2017

INSURANCEVO

CMPE 483 Sp. Top. in CMPE: Blockchain Programming

MERVE CERIT - 2012402015

SABRI KIZANLIK - 2007704452

Contents

1. Description of the Organization: InsuranceVO 2
1.1. InsuranceVO from Financial Viewpoint: How will the system finance its operations? 2
1.1.1. Revenue 2
1.1.2. Cost 2
1.2. The Goal of the Organization 3
1.2.1. Reducing Fraud causing from both parties: Patients and Hospitals. 3
1.2.2. Reducing Health Insurance Costs 3
1.2.3. Increasing Ease of Use 4
1.2.4. Increasing Cryptocurrency Penetration 4
2. Implementation of the Contract: msinsurancevo 4
2.1. Structs 4
2.1.1. Patient 4
2.1.2. Hospital 4
2.1.3. Offer 5
2.1.4. Record 5
2.2. Functions (APIs) 6
2.2.1. Registration Functions 6
2.2.2. Payment Functions 6
2.2.3. Voting Functions 7
2.2.4. Operational Functions 7
2.2.5. Helpers and Accessors 8
2.3. Tests 10
2.4. Web Application 11
3. How Does the System Work? 17
3.1. Registration 17
3.2. Treatment 17
3.3. Claim 17
4. A Brief Examination of the InsuranceVO 20
4.1. Attractiveness 20
4.2. Sustainability 20
4.3. Fairness 20
5. Conclusions and Further Studies 20
6. References 21

1. Description of the Organization: InsuranceVO

InsuranceVO is a crowdfunded, operated and governed smart contract based health insurance organization. With its blockchain-inspired infrastructure,

- It is not owned by an authority.
- Its business model does not contain any kind of profit for the sake of a centralized company.
- It cannot be stopped or interrupted by anyone.
- It does not contain any intermediate agents.
- It works based on a trustless peer-to-peer network.
- All transactions processed in the system are immutable, namely cannot be changed with any effect.

1.1. InsuranceVO from the Financial Viewpoint: How will the system finance its operations?

1.1.1. Revenue

The organization, InsuranceVO, will have the revenue coming from the user registration. In order to be enrolled to the system, users (prospective patients) have to pay certain amount of money, which is called as premium: annual registration fee for the health insurance service. This premium amount is set as 500 Ether in the implementation.

1.1.2. Cost

The cost structure of the InsuranceVO can be explained in two different classes. The first one is composed of the subscription percentages will be given to the hospital. In order to attract the hospitals into the system, 2% subscription percentage is applied. That is to say, for each approved claim transaction from the organization to the hospital, organization will give a commission of 2%.

1.2. The Goal of the Organization

1.2.1. Reducing Fraud causing from both parties: Patients and Hospitals.

Nowadays, health insurance companies are trying to deal with the claim frauds. InsuranceVO is aiming at minimizing this problem by having time-stamped files and transactions. Increasing the transparency in the insurance system would provide various advantages to both patients and hospitals while reducing the risk of fraud.

1.1.1.1. Further explanation on how to reduce the fraud

- Voting Mechanism:

Each year, registered users of the system will vote for hospitals, by means of voting, only the hospitals which have 51% votes will stay in the system in that year. Then, a fictional hospital cannot survive in the system due to the voting barrier; and, because of the premium, no malicious user can exist in the system.

- Transaction Check:

All transactions will be matched the private keys of the hospital and the patient. Apart from the reports and documents, having a transaction from patient to hospital is mandatory. For example, if the treatment is covered 80%, the patient has to pay to hospital by using her wallet. Even if the treatment is covered 100%, when the hospital provides the documents and claims the payment, if the claim is approved, system will send patient's wallet a notification to pay 0 ETHs only for approval. By necessitating this, we make sure that the all the information is matched and cannot be repeated.

1.2.2. Reducing Health Insurance Costs

Current system of health insurance contains three parties: Patients, Hospitals and Insurance Companies. The one and only goal of the insurance companies is making profit. The cost of these companies (as well as the profits) are paid from insurance packages bought by patients. By eliminating the intermediary element in this system, we are sure that there would be an immense reduction of premiums of the patients. Note that, premium is an annual payment made by prospective patients to buy the service of health insurance.

1.2.3. Increasing Ease of Use

Eliminating the third agent in the system results in direct communication between the patient and the hospital. Instead of cumbersome paperwork, using a smart-contract based web application will be enough. Furthermore, InsuranceVO system will also include a wallet application for smart phones, which enables making payment, claiming, and having offers in one-click.

1.2.4. Increasing Cryptocurrency Penetration

At the end of the day, we believe that an attractive, sustainable and fair health insurance organization is worth to enroll. Purchasing the same service at a lower price, would be the most preferable option for the people. Since the system is running on smart contracts, the organization requires cryptocurrencies, specifically Ethers. Such a system would increase the awareness of ETH in addition to its market penetration.

2. Implementation of the Contract: msinsurancevo

2.1. Structs

2.1.1. Patient

The Patient Struct contains 4 struct variables:

- The Patient Address
- A Boolean to check if the patient is paid premium
- Premium Amount
- A list of votes that the person gave to different hospitals, this list is kept as maps with the addresses of hospitals

2.1.2. Hospital

The Hospital Struct contains 13 struct variables:

- The Hospital Adress
- A Boolean to check if it is registered
- A Boolean to check if it is active
- A Boolean to check if it is in the voting period
- A Boolean to check if has an offer

- Subcription Percentage to be paid
- Number of “Yes” votes it has
- Number of “No” votes it has
- Start date of voting period
- End date of voting period
- Number of records it has
- A list of records
- An Offer

2.1.3. Offer

The Offer Struct contains 4 struct variables:

- The Hospital Name
- The Hospital Address
- The Description of the Offer
- Number of different coverages it contains
- A list of names of departments it contains
- A list of prices
- A list of percentage of coverages it provides

2.1.4. Record

A record can be explained as a detailed receipt of the treatment.

The Record Struct contains 12 struct variables:

- The Record ID
- A Boolean to check if the patient paid her portion
- A Boolean to check if the insured amount is paid to hospital
- A Boolean to check if the hospital’s subscription is paid
- A Boolean to check if the record is, active, if it has an uncompleted payment
- A Boolean to check if the patient approved the record
- Date
- The name of the department

- Total Cost of the treatment
- Coverage that offer provides
- Insured Amount
- The amount that patient will pay
- Subscription Amount

2.2. Functions (APIs)

2.2.1. Registration Functions

2.2.1.1. registerPersontoPeriod(Person Adress)

If a person is not registered, and the premium is paid, this function registers the person to the system. It returns true if the registration is done.

