



پروژه ی نرم افزار دوره کارشناسی

گزارش شماره ۹ - پیاده سازی پروژه (بخش ششم)

آریا رادمهر - ۹۷۴۶۳۱۲۵

دکتر سجاد حق زاد کلیدبری

March 10, 2022

بخش ششم (استخر تراکنش ها)

transaction-pool.test.js فایل

در فایل transaction-pool.test.js سعی در ساخت تست هایی جهت توصیف یک استخر از تراکنش ها میکنیم که بعد از ثبت شدن، میبایست معتبر باشند و در بلاکچین ذخیره و سپس از استخر به طور کامل حذف شوند.

Describe -> TransactionPool

Describe -> setTransaction ()

it -> adds a transaction

Describe -> existingTransaction

it -> returns an existing transaction given an input address

Describe -> validTransactions ()

it -> returns valid transaction

it -> logs errors for invalid transactions

Describe -> clear ()

it -> clears the transaction pool

Describe -> clearBlockchainTransactions ()

it -> clears the pool of any existing blockchain transactions

transaction-pool.js فایل

در فایل transaction-pool.js نیز شروع به ساخت کلاس transactionPool میکنیم و سپس متد های داخل کلاس آنرا به نحوی طراحی میکنیم که تست های تعبیه شده در مرحله ی قبل را با موفقیت به پایان برساند.

Class TransactionPool {

constructor ()

clear ()

```
setTransaction ( transaction )  
setMap ( transactionMap )  
existingTransaction({ inputAddress })  
validTransactions ( )  
clearBlockchainTransactions ({ chain })  
}
```

