

Car-Auction On Blockchain Using HyperledgerFabric

In this project one can list assets for sale (setting a reserve price), and watch as assets that have met their reserve price are automatically transferred to the highest bidder at the end of the auction.

- **Project Link**

- <https://github.com/manu461/ProjectsDemo/tree/master/BlockChainProjects/HyperLedgerComposerFabricProject/carauction-network>
- <https://github.com/manu461/ProjectsDemo/tree/master/BlockChainProjects/HyperLedgerComposerFabricProject/carauction-network/carauction-angular-app>

- **Project Description**

I created this project while completing **IBM Blockchain-Foundation Developer** course.

Car-Auction's (this project's) Business-network is developed using **Hyperledger Composer**, whereas the network been deployed locally using **Hyperledger Fabric**.

The complete package (**.bna**) containing **Business-Model** (.cto file), **script files** (.js file), **ACLs** (.acl file) and **Metadata** (.md file) is converted to an **Angular Application** using **Yeoman generator** (YO code generator).

In this project one can list assets for sale (setting a reserve price), and watch as assets that have met their reserve price are automatically transferred to the highest bidder at the end of the auction.

This business network defines:

1. Participants
 - a) Member
 - b) Auctioneer
2. Assets
 - a) Vehicle
 - b) VehicleListing

3. Transactions:
 - a) Offer
 - b) CloseBidding

The **makeOffer** function is called when an **Offer** transaction is submitted. The logic simply checks that the listing for the offer is still for sale, and then adds the offer to the listing, and then updates the offers in the **VehicleListing** asset registry.

The **closeBidding** function is called when a **CloseBidding** transaction is submitted for processing. The logic checks that the listing is still for sale, sorts the offers by bid price, and then if the reserve has been met, transfers the ownership of the vehicle associated with the listing to the highest bidder. Money is transferred from the buyer's account to the seller's account, and then all the modified assets are updated in their respective registries.

****NOTE:** This is not my idea to develop such project, it is already provided as a template for business-network by IBM at their composer-playground website. I have studied the code and logics thoroughly and implemented it during completing IBM Blockchain-Foundation Developer course.

- **Tools Used**

1. Data-Modelling is done with **composer modelling language**. It is domain specific language for describing the nature of the business network. It is developed by IBM.
2. Business-logic is written in **JavaScript**.
3. **Composer-Playground** is used to do ideation and very quickly prototype composer solution.
4. Two main **npm** modules
 - a) Composer-client
 - b) Composer-Adminwhich can be embedded in application for programmatic access.
5. **VSCode** is used as code-editor.
6. **CLI tools**
7. Complete package of Business network Archive (.bna) is converted into an Angular-Application using **Yeoman generator**(YO code generator).

- **Project Walkthrough**

A. Starting the Application

1. Start Fabric

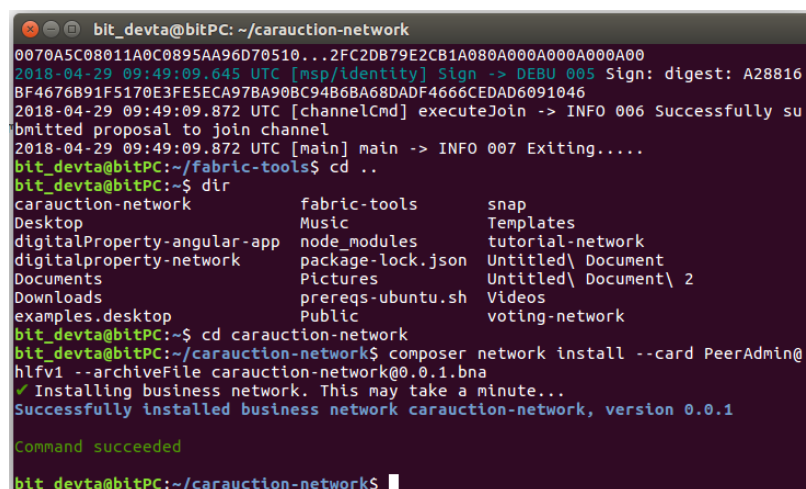
- Press **ctrl+alt+t** to open terminal.
- Change directory to fabric-tools by typing **cd fabric-tools** on the terminal.
- Then type this code on the terminal **./startFabric.sh**

2. Start Business Network

- Type **cd ..** and press enter on terminal to exit fabric-tools directory.
- Change directory to caraucation-network by typing **cd caraucation-network** on the terminal.
- Install the business network.

