At the top of the **IAM** page, click **Add**.

•For **New members**, type:

```
{project-number}-compute@developer.gserviceaccount.com
content_copy
```
Replace {project-number} with your project number.

•For **Role**, select **Project** (or Basic) > **Editor**. Click **Save**.


Add members to "qwiklabs-gcp-03-1df6c0aeaf28"

Add members, roles to "qwiklabs-gcp-03-1df6c0aeaf28" project

Enter one or more members below. Then select a role for these members to grant them access to your resources. Multiple roles allowed. Learn more

New members: 729328892908-compute@developer.gserviceaccount.com

Role: Editor

Edit access to all resources.

Condition: Add condition

+ ADD ANOTHER ROLE

Send notification email

This email will inform members that you've granted them access to this role for "qwiklabs-gcp-03-1df6c0aeaf28"

SAVE CANCEL

Launch Hyperledger Fabric and Composer

In this section, you will launch the Hyperledger Fabric and Composer solution available on Google Cloud Marketplace.

1.From the **Navigation menu**, select **Marketplace**.

2.Type **Hyperledger** in the search box then select **Hyperledger Fabric and Composer**.

3.Click **LAUNCH**.

You are now presented with the deployment properties setup.

4.For **Deployment name**, type hyperledger-fabric-and-composer.

5.Check the box to accept the Terms of Service.

6.Keep the rest of the settings at their defaults and click **Deploy**.

This solution will be deployed by **Deployment Manager** and you can watch the progress on the Deployment Manager screen, which is the screen you're on.

7.Once the solution is deployed, from the **Navigation menu**, select **Compute Engine**.
You should see a VM instance that was created as part of the deployment.

8.Click on the instance name to open the **VM instance details** page.

9.Click **EDIT** at the top.

10.Scroll down to **Network tags** and enter hyperledgerin the field.

This will allow you to configure specific firewall rules on instances which have been tagged with the Network tag - hyperledger. You will configure this in the next section.

11.Click **Save**.

Enable firewall access on the VM

To access Playground, you will need to enable port 8080 on the deployed VM. For security purposes, you will create a firewall rule that allows access to port 8080 only from your browser IP.

1.Open a new browser tab and navigate to http://www.whatismyip.com.

2.Copy the public IPv4 address.

3.In the Console, go to **Navigation menu** > **VPC network** > **Firewall**.

4.Click **CREATE FIREWALL RULE**.

5.Name for the firewall rule hyperleger-firewall-rule and add an optional description.

6.Under **Targets**, ensure that the **Specified target tags** option is selected.

7.For **Target tags**, enter the **Network tag** name that you created in the previous section, hyperledger.

8.Under **Source filter**, select **IP ranges**.

9.For **Source IP ranges**, enter your public IPv4 address with the mask /32. For example, if your public IPv4 address is 104.132.25.98, enter the range as 104.132.25.98/32.

10.Under **Protocols and ports**, select the **Specified protocols and ports** option.

a. Check the tcp checkbox

b. Enter the port number in the tcp box as 8080

Note: This is the port number on which Hyperledger Composer will be listening by default.

11.Click **Create**. Wait for the firewall rule to be created before moving on.

Deploy and access Hyperledger Composer Playground

1.From the **Navigation menu**, select **Compute Engine**.

2.Click the **SSH** button for the Hyperledger VM.

3.Inside the SSH session, run the following command to run as root:

```
sudo su
content_copy
```

4.Deploy Composer Playground by running:

```
composer-playground
content_copy
```

This will deploy the Playground solution and run it on port 8080 by default.