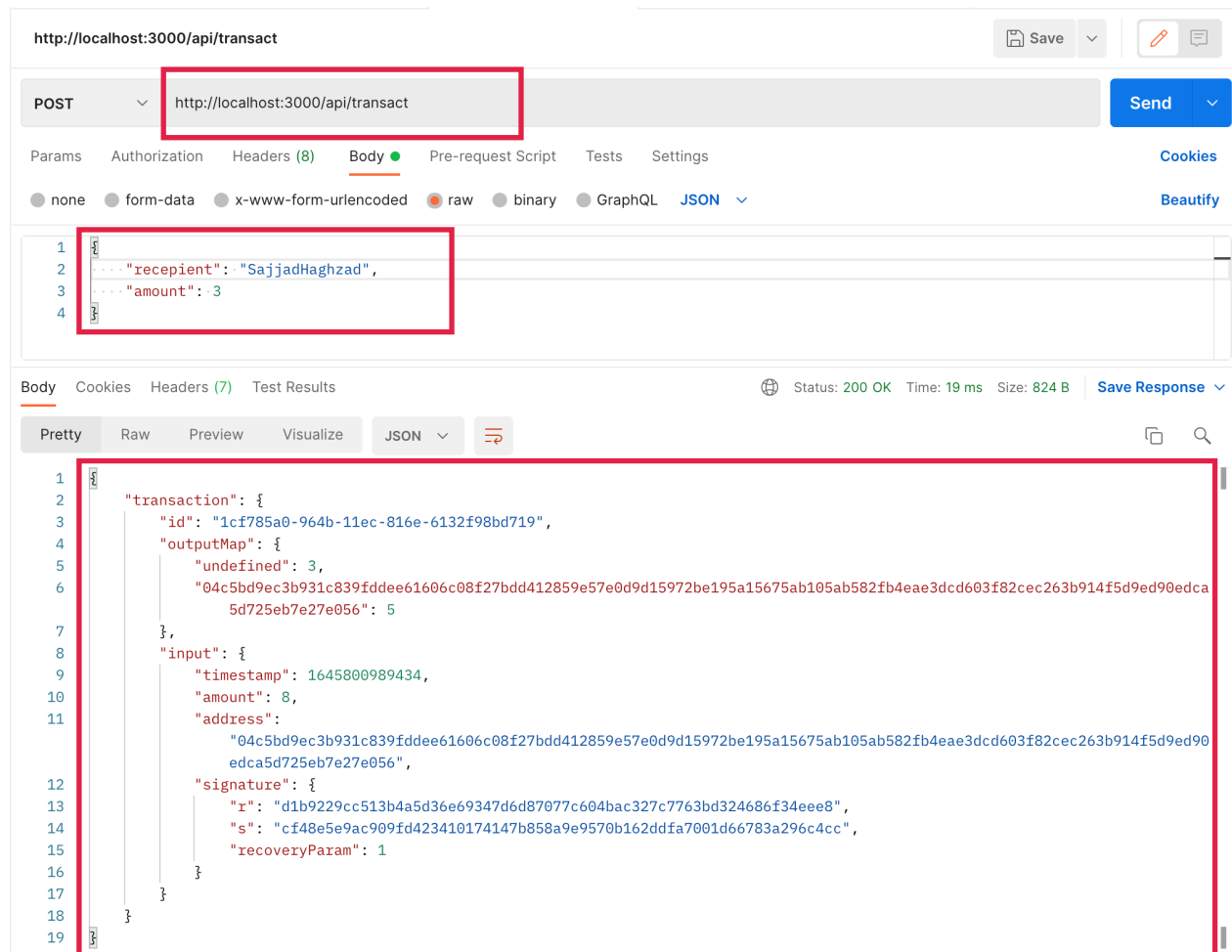
فایل index.js

در این فایل نیز سپس api مرتبط با transaction-pool-map را ساخته و توسط app آنرا با متد get پیاده سازی میکنیم.

```
app.get ('/api/transaction-pool-map', ( req, res ) => { });
```

در ادامه چند خروجی از قسمت api های پروژه با استفاده از ابزار postman را مشاهده میکنیم.

تصویر اول : توسط `/api/transact` یک تراکنش با نام گیرنده و مقدار سکه ی مبادله شده را ارسال میکنیم. سپس در قسمت پایین عکس، خروجی آن را به فرمت یک تراکنش کامل باز میگردانیم تا سپس برای ثبت در بلاکچین به استخر تراکنش ها ارسال شود.



تصویر دوم : چنانچه میزان سکه ی مبادله شده بیش از حد دارایی کیف پول شخص باشد، با پیغام خطا مواجه خواهیم شد.

The screenshot displays a REST client interface with the following details:

- URL:** `http://localhost:3000/api/transact`
- Method:** `POST`
- Request Body (JSON):**

```
{  "recepient": "SajjadHaghzad",  "amount": 100000}
```
- Response Status:** `400 Bad Request` (9 ms, 295 B)
- Response Body (JSON):**

```
{  "type": "error",  "message": "Amount exceeds balance"}
```

تصویر سوم : در این بخش نیز با فراخوانی `/api/transaction-pool-map` مشاهده میکنیم که یک تراکنش با

id تراکنش درخواستی در بخش قبل به استخر تراکنش ها فرستاده شده و آماده ی ماین و ثبت در بلاکچین

میشود.

همانطور که در قبل نیز گفته شد، تراکنش ها به طور کلی شامل یک id، outputMap و input هستند که اجزای

داخلی آنها جزئیات تراکنش را نشان میدهند.

The screenshot shows a REST client interface with a GET request to `http://localhost:3000/api/transaction-pool-map`. The response is a JSON object with the following structure:

```
1 {
2   "1df235e0-a0aa-11ec-ad1e-f59b495bb481": {
3     "id": "1df235e0-a0aa-11ec-ad1e-f59b495bb481",
4     "outputMap": {
5       "Aria": 2,
6       "04718852928423e40f943adfa10b602dabb709d98264b92d97e4c85e5aae5017b11de7d0f867886c2b94e26882324f9003b7f7db6b31663df6f
          f02376f9a3e11b9": 6
7     },
8     "input": {
9       "timestamp": 1646941304894,
10      "amount": 8,
11      "address":
12        "04718852928423e40f943adfa10b602dabb709d98264b92d97e4c85e5aae5017b11de7d0f867886c2b94e26882324f9003b7f7db6b31663
          df6f02376f9a3e11b9",
13      "signature": {
14        "r": "401825c308b6ef4458e6ba2de4f843d639caaddf52e549a984c428f2447aebf",
15        "s": "2ba66803e89394aee3b5904a731bcb9903f407486dc5f30a38b023e42a47eab6",
16        "recoveryParam": 0
17      }
18    }
19  }
```