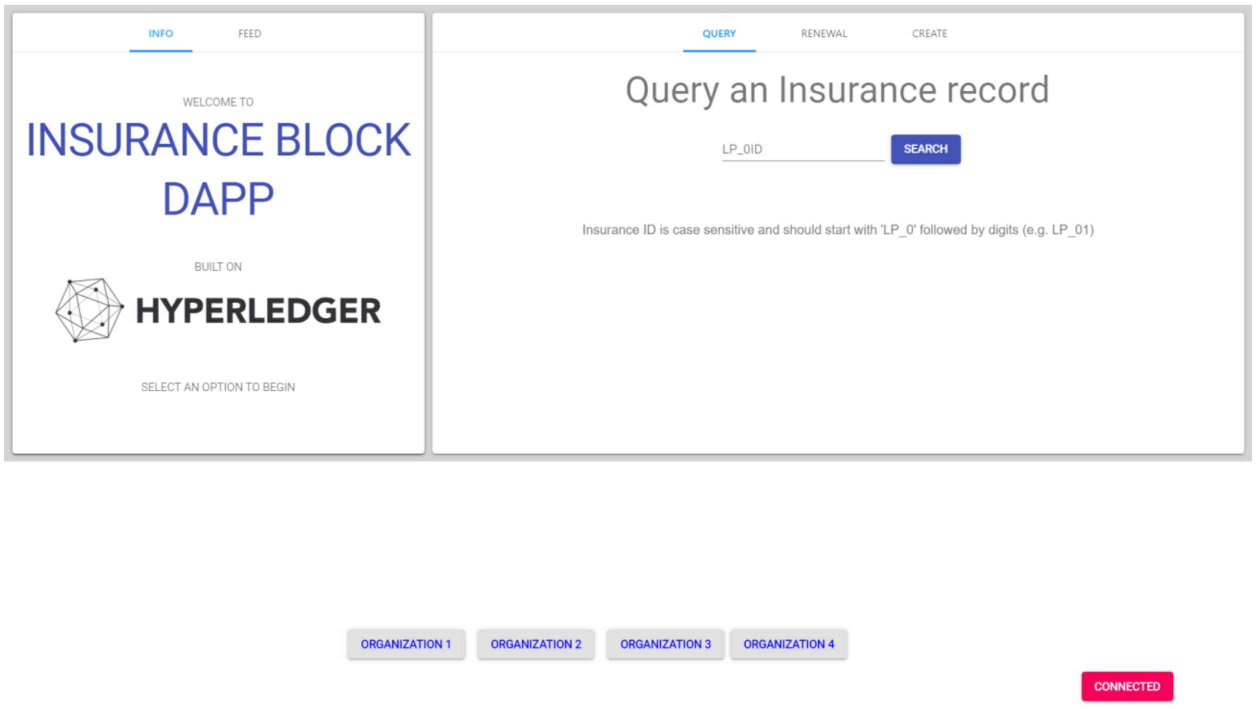
Consortium Boilerplate

Introduction

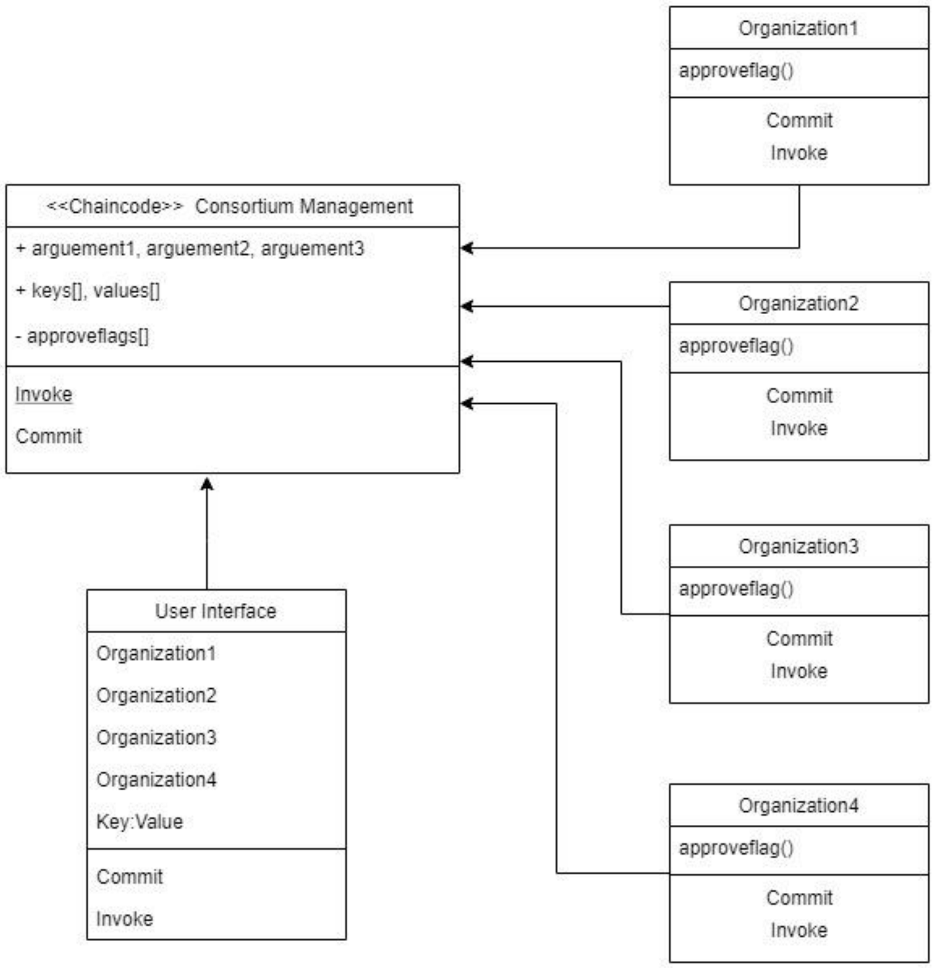
This is my capstone project for the blockchain development program at George Brown college. This project is based on private blockchain network using Hyperledger Fabric. Rather than focusing on applying a blockchain solution for a specific use case, I have focused on quality of life of existing blockchain users. Whenever I tried to learn something about Hyperledger Fabric, it was quite difficult to get it setup and grasp the abstract concept of consortium. I could only imagine the difficulty experienced by non-technical and businesspeople when trying to understand a potential private blockchain solution to their problem. This became the motivation for me to build a project that will not only help newcomers to understand the concepts of Hyperledger Fabric but also can serve as a boilerplate to start any consortium solution.

Image of the working project:



Insurance ID is case sensitive and should start with 'LP_0' followed by digits (e.g. LP_01)

