Project: Food Regulation

# Study Project: CS F211

Blockchain Architecture and Use Cases

2016A7PS0315U

**Topic:**

Food supplier verification using Blockchain

**Aim:**

Build a blockchain network that automates a smart contract

Summary:

Using Hyperledger Composer, we shall model the requirements for a verification onto a smart contract and deploy it as a distributed business network

Purpose of the topic:

Regulation using fabricated applications that automate processes are highly demanding. Blockchain presents a chance to do only that. We'll be using Hyperledger Composer to effectively deploy application on JavaScript and build the logic in a smart contract that is loaded on a business network. This project centers around a food supplier verification use case.

Let's say that a provider will exchange the food items to a merchant who checks that the provider, nation, and food type all match the right identifiers. At the port of entry, the provider is again checked against a list of known suppliers in a database (overseen by the regulator). If the provider is of sort excluded, at that point items exchange to the retailer. On the off chance that the provider is non-absolved, the items are checked against a rundown of known nourishment items in the database (overseen by the controller). In the event that the food is an excluded item, at that point it exchanges to the retailer. If the food is non-accompliant, the merchant must direct an inspection.

We will be deploying a distributed network, remembering that a business network files comprises of model (.cto), content (.js), and (.acl) records bundled as an archive bna. The file is then exported to Hyperledger Fabric Network

**Walkthrough to start:**

* Install Hyperledger Composer development tools.
* Configure and start Hyperledger Fabric network.
* Generate the Business Network Archive file.
* Import the BNA file into Hyperledger Composer Playground.

**Included Components:**

* Hyperledger Fabric
* Hyperledger Composer

**Included technologies:**

* Blockchain
* Containers
* Cloud

**Detailed walkthrough of setup:**

Start the hyperledger fabric network:

Start the Fabric and create a PeerAdmin card using the following commands:

*./downloadFabric.sh*

*./startFabric.sh*

*./createPeerAdminCard.sh*

Generate the business network archive (.bna):

Head back to the repository of the project and enter *npm install*

**Output:**

*Creating Business Network Archive*

*Looking for package.json of Business Network Definition*

*Input directory: /Users/ishan/Documents/git-demo/BlockchainPublicRegulationFabric-Food*

*Found:*

*Description: Sample food supplier verification network*

*Name: food-supply*

*Identifier: food-supply@0.0.1*

*Written Business Network Definition Archive file to*

*Output file: ./dist/food-supply.bna*

*Command succeeded*

**Deploy on composer playground and initiate the chaincode:**

Once the business network archive is ready (.bna), we have two options: either use Composer Playground to deploy the business network and leverage it, or localhost it using SwaggerUI and let RESTful APIs handle the connection between the AngularJS front end and the composer blockchain server.

We shall opt for the latter. Navigating to the dist folder of the project,

*cd dist*

*composer network install --card PeerAdmin@hlfv1 --archiveFile food-supply.bna*

*composer network start --networkName food-supply --networkVersion 0.0.1 --networkAdmin admin --networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file networkadmin.card*

*composer card import --file networkadmin.card*

Now we have installed a network card for the admin of the given business network. We shall next start the server.

## *cd ..*

## *composer-rest-server*

Answer the questions posed at startup. These allow the composer-rest-server to connect to Hyperledger Fabric and configure how the REST API is generated.

* *Enter admin@food-supply as the card name.*
* *Select never use namespaces when asked whether to use namespaces in the generated API.*
* *Select No when asked whether to use an API key to secure the generated API.*
* *Select No when asked whether to enable authentication for the REST API using Passport.*
* *Select Yes when asked whether to enable event publication.*
* *Select No when asked whether to enable TLS security.*