2.2.1.2. registerHospitaltoPeriod(Hospital Adress, Subscription Percentage)

If a hospital is not registered, this function registers the hospital for only 1 year(period), with the given subscription percentage. The voting stage starts with the registration of hospital and ends 1 week later. Adding offer to the system is also done while registering, there are separate functions for adding an offer. It returns true if the registration is done.

2.2.2. Payment Functions

2.2.2.1. payPremiumToRegister()

A person has to pay the premium in order to be registered in the system. This is a payable function, therefore after calculating the premium, the person should send the exact amount of premium by calling this function. The person that pays the premium is recorded to a list, for later checks in the registration process. It returns true if the payment is done.

2.2.2.2. payPatientPartion(Record ID)

When a record is created, namely, the patient goes to the hospital for a treatment; the uninsured amount which the patient has to pay is calculated. The patient is informed about this amount, and the hospital will not receive the payment unless the patient pays her amount. This is a payable function, that is to say, patient should call this function with the record id and amount given. It returns true if the payment is done.

2.2.2.3. payHospitalThePrice(Record ID)

After the patient is paid the uninsured amount, and the hospital is not paid; the organization does the payment to the hospital. The amount of this payments id the cost of the treatment that is indicated in the offer details. This is not a payable function, the contract itself sends the amount to the hospital. It returns true if the payment is done.

2.2.2.4. payHospitalTheSubscription(Record ID)

After the hospital is paid, the subscription payment is realized. This is not a payable function, the contract itself sends the amount to the hospital. It returns true if the payment is done.

2.2.3. Voting Functions

2.2.3.1. voteTheHospitalForPeriod(Voter Adress, Hospital Adress, Vote)

In the voting period of the registered hospital, a registered person can give a vote for the hospital. The vote can be YES (1) or NO (2). A person can vote a hospital only once. For each hospital, the number of votes is calculated. Only the ones having YES's more than NO's are allowed to give an offer. Otherwise, the hospital will be passive until the next period (1 year later). It returns true if the voting operation is done.

2.2.4. Operational Functions

2.2.4.1. startRecord(Hospital Adress, Patient Adress, Department Name)

When a registered person goes to a registered hospital as a patient, a detailed receipt of her treatment is created. This function helps to store the treatment details in the Patient, Hospital and Record; also, it calculates the insured amount, subscription amount and the amount that patient will pay.

2.2.4.2. addOfferOfHospital(Hospital Adress, Hospital Name, Hospital Description)

The offer of the hospital is introduced to the system in 2 stages. The first stage is calling this function. It gives the general information about the hospital, and stores these details in the offer.

2.2.4.3. addOfferCoverageOfHospital(Hospital Adress, Department Name, Cost of Treatment, Percentage of Coverage)

The second stage of adding an offer is calling this function. In an offer, there may be multiple departments and plans; and these should be stored in the offer. For example, Hospital X has its offer, and in the offer, Psychology treatments cost 100 ETH and the coverage for them is 80%. In addition, Neurosurgery treatments cost 1000 ETH and the coverage is 10%. By using this function, hospital can add more and more elements, like those mentioned, to its offer.

2.2.5. Helpers and Accessors

Apart from the main functions explained above, there are a lot of helper and getter functions. Some of them are serves for only testing purposes, and some of them provides parameters to main functions. They are also useful in the Web Implementation. Some of these functions can be seen in the list below.

- calculatePremium(Person Address): returns Premium Amount
- calculateSubscription(Hospital Adress): returns Subscription Percentage
- getRegisteredPerson(related adress): returns the registered Person's variables
- getPremiumPaid(related adress): returns the Person and its paid premium amount
- getRegisteredHospital(related adress): returns the registered Hospital's variables
- getRegisteredHospitalOffer(related adress): returns the offer of the hospital
- getRegisteredHospitalOfferAfterCoverage(related adress): returns the offer of the hospital with more details
- getPatientAmount(Record ID): returns the amount which patiet will pay
- getHospitalAmount(Record ID): returns the amount to be paid to the hospital
- getHospitalSubscription(Record ID): returns the subscription amount to be paid to the hospital

- checkifPersonRegistered(Person Address): returns true or false related to registration status
- checkifHospitalRegistered(Hospital Adress): returns true or false related to registration status
- checkifPremiumPaid(Person Address): returns true or false after checking if the person is paid her premium or not
- checkOffersOfHospital (Hospital Adress): returns true or false after checking if the hospital has an offer or not
- displayThePeriod (): returns the period id(number)
- displayTheMaxTrnxNum (): returns the number of records
- getHospNameofOffer (Hospital Adress): returns hospital name
- getDescofOffer (Hospital Adress): returns offer description
- getCoverCountofOffer (Hospital Adress): returns the number of elements in the offer
- getDescofCoverage (Hospital Address, index): returns the department name, in the given index of the coverage list
- getPricecofCoverage (Hospital Adress, index): returns the cost of treatment, in the given index of the coverage list
- getPerccofCoverage (Hospital Adress, index): returns the coverage percentage of treatment, in the given index of the coverage list
- getHospStatus (Hospital Adress): returns hospital status 0 or 1 in terms of being active or not
- getHosRecordCount (Hospital Adress): returns the number of records that a hospital has
- getRecordPatAddr (Hospital Address, index): returns the patient address in the given index of records list
- getRecordDepartment (Hospital Address, index): returns the hospital address in the given index of records list

- getRecordPatPayStat (Hospital Address, index): returns true of false after checking if the patient paid the unisured amount or not, in the given index of records list
- getRecordHosPaidStat (Hospital Address, index): returns true of false after checking if the hospital is paid the cost of the treatment or not, in the given index of records list
- getRecordSubscPaidStat (Hospital Address, index): returns true of false after checking if the hospital is paid subscription amount or not, in the given index of records list
- getPriceofDeptofHos (Hospital Address, Department Name): returns the cost of the treatment of the the department
- getPercofDeptofHos (Hospital Address, Department Name): returns the coverage percentage of the treatment of the the department
- getTheNumberofHos (): returns the number of hospitals
- getTheAddrofHos (index): returns the address of the hospital in the given index in the list of hospitals
- displayTheVote (Person Address, Hospital Address): returns the vote (1,2) which person gave to the hospital. 1 means YES, 2 means NO. It will return 0 if the person did not give the vote.
- displayTheVoteCount1 (Hospital Address): returns the number of YES votes of the hospital
- displayTheVoteCount2 (Hospital Address): returns the number of NO votes of the hospital