From the caraucation-network directory, run the following command,

composer network install --card PeerAdmin@hlfv1 --archiveFile caraucation-network@0.0.1.bna



```
bit_devta@bitPC: ~/caraucation-network
0070A5C08011A0C0895AA96D70510...2FC2DB79E2CB1A080A000A000A000A00
2018-04-29 09:49:09.645 UTC [msp/identity] Sign -> DEBU 005 Sign: digest: A28816
BF4676B91F5170E3FE5ECA978A90BC94B6BA68DADF4666CEDAD6091046
2018-04-29 09:49:09.872 UTC [channelCmd] executeJoin -> INFO 006 Successfully su
bmitted proposal to join channel
2018-04-29 09:49:09.872 UTC [main] main -> INFO 007 Exiting....
bit_devta@bitPC:~/fabric-tools$ cd ..
bit_devta@bitPC:~$ dir
caraucation-network      fabric-tools             snap
Desktop                  Music                    Templates
digitalProperty-angular-app  node_modules            tutorial-network
digitalproperty-network    package-lock.json       Untitled\ Document
Documents                 Pictures                 Untitled\ Document\ 2
Downloads                 prereqs-ubuntu.sh        Videos
examples.desktop          Public                   voting-network
bit_devta@bitPC:~$ cd caraucation-network
bit_devta@bitPC:~/caraucation-network$ composer network install --card PeerAdmin@
hlfv1 --archiveFile caraucation-network@0.0.1.bna
✓ Installing business network. This may take a minute...
Successfully installed business network caraucation-network, version 0.0.1

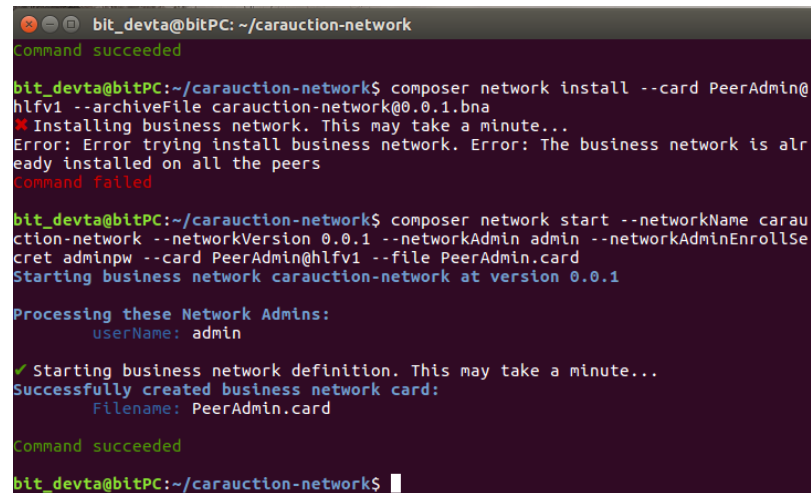
Command succeeded
bit_devta@bitPC:~/caraucation-network$
```

