CSE 446: Project on HyperLedger Fabric

Deadline: 22-04-2024

Company Register

In this project, you have to develop an application which tracks the monthly expenses of companies. Each company should have a ***name***, ***ID(key)***, ***companyType(private/public/ngo)***, ***cashOutFlow(amount of money spent by a company in a month)***, ***cashInflow(amount of money earned by a company in a month), employeeCount, countryOfOrigin*** *and* ***companyReputation(values are: banned, poor, fair, good, excellent)***. Each company is identified with a unique ID. You have to introduce the following features:

1. **Users** can add a new company. However, Initially the ***cashInflow, cashOutFlow,*** *and* ***companyReputation*** property should be empty.
2. **Users** can see all the registered companies.
3. **Users** can search any registered companies by their unique id, country name or company reputation.
4. **Users** can update any information except ***cashInflow, cashOutFlow,*** and ***companyReputation*** of a company. (This is not logical but for the simplicity you can assume all users can do it)

In addition, there will be some **admins** in each company(there is already a built in identity module that maintains roles in hyperledger fabric. However, for simplicity, you can avoid this identity module. You just need to create 3 admin using their NID number and a **role** property.). Only the admin can update the ***companyReputation*** property. If a company has a **poor** reputation, their ***cashOutFlow*** amount should be determined by the admins and cannot be updated more than the determined amount. Moreover, if the ***companyReputation*** remains poor for three consecutive months, in this case if the admin wants, they can update the ***companyReputation*** with a status:**banned** and cannot be deleted/updated in future even by the admins. So that users can see the banned companies.

Instruction to develop the application

The list provided below is not mandatory. However these suggestions should reduce your workload.

- For this application you do not need to modify or manage wallet, admin or other users configuration. To make it more clear, you do not need to modify network configuration, enrollAdmin.js and registerUser.js files. You can use these files similarly to what we used in our lab.

- In addition, before executing every transaction there are some segments of code that gather the chaincode configuration path, wallet data, network and contract instances. You can use these parts without any modifications.
- Remember, whenever you update the chaincode you must start from the step where you shutdown and start the server. Otherwise, the update of the chaincode will not work.

To help your development further, we are providing a diagram and also flow of query(getting data from API / chaincode) and search feature in the next page. For the better understanding of the application, we are providing an illustration **based on our lab**.

The diagram shows how the backend, frontend and chaincode maintain their communication. Here, the respective file/folder names are also included for easier understanding. Here fabric-client is the frontend part, javascript folder contains the backend related services and fabric.js is the chaincode which is located in chaincode/fabcar/javascript/lib directory.