2.3. Tests

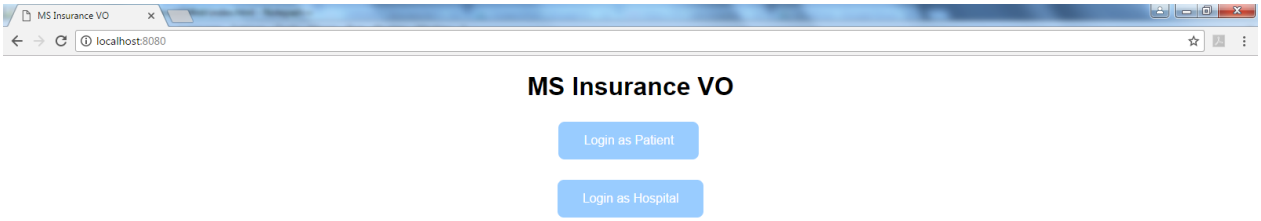
The functions explained in the previous section have been tested in 4 different platforms. The first one is the Solidity Browser. Since the source code is written and compiled there, all functions are tested in the real blockchain. Also, during web implementation, these functions are tested again in terms of functionality. Apart from those two, Populus and JavaScript scripts are used and the contract is tested just to be sure that the functions do

not give erroneous results. Populus is used in order to be fast in testing, and NodeJS is used for checking the behavior in the real environment with the companionship of Web3.

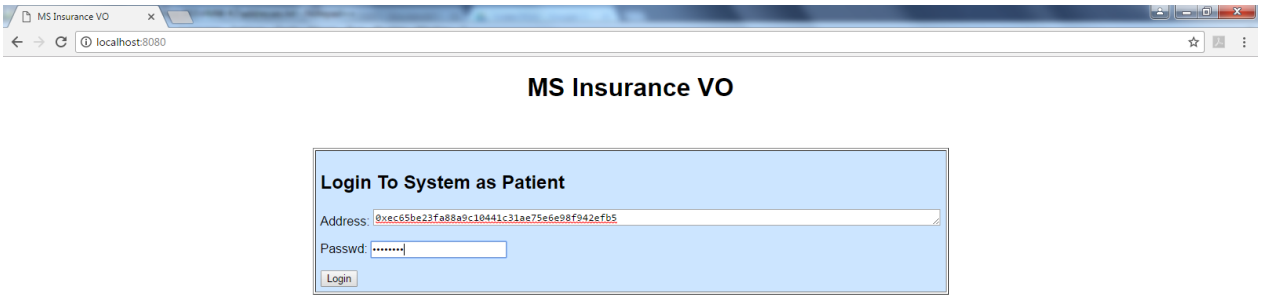
2.4. Web Application

In the implementation of the web application, JavaScript and HTML are used. JavaScript API of the Ethereum with its Web3 support is utilized. The functions explained above are called from contract and they connected to buttons and form in the HTML to work in a trigger-action fashion.

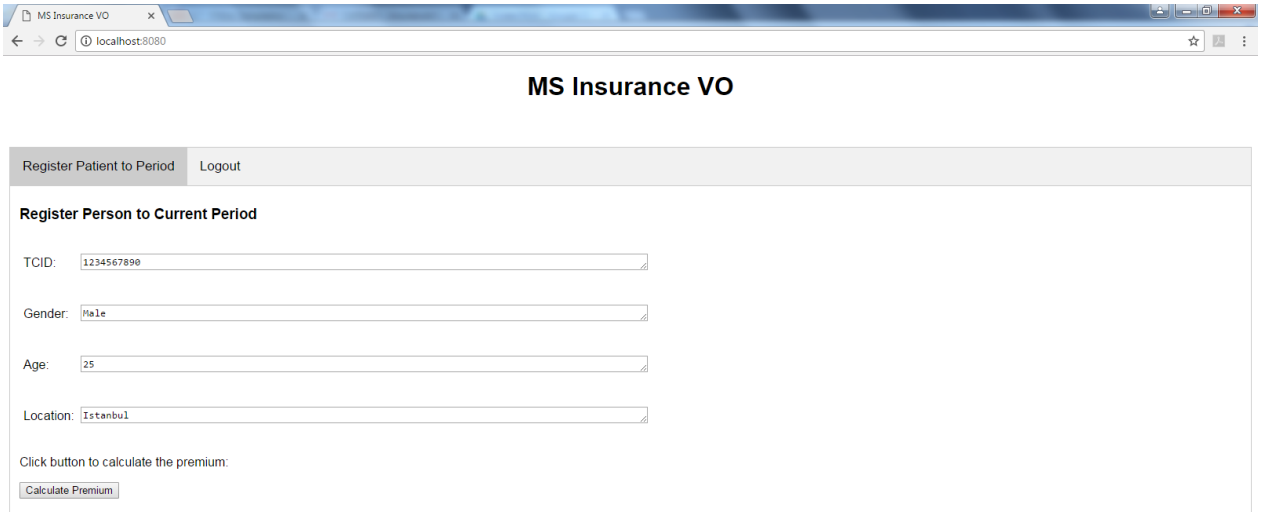
Below, one can see some screenshots from web application as an illustration of how the system works.