Figure 1. Install the business network

iv. Start the business network

Do not change the directory and run the following command,

```
composer network start --networkName carauktion-network --  
networkVersion 0.0.1 --networkAdmin admin --  
networkAdminEnrollSecret adminpw --card PeerAdmin@hlfv1 --file  
PeerAdmin.card
```



```
bit_devta@bitPC: ~/carauktion-network
Command succeeded

bit_devta@bitPC:~/carauktion-network$ composer network install --card PeerAdmin@
hlfv1 --archiveFile carauktion-network@0.0.1.bna
✖ Installing business network. This may take a minute...
Error: Error trying install business network. Error: The business network is alr
eady installed on all the peers
Command failed

bit_devta@bitPC:~/carauktion-network$ composer network start --networkName carau
ktion-network --networkVersion 0.0.1 --networkAdmin admin --networkAdminEnrollSe
cret adminpw --card PeerAdmin@hlfv1 --file PeerAdmin.card
Starting business network carauktion-network at version 0.0.1

Processing these Network Admins:
  userName: admin

✔ Starting business network definition. This may take a minute...
Successfully created business network card:
  filename: PeerAdmin.card

Command succeeded

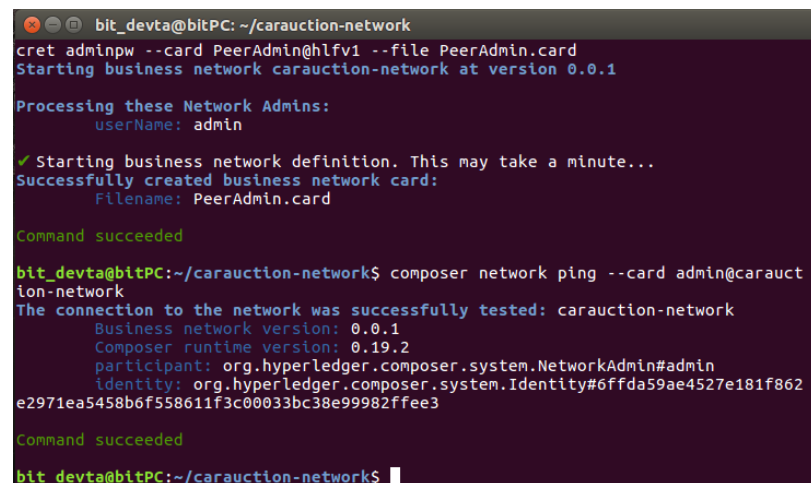
bit_devta@bitPC:~/carauktion-network$
```

Figure 2. Start the business network

v. Check Ping

To check that the business network has been deployed successfully, run the following command to ping the network.

```
composer network ping --card admin@carauktion-network
```



```
bit_devta@bitPC: ~/carauktion-network
cret adminpw --card PeerAdmin@hlfv1 --file PeerAdmin.card
Starting business network carauktion-network at version 0.0.1

Processing these Network Admins:
  userName: admin

✔ Starting business network definition. This may take a minute...
Successfully created business network card:
  filename: PeerAdmin.card

Command succeeded

bit_devta@bitPC:~/carauktion-network$ composer network ping --card admin@carauk
tion-network
The connection to the network was successfully tested: carauktion-network
  Business network version: 0.0.1
  Composer runtime version: 0.19.2
  participant: org.hyperledger.composer.system.NetworkAdmin#admin
  identity: org.hyperledger.composer.system.Identity#6ffda59ae4527e181f862
e2971ea5458b6f558611f3c00033bc38e99982ffee3

Command succeeded

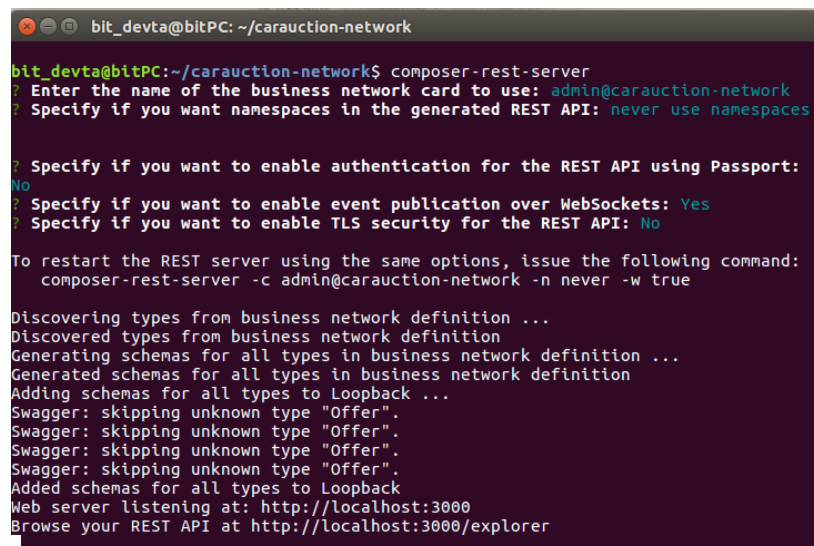
bit_devta@bitPC:~/carauktion-network$
```