5. Go back to the Console and copy the **External IP** of the Hyperledger VM.

6. Open a new browser tab and navigate to http://<ExternalIP>:8080 to access Hyperledger Composer Playground.

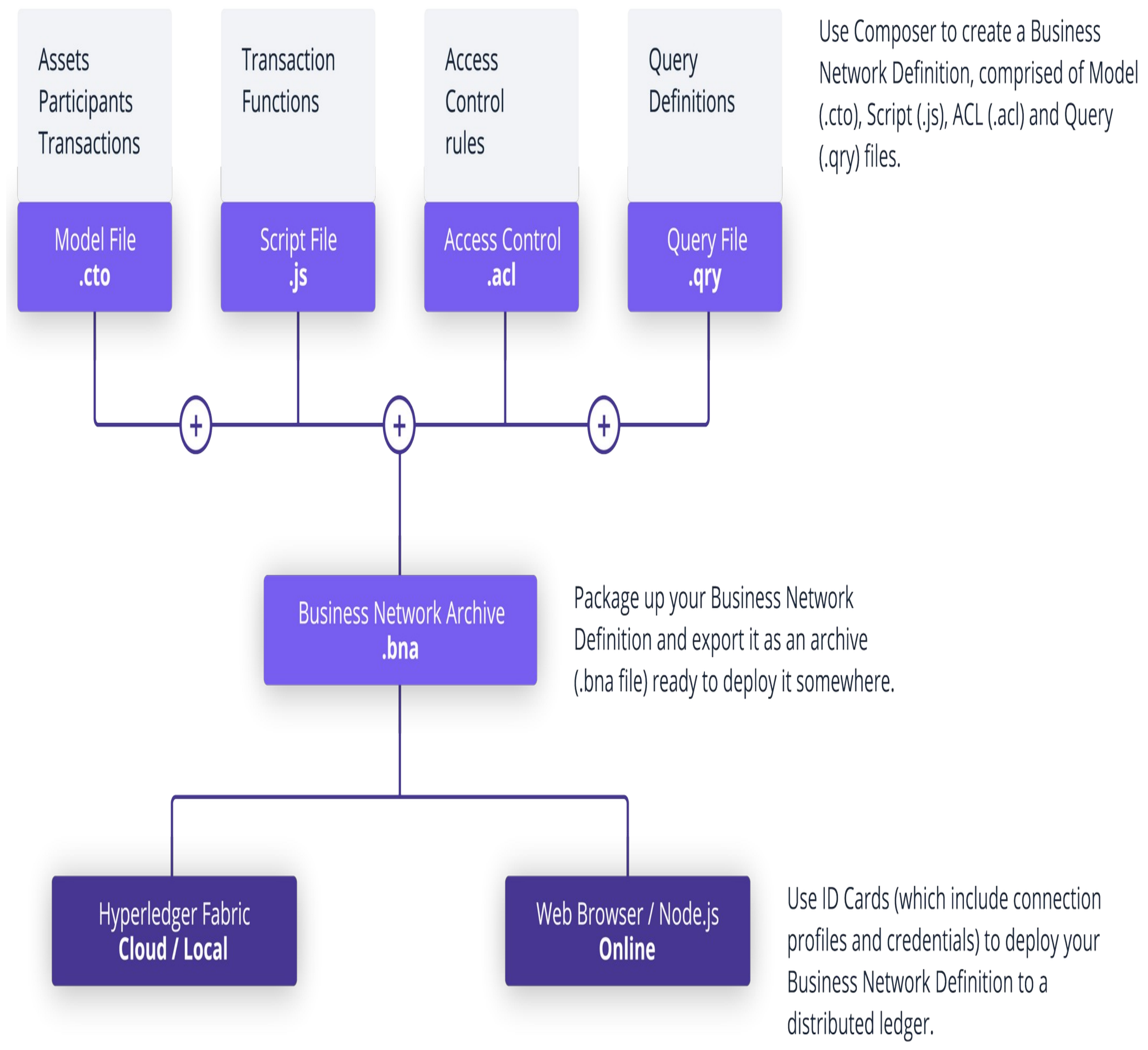
7. Click the **Let's Blockchain!** button to get started.

Using Playground

In this section, you will set up a business network, defining assets, participants and transactions, and testing the network by creating some participants, an asset, and submitting transactions to change the ownership of the asset from one to another.

Create and connect to a new business network

A business network has a couple of defining properties; a name, and an optional description. You can also choose to base a new business network on an existing template, or import your own template. This represents the different components of the business network:


Assets
Participants
Transactions
Transaction
Functions
Access
Control
rules
Query
Definitions
Use Composer to create a Business Network Definition, comprised of Model (.cto), Script (.js), ACL (.acl) and Query (.qry) files.
Model File
.cto
Script File
.js
Access Control
.acl
Query File
.qry
Business Network Archive
.bna
Package up your Business Network Definition and export it as an archive (.bna file) ready to deploy it somewhere.
Hyperledger Fabric
Cloud / Local
Web Browser / Node.js
Online
Use ID Cards (which include connection profiles and credentials) to deploy your Business Network Definition to a distributed ledger.

For typical solution architecture, refer to this link.

1.Click **Deploy a new business network** under the **Web Browser** heading to get started.

Note: Do not click the **Deploy a new business network** under the heading **Connection: hlfv1**.

2.In Basic Information, name the network tutorial-network and optionally add a description.

3.In Model Network Starter Template, select the **empty-business-network** card. You will be building a business network from scratch.

4.Keep the rest of the default settings and click **Deploy**.

You should see a new business network card called **admin** for the business network tutorial-network in your wallet. The wallet can contain business network cards to connect to multiple deployed business networks.

When connecting to an external blockchain, business network cards represent everything necessary to connect to a business network. They include connection details, authentication material, and metadata.

5.Now that the business network is deployed, click **Connect now** on tutorial-network.