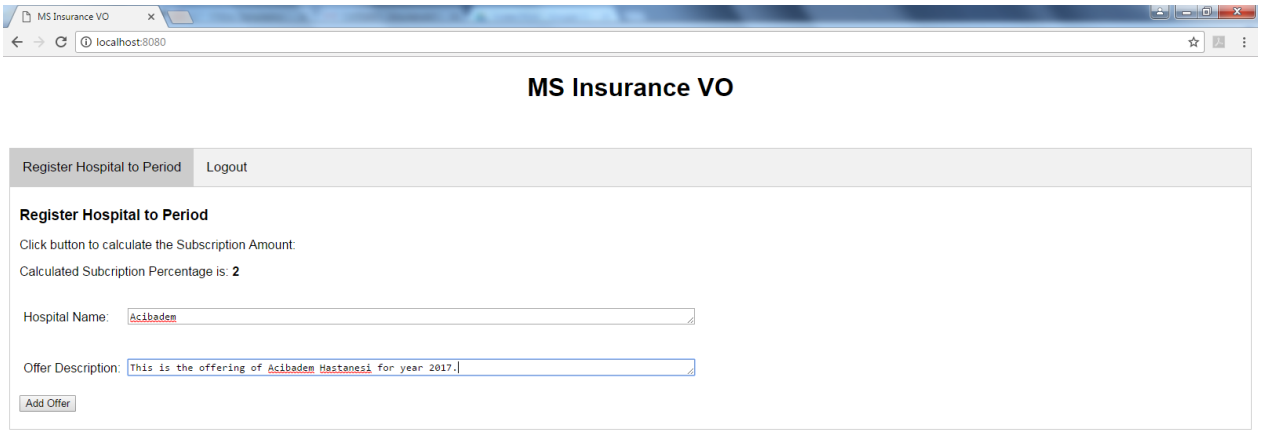
Screenshot 1 - Homepage



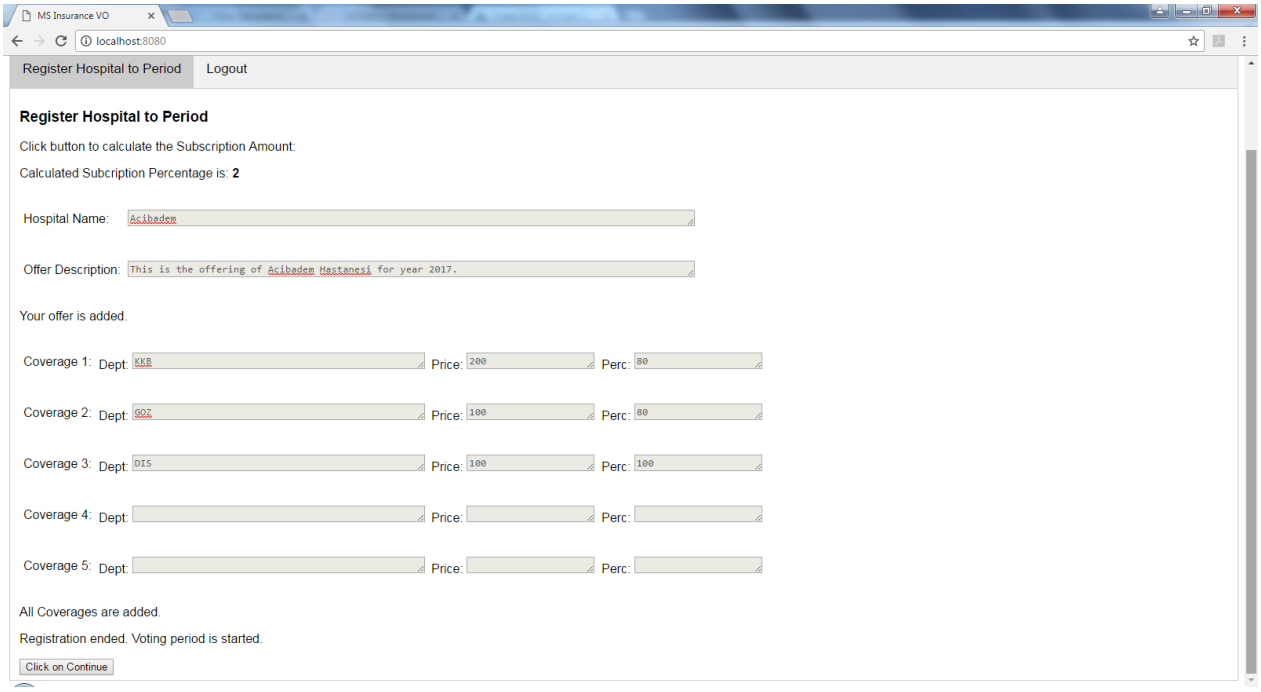
Screenshot 2 - Patient Logins(Unlock Operation)