Figure 3. Ping check the Business network

3. Start Rest API

Type **composer-rest-server** on the terminal, and then,

- i. Enter **admin@carauction-network** as business network card to use.
- ii. Select **never use namespaces**
- iii. Press **n** to specify if you want to enable authentication for REST API.
- iv. Press **y** to specify if you want to enable event publication.
- v. Press **n** to specify if you want to enable TLS security.



```
bit_devta@bitPC: ~/carauction-network
bit_devta@bitPC:~/carauction-network$ composer-rest-server
? Enter the name of the business network card to use: admin@carauction-network
? Specify if you want namespaces in the generated REST API: never 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@carauction-network -n never -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 ...
Swagger: skipping unknown type "Offer".
Swagger: skipping unknown type "Offer".
Swagger: skipping unknown type "Offer".
Swagger: skipping unknown type "Offer".
Added schemas for all types to Loopback
Web server listening at: http://localhost:3000
Browse your REST API at http://localhost:3000/explorer
```

Figure 4. Start the rest API

4. Start the Angular-App

- i. Open a new terminal using **ctrl+alt+t**
- ii. Change the directory using **cd carauction-network/carauction-angular-app**
- iii. Enter the following command to start the app **npm start**

```
bit_devta@bitPC: ~/carauction-network/carauction-angular-app
69% building modules 854/860 modules 6 active ...s/observable/SubscribeOnObserv
69% building modules 855/860 modules 5 active ...s/observable/SubscribeOnObserv
69% building modules 856/860 modules 4 active ...s/observable/SubscribeOnObserv
69% building modules 856/861 modules 5 active ...-app/node_modules/rxjs/util/as
69% building modules 857/861 modules 4 active ...-app/node_modules/rxjs/util/as
69% building modules 858/861 modules 3 active ...-app/node_modules/rxjs/util/as
69% building modules 859/861 modules 2 active ...-app/node_modules/rxjs/util/as
69% building modules 859/862 modules 3 active ...-angular-app/node_modules/ms/i
69% building modules 860/862 modules 2 active ...-app/node_modules/rxjs/util/as
69% building modules 861/862 modules 1 active ...-app/node_modules/rxjs/util/as
Hash: 9c069cf3ac401fc2ea99
Time: 17313ms
chunk {0} polyfills.bundle.js, polyfills.bundle.js.map (polyfills) 270 kB {5}
[initial] [rendered]
chunk {1} main.bundle.js, main.bundle.js.map (main) 183 kB {4} [initial] [ren
dered]
chunk {2} styles.bundle.js, styles.bundle.js.map (styles) 184 kB {5} [initial
] [rendered]
chunk {3} scripts.bundle.js, scripts.bundle.js.map (scripts) 439 kB {5} [init
ial] [rendered]
chunk {4} vendor.bundle.js, vendor.bundle.js.map (vendor) 4.12 MB [initial] [
rendered]
chunk {5} inline.bundle.js, inline.bundle.js.map (inline) 0 bytes [entry] [re
ndered]
webpack: Compiled successfully.
```