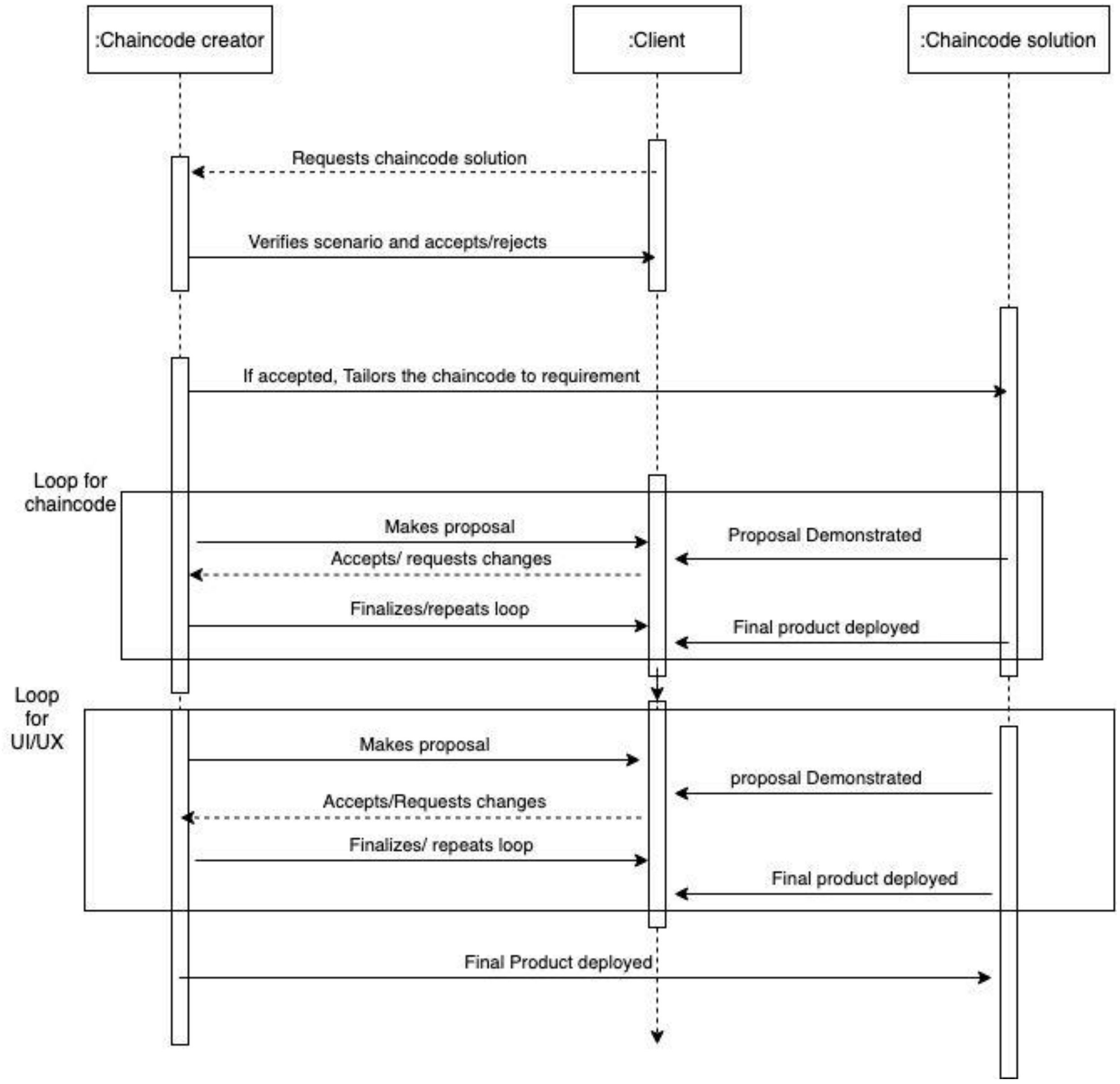
Technical Details



Hyperledger Fabric-samples provided by the Hyperledger community has been utilized in building this project. This project can be used to with up to 4 organizations but currently as a Minimum Viable Product, it has 1 organization with admin level priorities connected. Two more organization can easily be connected. For demonstration purposes, this project has been made for a group of insurance clients in need of a consortium solution. Currently this solution is designed from a consortium administrator point of view. Although some material of the Fabcar example has been used, the underlying Chaincode, backend and Frontend has been completely remodified to this use case. A readme file on steps to bring up the project can also be found in the root directory of this project.

How can this be used?

The sequence diagram below would help in visualizing the usage of this project in real-time. Once this project has been tailored to your specific needs, it can used to show your clients of your potential solution and ideas.



A little explanation of this would be that once the discussion for the use case and potential solution has been finalized, a demonstration of your solution can be made with this project. Once the solution has been approved, the development can be started right away using the Chaincode in this project as a base. Later on, after this solution has been approved, the backend and frontend of this project can also be utilized to start development on the user interface.

Steps to bring up the project

This project was built using Google Cloud Platform VM Instance and its highly recommended to run so. Make sure to clone the Fabric Samples as it is required to setup a test network with /bin and /config folder. Copy the following commands to setup the network, inside the /fabric-samples folder

```
git clone https://github.com/nerajok/demo.git

curl -sSL http://bit.ly/2ysbOFE | bash -s

cp -R ./PlateRegistry-DApp/platechain/ ./