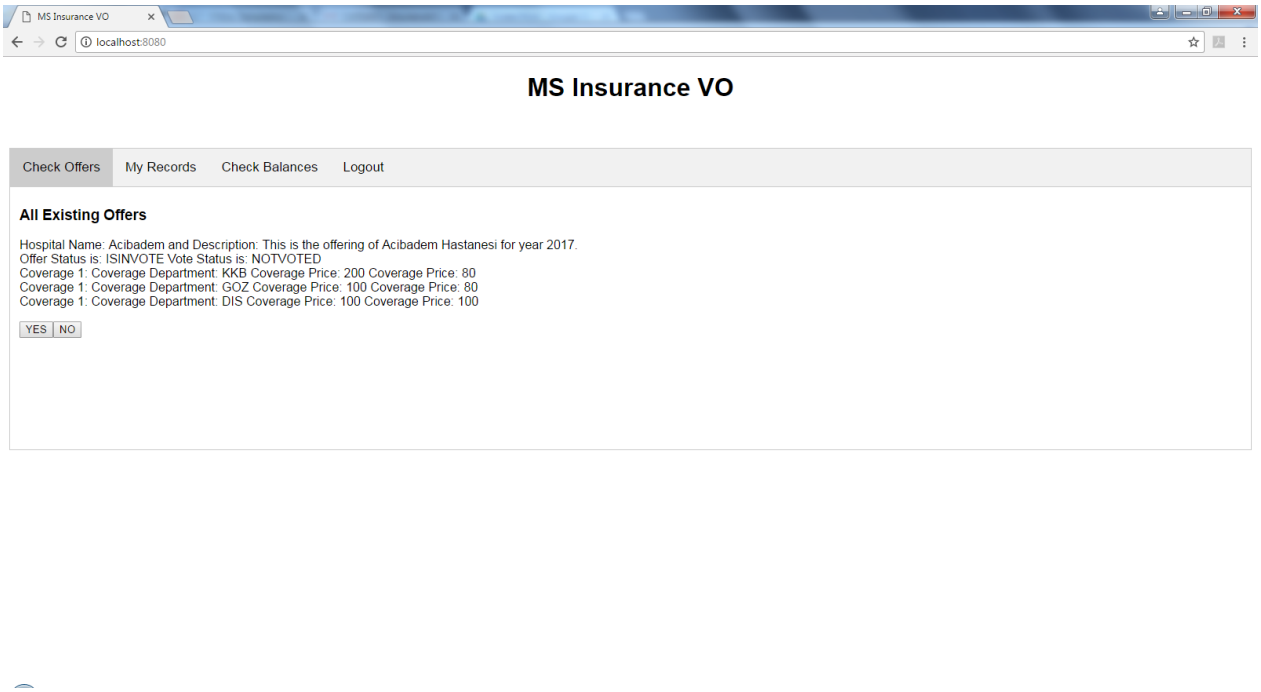
Screenshot 3 - Registration



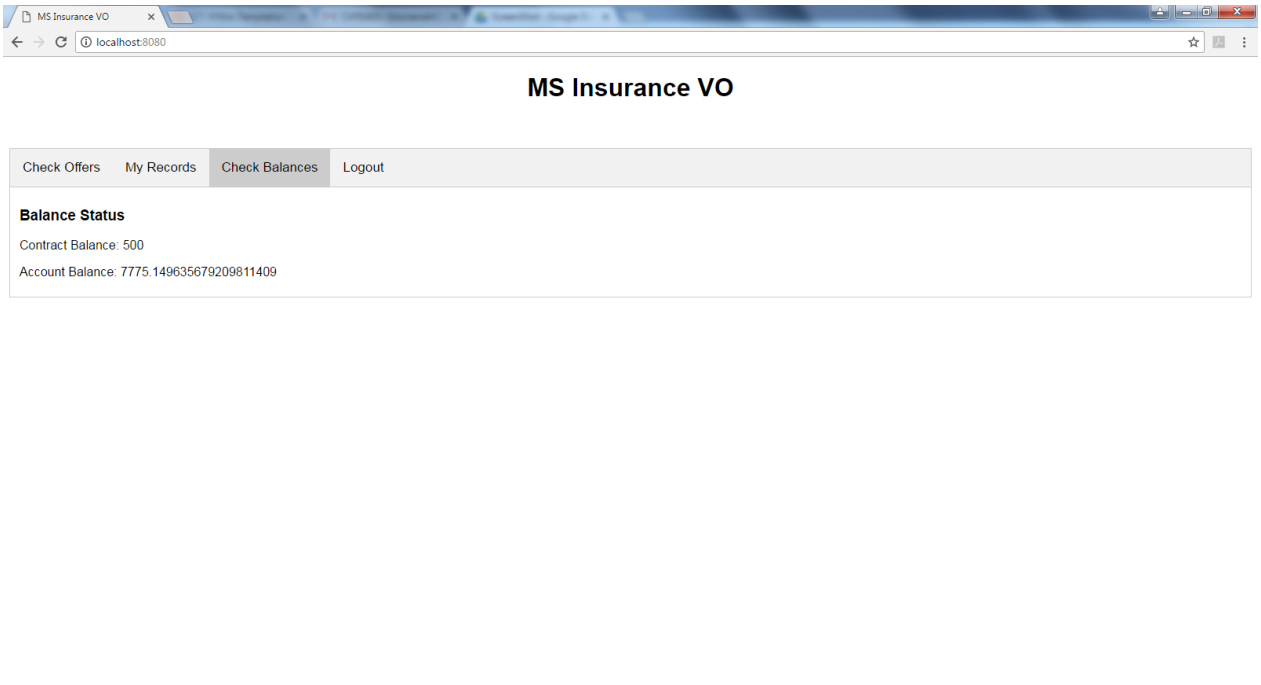
Screenshot 4 - Registration if Hospital - Adding Offers



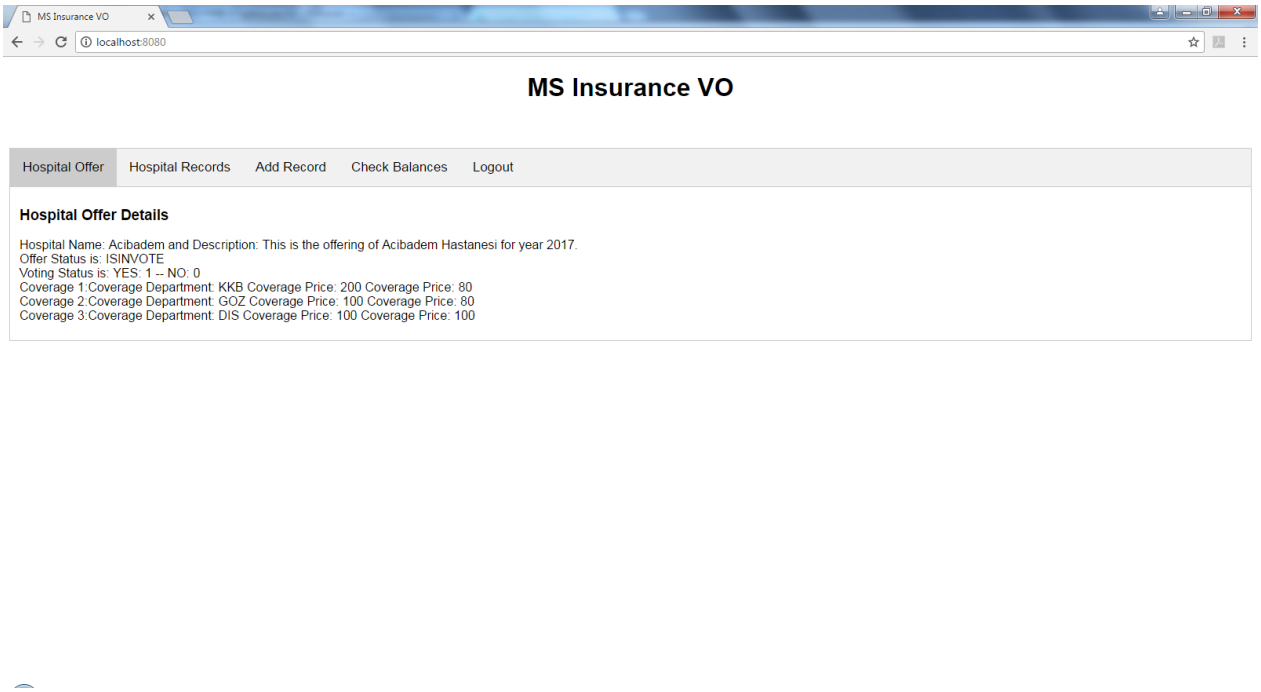
Screenshot 5- Registration if Hospital - Adding Offer Details