Figure 5. Start the angular-app

- iv. Angular-App is now started, open <http://localhost:4200>

B. Application Walkthrough

1. Home Page

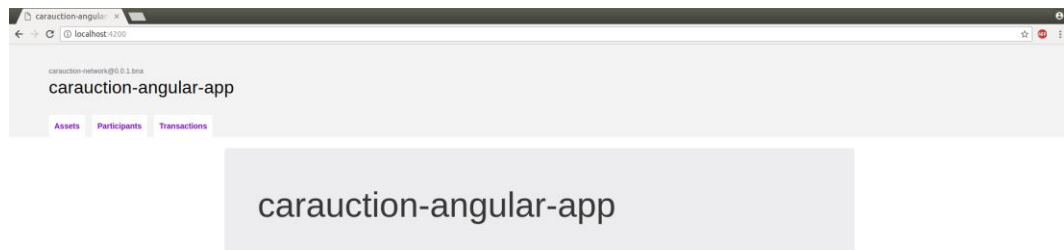


Figure 6. Home Page

2. Assets Tab

There are two types of assets:-

- i. Vehicle

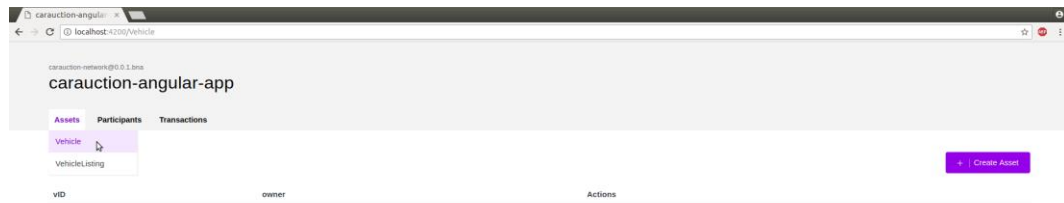


Figure 7. List of all Vehicle Asset

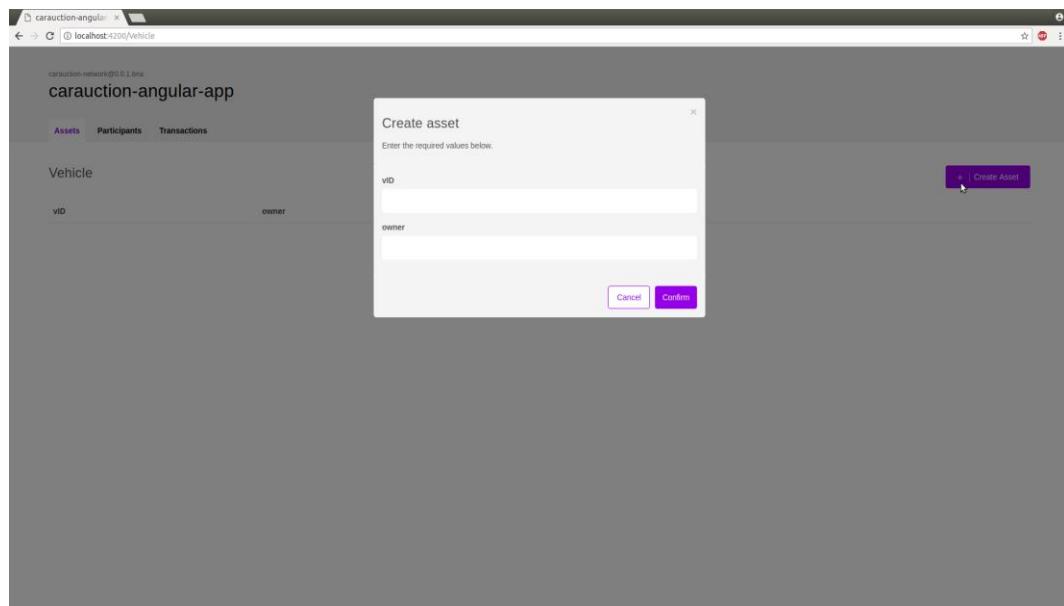


Figure 8. Create a new Vehicle Asset

Every vehicle is identified by a unique variable **vid**, and contains a reference to a member variable called **Owner**.

To create a new vehicle, one is required to insert a unique **vid** and a reference to a member who is supposed to be the **Owner** of the vehicle.

