CMPE 483 - Homework II

Safa Andac
Sahin Batmaz

InsuranceVO Smart Contract

In this homework, a modified model from homework I is implemented as a contract in order to use it in the blockchain system. In order to construct the model, a third party library is used in the contract. In our model, we need a iterable mapping in order to obtain active patient number and active hospital number in the system. In addition to that, we benefited from the library in deleting patients and hospitals from the system, whose contracts are expired.

1. Structures

In the execution of insurance, when a customer with valid insurance tries to get treatment in an hospital with valid deal, the insurance and the deal must be checked. For this reason two structures are needed.

a. The structure for customers consists of,
   i. end date of insurance,
   ii. remaining insurance limit

b. The structure for hospitals consists of,
   i. end date of deal,
   ii. treating costs,

c. The iterable mapping for customers which is used in controlling end date.
d. The iterable mapping for hospitals which is used in controlling end date.

2. Contract Functions

getInsuranceNote()

- This function takes parameters from customers; duration and payment. From the desired insurance duration and desired insurance payment, it calculates a multiplier number, in other words insurance note. With this number, insurance limit will be calculated. Since solidity contract does not support double or float variables, multiplier is kept in integer.
  insurance limit = payment * multiplier / 100

- **arguments :** duration of insurance
- **returns :** multiplier

getHospitalNumber() constant

- It gives the number of hospitals which have deals with the insurance company. In this function, the hospitals whose deals end are deleted thanks to using the iterable mapping for hospitals.
- **returns :** hospital number

getCustomerNumber() constant

- It gives the number of customers who have insurance in our company. In this function, the customers, whose insurance are expired, are deleted thanks to using the iterable mapping for customers.
- **returns :** customer number

payInsurance() payable

- It helps a customer to pay for an insurance with the given parameters.
- **arguments :** duration of insurance
- **returns :** success when the all conditions are met

In order to deal with a hospital, there is decision algorithm for deals. The decision algorithm depends on the number of customers the insurance company have and number of hospitals it deals, treating costs, Thanks to this decision algorithm, fairness and attractiveness can be observed in this system.

hospitalOffer() payable

- Hospital defines a treating cost and pays money to our insurance company. The function calculates an expected/desired payment from several parameters: It takes into account the number of customers that have contract with our insurance company and the number of hospitals that have contract with our insurance company and the treating cost. If the calculated value is less than the value that hospital offers, then the offer got accepted and hospital have an contract with our company and become active

- **arguments :** treating costs, deal payment
- **returns :** accepted / rejected

withdraw()

- Hospital gets money from our insurance company for treating a customer. Customer must run this code to be treated by the hospital. Hospital will not treat the customer before the customer run this function.

- **arguments :** treating costs, deal payment
- **returns :** accepted / rejected