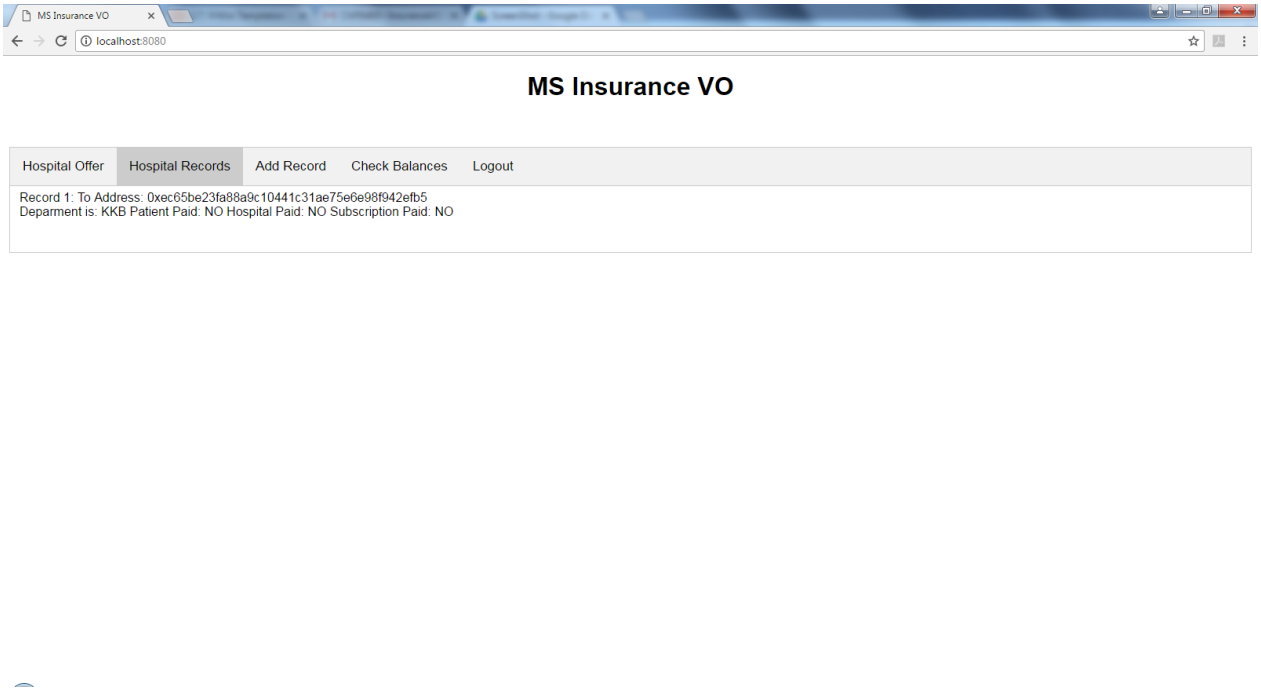
Screenshot 6 - Patient checks offers



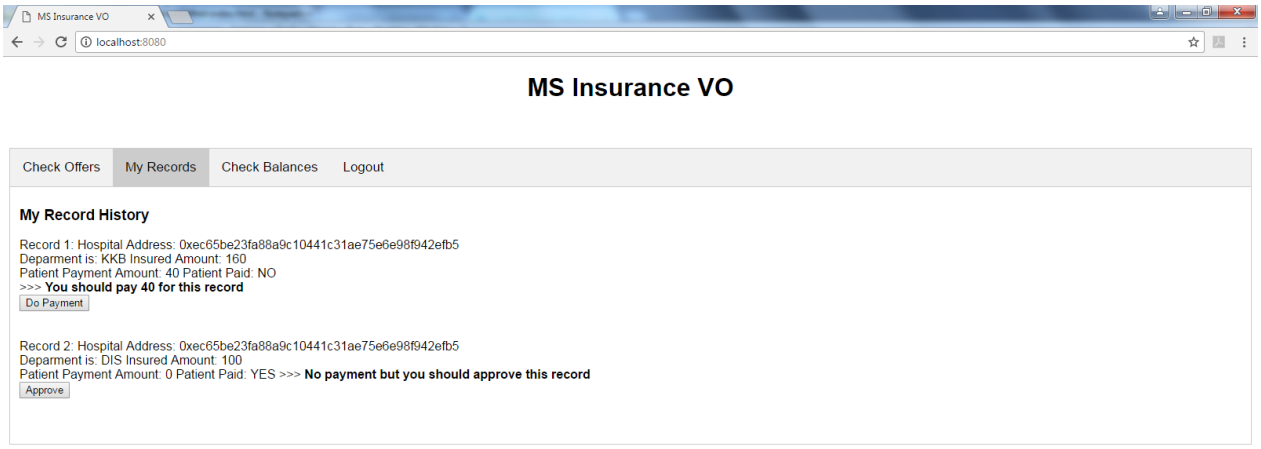
Screenshot 7 - Patient checks balance



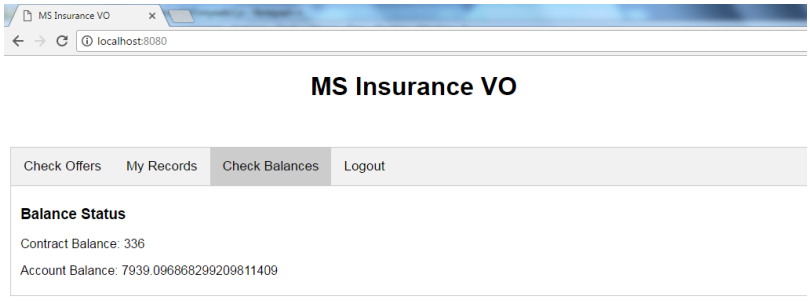
Screenshot 8 - Hospital's view, patient voted YES!