ii. Vehicle Listing

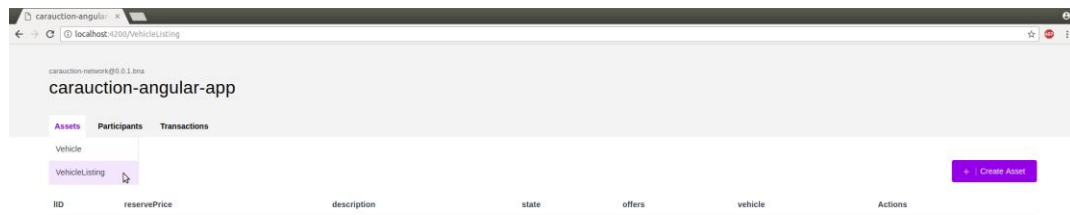


Figure 9. List of all VehicleListing

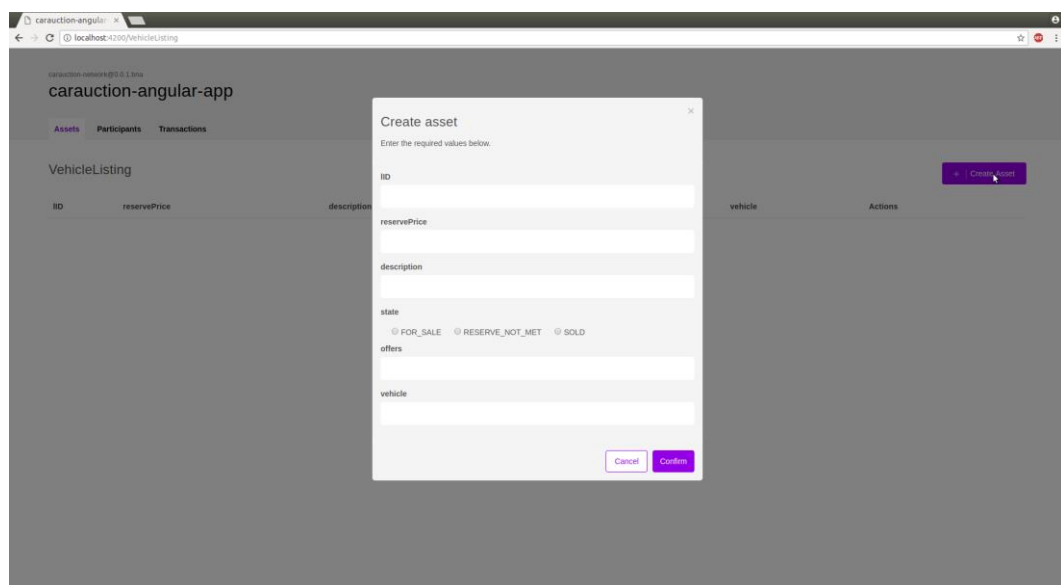


Figure 10. Create a new VehicleListing

Vehicle listing is used to list a vehicle for the auction.

It is identified by a unique variable **ID**. Every vehicle listing is associated with a vehicle using a reference to asset variable called **Vehicle**.

There is a threshold price known as **reserved price** associated, the bid price has to be more than reserved price in order to successfully buy the asset.