Add a model file

You are now on the **Define** tab now. This is where you create and edit files that make up the business network definition.

Since you selected an empty business network template, you need to modify the default template files provided. The first step is to update the Model File. Model files define the assets, participants, transactions, and events in your business network.

1.Click **Model File** to view it.

2.Delete the existing lines of code in the Model File and replace it with this:

```
/**
  * My commodity trading network
  */
namespace org.example.mynetwork
asset Commodity identified by tradingSymbol {
        o String tradingSymbol
        o String description
        o String mainExchange
        o Double quantity
        --> Trader owner
}
participant Trader identified by tradeId {
        o String tradeId
        o String firstName
        o String lastName
}
transaction Trade {
        --> Commodity commodity
        --> Trader newOwner
}
content_copy
```

This domain model defines a single asset type Commodity and single participant type Trader and a single transaction type Trade that is used to modify the owner of a commodity.

Add a transaction processor script file

You can now define the transaction logic for the business network. Composer expresses the logic for a business network using JavaScript functions. These functions are automatically executed when a transaction is submitted for processing.

1.In the bottom left, click the **Add a file** link.

2.Click the **Script File** radio button, then **Add**.

3.Delete the lines of code in the script file and replace it with the following code:

```
/**
  * Track the trade of a commodity from one trader to another
  * @param {org.example.mynetwork.Trade} trade - the trade to be processed
  * @transaction
  */
async function tradeCommodity(trade) {
        trade.commodity.owner = trade.newOwner;
        let assetRegistry = await getAssetRegistry('org.example.mynetwork.Commodity');
        await assetRegistry.update(trade.commodity);
}
content_copy
```

This function changes the owner property on a commodity based on the newOwner property on an incoming Trade transaction. It then persists the modified Commodity back into the asset registry, used to store Commodity instances.

Deploy the updated business network

Now that you have the model, script, and access control files, you can deploy and test your business network.

Click **Deploy changes** to upgrade the business network.