Screenshot 9 - Hospital's View to its records



Screenshot 10 - Patient's view to records, First is payable, Second is covered 100%



Screenshot 11 - Patient did the payment, post balances

3. How Does the System Work?

3.1. Registration

- The person registers in to the organization by paying the premium at each year. The required premium is calculated by the system. This calculation always gives 500 ETH in the implementation part, for the sake of simplicity.
- The hospital registers into the organization at each year. Subscription percentage is calculated for the hospital at the registration by its credibility value. This calculation always gives 2% in the implementation part, for the sake of simplicity.
- After registration of the hospital, the hospital is voted in the organization. If voting is resulted as 51% and more positive, the hospital is kept in the organization. Otherwise, the hospital is removed.
- After registration, the hospital can declare its offers into the system that states the insurance details of its treatment and department with the covered percentage and the price. These offers are stored in the organization and cannot be changed. All prospective patients can list and see hospital offers and select one for treatment. The hospital can create only one offer within a period.

3.2. Treatment

- For treatment, the person selects a hospital and an offer by looking the available listed offers through the organization interface. (web)
- The person goes to the hospital and gets the health check.
- The hospital creates a record in the system for the person and her treatment.

3.3. Claim

- The payment process is done in the hospital after the treatment with the involvement of the hospital and the person.
- Since the record is stored in the organization, and not closed yet; payable amounts are calculated by the organization.
- The organization firstly takes the patient payment (uninsured amount). Insured amount payment is done to the hospital, only after the patient's payment is received.

- At the end of the process, the organization does the subscription payment as agreed percentage at the registration, to the hospital. The subscription amount is transferred from organization to hospital.

In Figure-2 below, a claim process is visualized.

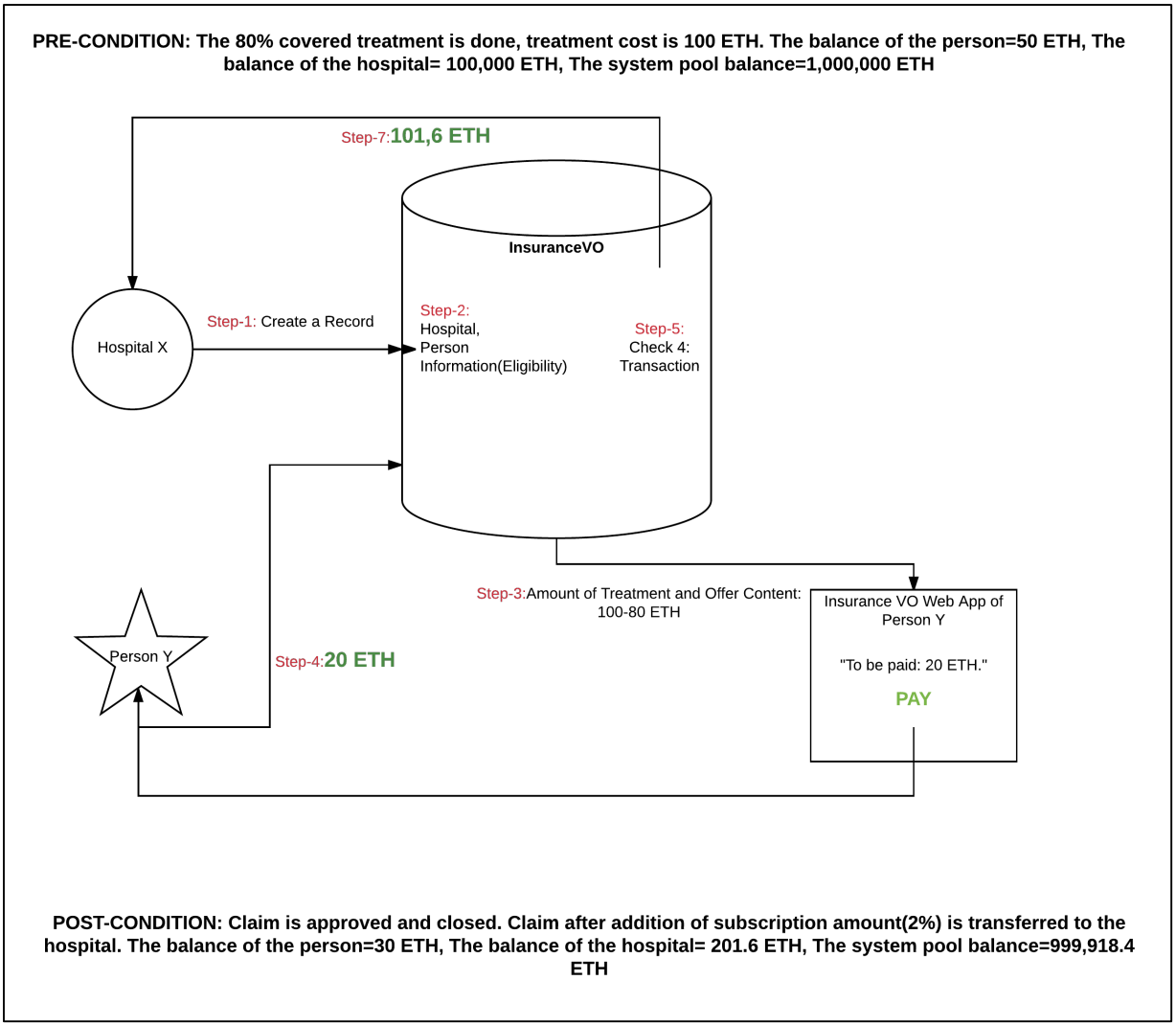


Figure 1 - 80% covered treatment claim

Please note that, InsuranceVO, InsuranceVO Web App, Hospital X and Person Y are visualized in that way for the sake of understanding. It is depicted in the figure below as if InsuranceVO is the intermediary agent. In the real system, everything occurs in the InsuranceVO, InsuranceVO Web App is an extension of the system. Hospital X and Person Y are the accounts in the InsuranceVO system. Then, there is no intermediary steps. In Figure-3 below, almost the same claim process is visualized except the coverage percentage. All the explanations above is valid for this figure, too.

