

Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Кафедра информатики

Отчет по предмету:
«Технология блокчейн»
По лабораторной работе №1

«Create a Hyperledger Composer solution»

Выполнил: Зюсько Кирилл Дмитриевич
магистрант кафедры информатики
группы №858642

Проверил: Прудник Александр Михайлович
доцент, кандидат технических наук

Минск 2020

Оглавление

1 Цель работы	2
2 Ход работы	3
2.1 Выбор и подготовка среды	3
2.2 Создание простейшей business network	4
Вывод	9

1 Цель работы

Создать простейшее блокчейн решение на основе фреймворка Hyperledger Composer.

2 Ход работы

2.1 Выбор и подготовка среды

Как видим в требованиях к использованию фреймворка <https://hyperledger.github.io/composer/latest/installing/installing-prereqs.html>

Важные пункты помечены в скобках (например, что NodeJS должна быть только 8.9.x – так, на последней стабильной версии 12 установка npm-пакетов завлилось на постинсталляции на сборке сс-файлов при постинсталляции). Данная работа выполнялась на MacOS 10.15.4

Далее, были выбраны следующие версии ОС, библиотек, фреймворков и т.д.:

- MacOS 10.15.4
- Docker 19.03.8
- Docker Compose 1.13.0
- NodeJS 8.17.0
- npm 5.10.0
- Git 2.26.0
- Python 2.7.12
- IntelliJ IDEA 2019.3.1 в качестве текстового редактора

Теперь можно приступать к установке пакетов hyperledger, а именно composer-cli (cli инструменты), composer-rest-server(REST сервер для выставления API для business network), generator-hyperledger-composer(для генерации ассетов), yo(кодогенерация приложений):

```
npm install -g composer-cli@0.20
npm install -g composer-rest-server@0.20
npm install -g generator-hyperledger-composer@0.20
npm install -g yo
npm install -g composer-playground@0.20
```

Установим Hyperledger Fabric:

```
mkdir ~/fabric-dev-servers && cd ~/fabric-dev-servers
curl -O https://raw.githubusercontent.com/hyperledger/composer-tools/master/packages/fabric-dev-servers/fabric-dev-servers.tar.gz
tar -xvf fabric-dev-servers.tar.gz
```

```
cd ~/fabric-dev-servers
export FABRIC_VERSION=hlfv12
./downloadFabric.sh
```

Проверим работоспособность среды:

```
cd ~/fabric-dev-servers
export FABRIC_VERSION=hlfv12
./startFabric.sh
./createPeerAdminCard.sh
```

Hyperledger Fabric успешно стартовала и admin card сгенерирована со следующими параметрами:

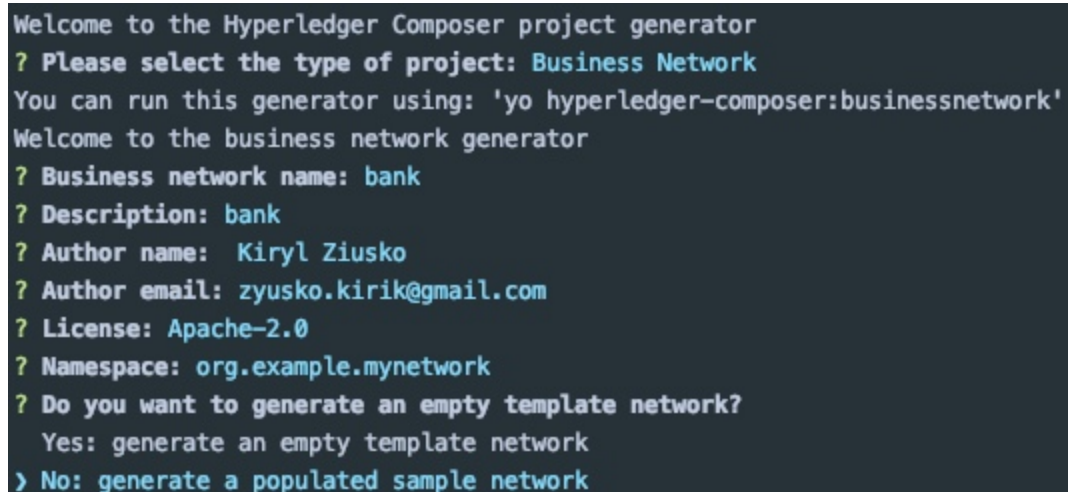
```
Card name: PeerAdmin@hlfv1
UserId: PeerAdmin
```

2.2 Создание простейшей business network

Создадим business network lab1 по обработке транзакций – переводе денег между двумя карточками.