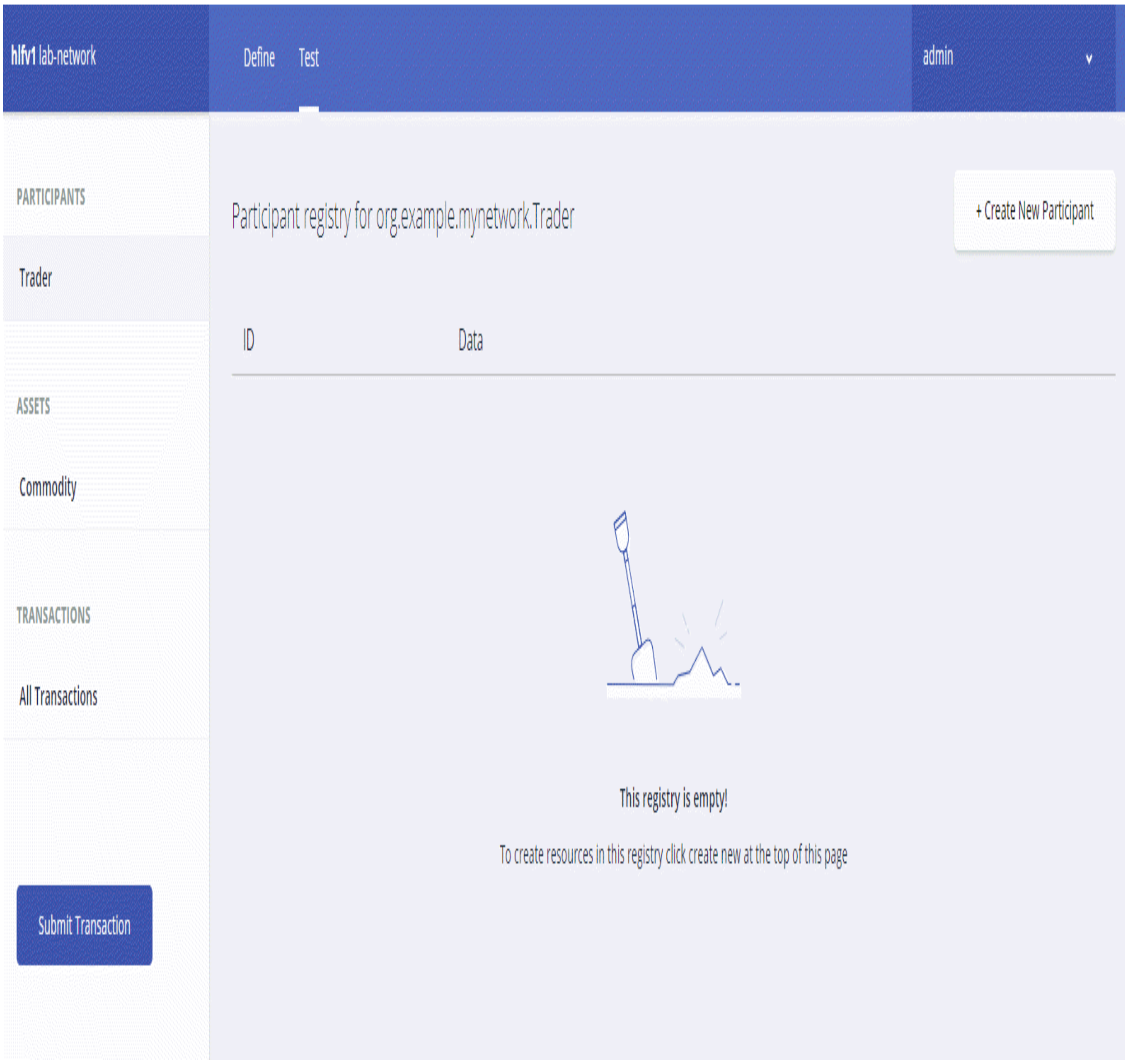
Test the business network definition

Next, test your business network by creating some participants (in this case Traders), creating an asset (a Commodity), and then using your Trade transaction to change the ownership of the Commodity.

Click the **Test** tab to get started.

Create participants

1. Ensure that you have the **Trader** tab selected on the left, and click **Create New Participant** in the upper right.



Participant registry for org.example.mynetwork.Trader

This registry is empty!

To create resources in this registry click create new at the top of this page

2.What you can see is the data structure of a Trader participant. We want some easily recognizable data, so delete the code that's there and paste the following:

```
{
   "$class": "org.example.mynetwork.Trader",
   "tradeId": "TRADER1",
   "firstName": "Jenny",
   "lastName": "Jones"
}
```

content_copy

3.Click **Create New** to create the participant.

4.You should be able to see the new Trader participant you created. You'll need another Trader to test your Trade transaction though, so create another Trader, but this time, use the following data:

```
{
   "$class": "org.example.mynetwork.Trader",
   "tradeId": "TRADER2",
   "firstName": "Amy",
   "lastName": "Williams"
}
```
content_copy

Make sure that both participants exist in the Trader view before moving on.

Create an asset

Now that you have two Trader participants, you need something for them to trade. Creating an asset is very similar to creating a participant. The Commodity you are creating will have an owner property indicating that it belongs to the Trader with the tradeId of TRADER1.

1.Click **Commodity** under Assets and click **Create New Asset**.

2.Delete the asset data and replace it with the following:

```
{
   "$class": "org.example.mynetwork.Commodity",
   "tradingSymbol": "ABC",
   "description": "Test commodity",
   "mainExchange": "Euronext",
   "quantity": 72.297,
   "owner": "resource:org.example.mynetwork.Trader#TRADER1"
}
```
content_copy

3.After creating this asset, you should be able to see it in the Commodity page.

Transfer commodity between participants

With two Traders and a Commodity to trade between them you can test your Trade transaction. Transactions are the basis of all changes in a Hyperledger Composer business network.

1.Click the **Submit Transaction** button on the bottom left.

2.Ensure that the transaction type is Trade.

3.Replace the transaction data with the following, or just change the details:

```
{
   "$class": "org.example.mynetwork.Trade",
   "commodity": "resource:org.example.mynetwork.Commodity#ABC",
   "newOwner": "resource:org.example.mynetwork.Trader#TRADER2"
}
content_copy
```

4.Click **Submit**.

5.Check that the asset has changed ownership from TRADER1 to TRADER2 by expanding the data section for the asset. You should see that the owner is listed as: org.example.mynetwork.Trader#TRADER2.

To view the full transaction history of the business network, click **All Transactions** on the left.

You will get a list of each transaction as they were submitted. You can see that certain actions performed using the UI, like creating the Trader participants and the Commodity asset, are recorded as transactions even though they are not defined as transactions in the business network model. These transactions are known as **System Transactions** and are common to all business networks and defined in the Hyperledger Composer Runtime.

In this lab, you created a business network from scratch and defined a model file inside it. Alternatively, you could deploy one of the sample networks available from Hyperledger.

Integrating with existing applications

Hyperledger Composer can be integrated with existing systems by using a Loopback API. Integrating existing systems allows you to pull data from existing business systems and convert it to assets or participants in a Composer business network. Refer to this link for more details.

Congratulations!

You have successfully created, deployed, and tested a business network using Hyperledger Composer Playground.

Next Steps / Learn More

- For more information on Hyperledger Composer, visit the site.
- For more information on writing transaction processor functions, read the documentation.

•For more information on the modeling language, read the documentation.

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated October 13, 2020

Lab Last Tested October 13, 2020