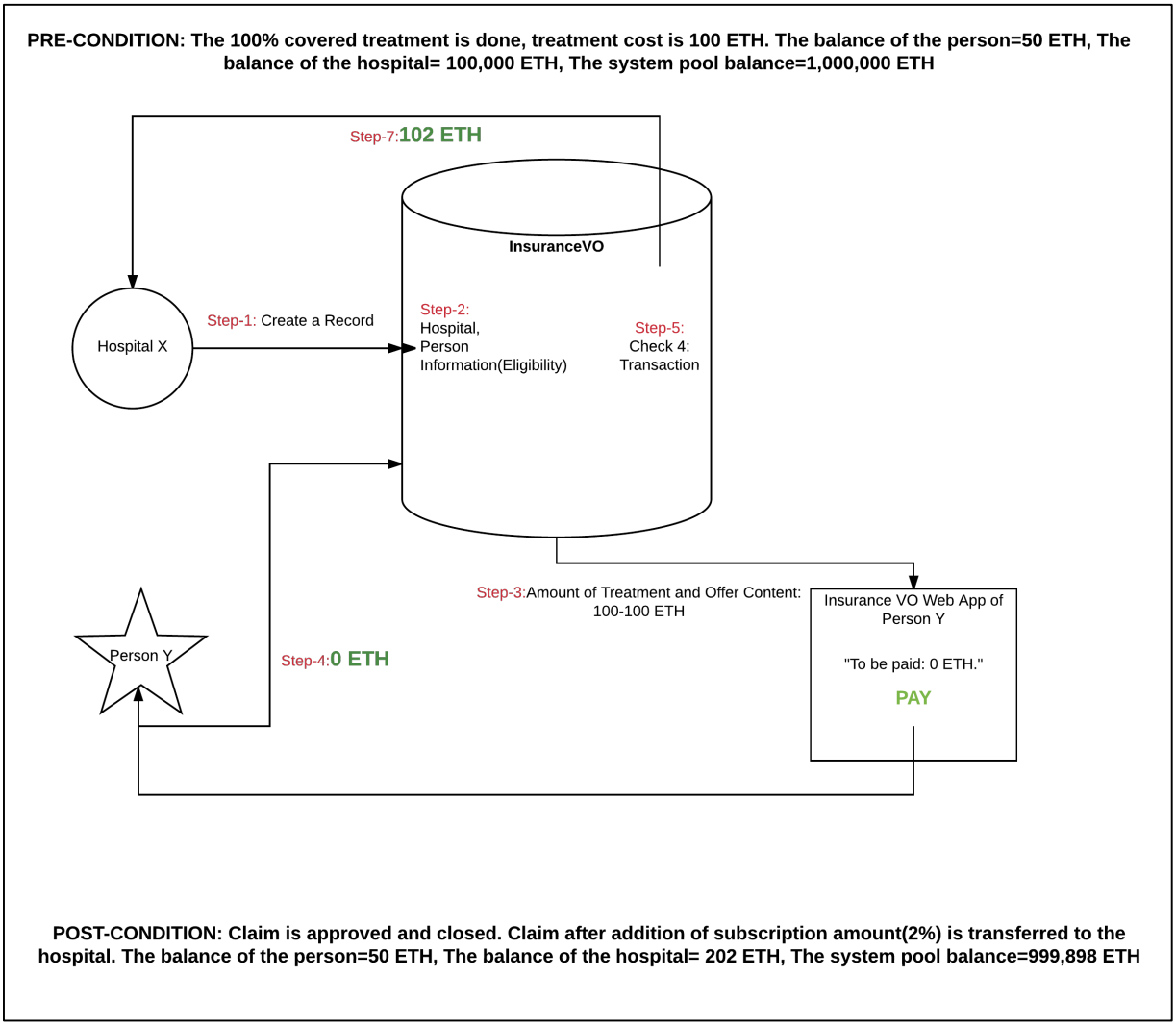


Figure 2 - 100% covered treatment claim

4. A Brief Examination of the InsuranceVO

4.1. Attractiveness

InsuranceVO provides the same insurance service at a lower price, while eliminating intermediary agents. Also, from the users point of view, there is only a web application to manage and use the system. From the hospital's point of view, they are paid subscription percentages for each transaction, and they can deliver their claim right away after treatment. Also, they do not have to deal with hardcopy paperwork.

4.2. Sustainability

InsuranceVO is a system that is crowdfunded, no company is associated with it. All premiums gained from registered users are spent on payments to hospitals. The amounts of the premiums will be calculated based on risk measures and past data, it is not the same case for implementation though, so that positive return will be obtained, and transferred to the system pool for future usage on again patients.

4.3. Fairness

We believe that our model will minimize the fraud, so it will prevent any injustice. Also, decisions will be given by means of voting and they will represent 51% of the system users. No personal and company interest exists in the system besides the patient's own interest.

5. Conclusions and Further Studies

In conclusion, we believe that the health insurance model we developed, InsuranceVO, will succeed in getting attention of the current health insurance customers, and providing them an attractive, sustainable and fair system. Undoubtedly that, the implementation and model are open for improvement. As a future work, premium determination may be made more deliberately. Since premiums are the only component to be counted as revenue, the determination of the premiums is vital for the system sustainability. The features below may be evaluated to calculate the premium of a patient:

- Past surgical operations data
- Number of people enrolled in health insurance companies

- Annual reports of the insurance companies, mainly the balance sheet and the income accruals
- An estimation of number of people that will use our system soon
- A classification of the people by the risk factors: Age and Check-Up Results

Also, discounts in premium as an incentive to register again can be made; and reports can be stored using IPFS to reduce fraud. These improvements can be implemented easily with enough capital and time, since the main structure of the organization is built.

6. References

[1] Nissim, Doron. "Analysis and Valuation of Insurance Companies." *SSRN Electronic Journal*(n.d.): n. pag. Columbia Business School. Web. 15 Mar. 2017.

[2] https://www.zurich.com/_/media/dbe/corporate/docs/financial-reports/2015/annual-report-2015.pdf?la=es&hash=280638A612F956268FC69F3F0CCF69FD86C3C21F

[3] Company, Nippon Life Insurance. "Annual Report." *Annual Report*. Nippon Life Insurance Company, n.d. Web. 14 Mar. 2017.

[4] "Surgical Operations and Procedures Statistics." *Surgical Operations and Procedures Statistics - Statistics Explained*. Eurostat, n.d. Web. 14 Mar. 2017.

[5] "Private Health Insurance Statistics." APRA. Http://www.apra.gov.au/phi/publications/pages/industry-statistics.aspx, n.d. Web. 15 Mar. 2017.

[6] "Insurance Balance Sheet Review of the Bulgarian Insurance Sector." ,Eiopa, Web. 15 Mar. 2017.

[7] Ethereum JS/Web3 API, https://github.com/ethereum/web3.js/

[8]" CMPE483 Lecture Notes", Ozturan. Bogazici University, 2017.