3. Participants Tab

There are two types of participants:-

- i. Member

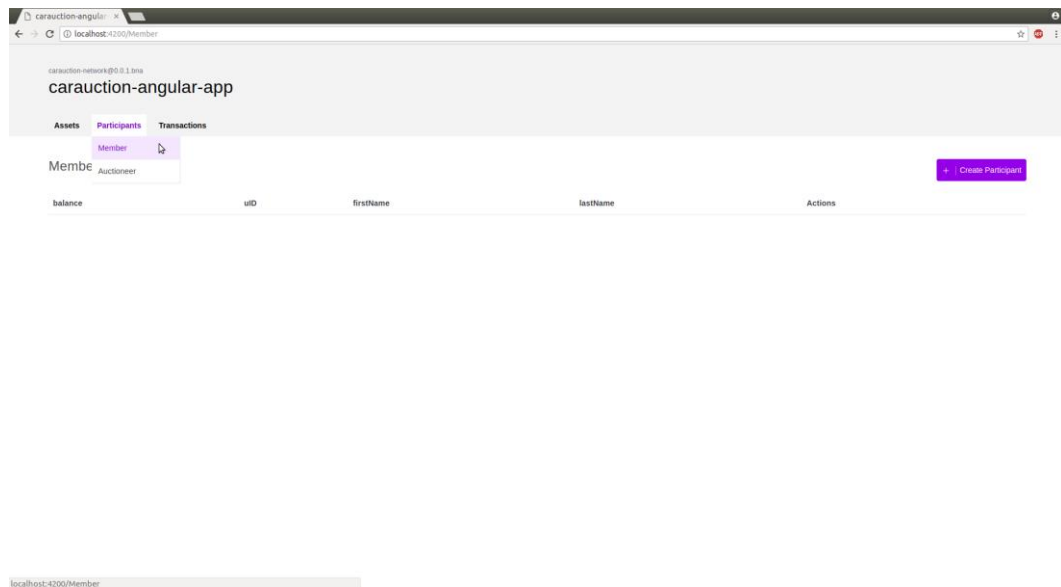


Figure 11. List of all members

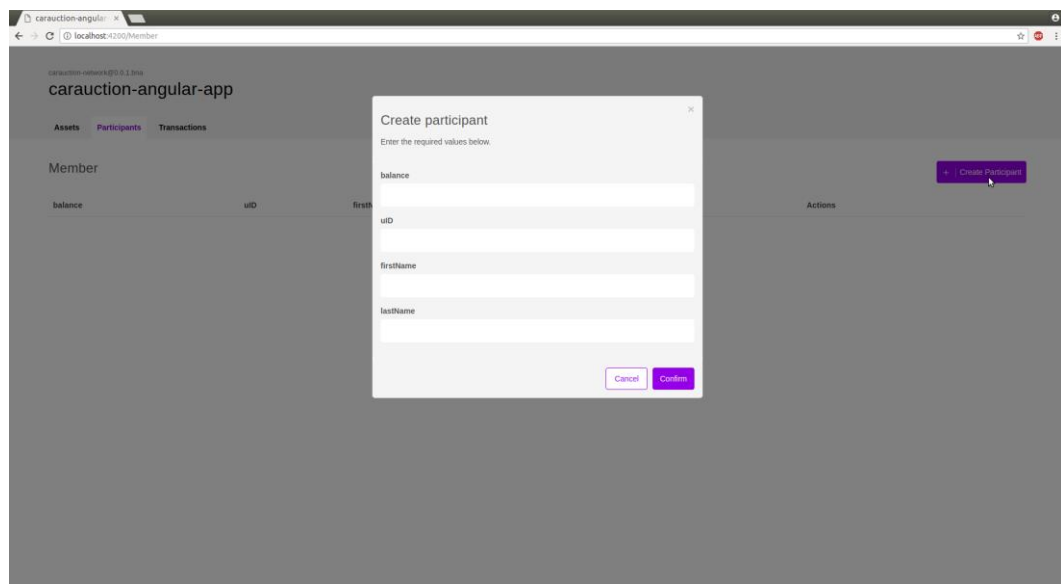


Figure 12. Add a new Member

Member is a participant of the auction, members are identified by unique member id **uid**. Every member has **Fist name**, **last name**, and **Balance** and is identified by a unique **uid**.