Сгенерируем новую сеть:

```
yo hyperledger-composer:bank
```



```
Welcome to the Hyperledger Composer project generator
? Please select the type of project: Business Network
You can run this generator using: 'yo hyperledger-composer:businessnetwork'
Welcome to the business network generator
? Business network name: bank
? Description: bank
? Author name: Kiryl Ziusko
? Author email: zyusko.kirik@gmail.com
? License: Apache-2.0
? Namespace: org.example.mynetwork
? Do you want to generate an empty template network?
  Yes: generate an empty template network
  > No: generate a populated sample network
```

Рисунок 2.1 – интерактивный терминал

Создадим описание модели ассета Transfer, участника Bank и транзакции Trade в models/org.example.mynetwork.cto:

```
namespace org.example.mynetwork

asset Transfer identified by transferId {
  o String transferId
  o String description
  o Double amount
  --> Bank acquirer
}
participant Bank identified by bankId {
  o String bankId
  o String name
}
transaction Trade {
  --> Transfer money
  --> Bank issuer
}
```

Теперь создадим логику обработки транзакции в нашей сети в файле lib/logic.js:

```
/**
 * 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 tradeTransaction(trade) {
  trade.money.acquirer = trade.issuer;
  let assetRegistry = await getAssetRegistry('org.example.mynetwork.Transaction');
  await assetRegistry.update(trade.money);
}
```

При транзакции просто будет меняться банк, который хранит деньги.

Пропишем разрешения в сети. Так как сеть простейшая, то правила будут разрешать все всем:

```
rule Default {
  description: "Allow all participants access to all resources"
  participant: "ANY"
  operation: ALL
  resource: "org.dmitrysenkovich.personal.*"
  action: ALLOW
}
rule SystemACL {
  description: "System ACL to permit all access"
```

```
participant: "ANY"
operation: ALL
resource: "org.hyperledger.composer.system.*)"
action: ALLOW
}
```

На Hyperledger Fabric можно устанавливать сети в форме специальных архивов, сгенерируем такой для нашего проекта:

```
composer archive create -t dir -n .
```

Установим нашу сеть:

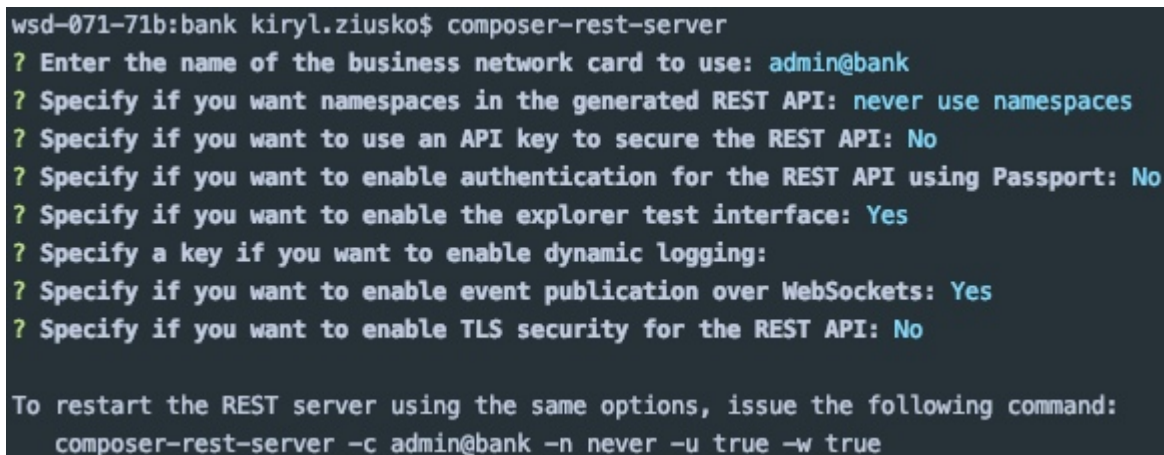
```
composer network install --card PeerAdmin@hlfv1 --archiveFile bank@0.0.1.bna
```

Проверим ее работоспособность:

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

Сгенерируем REST сервер для взаимодействия с сетью:

```
composer-rest-server
```



```
wsd-071-71b:bank kiryl.ziusko$ composer-rest-server
? Enter the name of the business network card to use: admin@bank
? Specify if you want namespaces in the generated REST API: never use namespaces
? Specify if you want to use an API key to secure the REST API: No
? Specify if you want to enable authentication for the REST API using Passport: No
? Specify if you want to enable the explorer test interface: Yes
? Specify a key if you want to enable dynamic logging:
? 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@bank -n never -u true -w true
```