cp -R ./PlateRegistry-DApp/chaincode/ ./

cp -R ./PlateRegistry-DApp/test-network/ ./

cd chaincode/platechain/javascript/

npm install

cd ../../../

cd platechain/

./networkDown.sh

./startFabric.sh  javascript
```

This will setup the HyperLedger fabric network where Chaincode is installed on org1 and org2.

Backend:

Next, we will need two separate VM Instances for backend and frontend. For backend server setup, copy the following the commands [current directory: /fabric-samples/platechain/]

```
cd express-backend/

npm install

node enrollAdmin.js

node registerUser.js

node backend.js
```

The backend server will start run on port 4001

Frontend:

Before setting the frontend server , make sure to do the following changes [current directory : /fabric-samples/platechain/]. Go to App.js which is inside [/react-frontend/src] and change the YOUR_EXTERNAL_IP value with your VM Instance External IP Address

```
constructor() {
    super()
    this.state = {
        showFeed: false,
        connected: false,
        socket : socketIOClient("http://[YOUR_EXTERNAL_IP]:4001/"),
        blocks : [],
    }
    this.switchFeedHandler = this.switchFeedHandler.bind(this);
}
```

This will connect to the backend server. Make sure you connection is just 'http' and not 'https'. After changing the value, copy the following the commands to setup frontend server [current directory : /fabric-samples/platechain/]

```
cd react-frontend/
npm install
npm start
```

The backend server will start run on port 3000

Both servers can be used with YOUR_EXTERNAL_IP value, so it will be

Backend: http://[YOUR_EXTERNAL_IP]:4001/

Frontend: http://[YOUR_EXTERNAL_IP]:3000/

Note: If you are not able to access the server outside the instance, check the firewall rules for VM Instance to allow these ports 4001 and 3000.

Resources used

FRAMEWORK	TOOLS		
Blockchain Network	Hyperledger fabric samples(test-network is used)	Fabric node SDK	Docker
Backend	Fabric network	Express	Socket.io
Frontend	React	Socket.io client	-