ii. Auctioneer

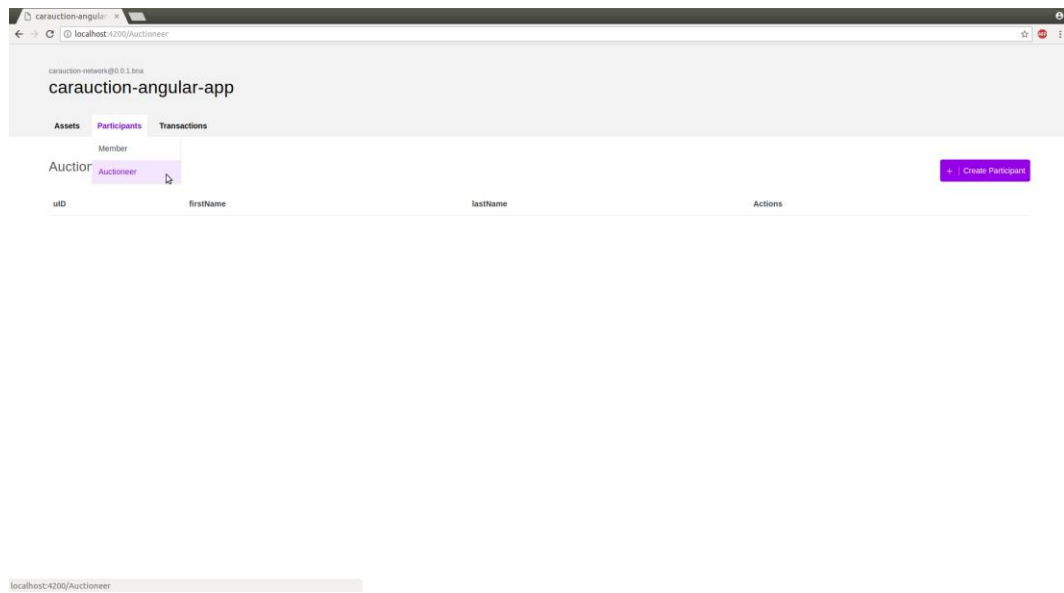


Figure 13. List of all Auctioneer

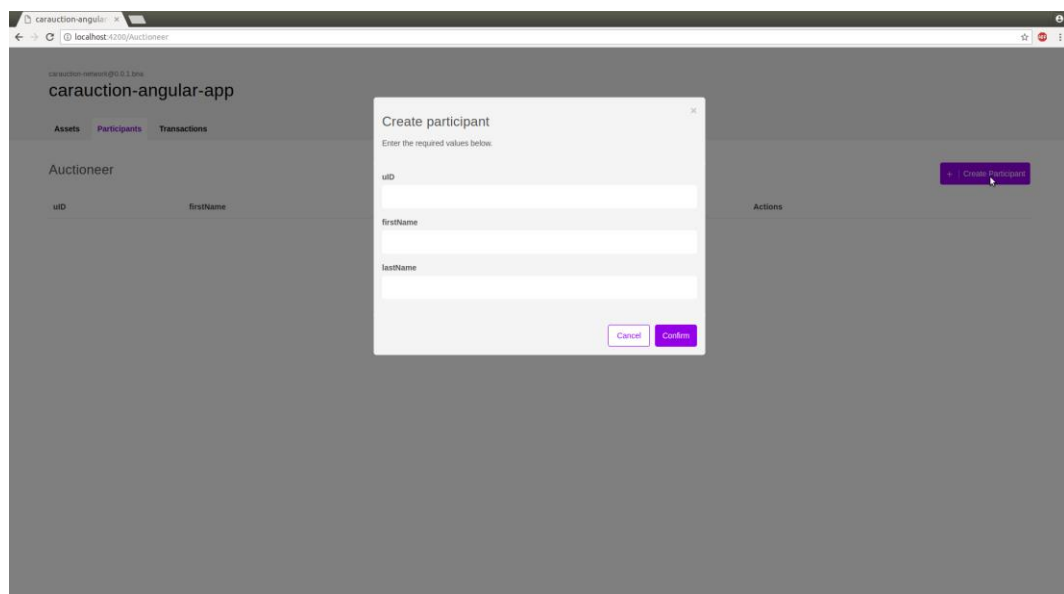


Figure 14. Add a new Auctioneer

Auctioneer is also a participant of the auction, but unlike member-auctioneer is used in order to provide oversight of the network.

Auctioneer is identified using a unique id **uid**. An auctioneer only has **First name** and **Last name** as its attribute.

4. Transaction Tab

There are two types of transactions:-

- i. Offer
- ii. CloseBidding

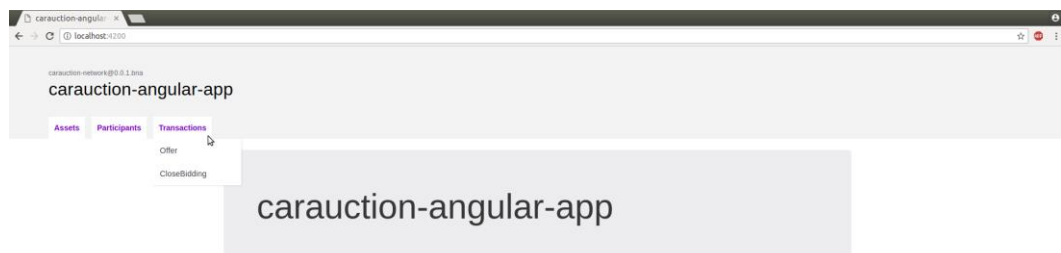


Figure 15. Transactions Tab

The **makeOffer** function is called when an **Offer** transaction is submitted. The logic simply checks that the listing for the offer is still for sale, and then adds the offer to the listing, and then updates the offers in the **VehicleListing** asset registry.

The **closeBidding** function is called when a **CloseBidding** transaction is submitted for processing. The logic checks that the listing is still for sale, sorts the offers by bid price, and then if the reserve has been met, transfers the ownership of the vehicle associated with the listing to the highest bidder. Money is transferred from the buyer's account to the seller's account, and then all the modified assets are updated in their respective registries.