Рисунок 2.2 – интерактивный терминал

В итоге на 3000 порту откроется REST сервис:

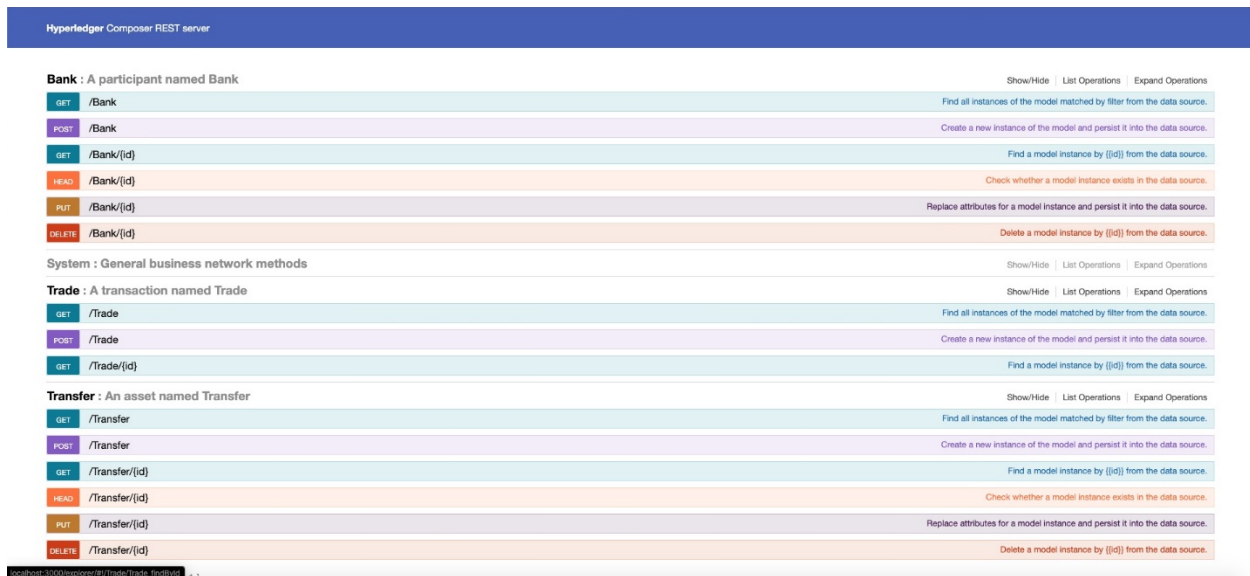


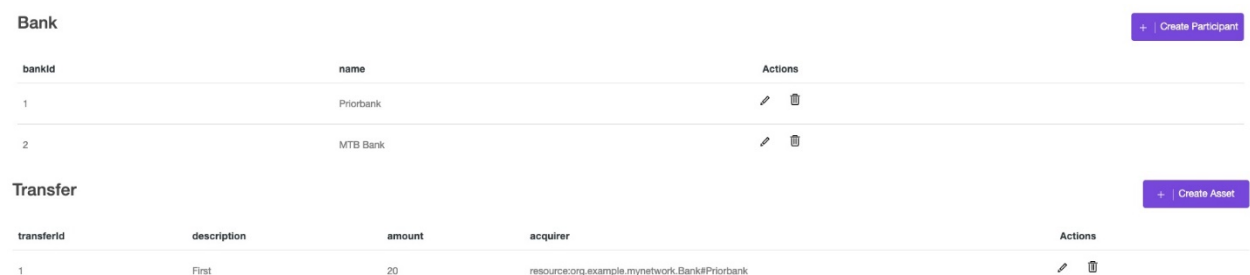
Рисунок 2.3 – UI сгенерированного REST сервиса

Сгенерируем Angular проект:

yo hyperledger-composer:angular

После этого перейдём в проект и запустим его:

npm start



Assets

Participants

Transactions

Trade

When invoked, this transaction will have the following attributes:

- money
- issuer

Invoke

Create participant

Enter the required values below.

bankId

name

Cancel

Confirm

Рисунок 2.4 – результат работы Ангуляр приложения

Вывод

В ходе данной лабораторной работы я развернул среду для разработки решений на Hyperledger Fabric с помощью Hyperledger Composer, создал свою сеть с предметной областью в банковской сфере, получил минимальные навыки работы с фреймворком.