**SIMULATION**

In the following lines we are going to provide you with a simulation of how our prototype works. **Players:**

1. **Person\_1**: An individual that initialize a personal profile with his/her information and wants to buy a car insurance.
2. **Person\_2**: Same as Person\_1.
3. **Insurer**: The insurance company from which the two individuals buy the insurance policy.
4. **Police**: The official *source* that ascertains car accidents ad draws up official reports.

**Contracts**:

1. **PersonalProfile:** This SmartContract manage all the individual personal information, it is used by Person\_1 and Person\_2.
2. **InsurerContract:** The insurance company uses this SmartContract to manage the relationships with insured individuals.
3. **PoliceReport:** This SmartContract is used by police in order to document traffic accidents, it contains all the information about a car accident.

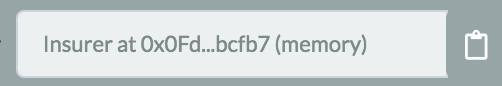
**Notice** : Every SmartContract should be deployed using a different account, since each is used by a different entity.

**Procedure**

In a very logical way we start with the initialization of the insurance company through the InsurerContract:

* *Compile* InsurerContract.
* Select an account and *Deploy* the contract.
* Click on Balance, as you can see it is equal to 0, it means that the insurance company in this moment has no money, Try to click on *insured\_account* and as you can see it does not return anything, because no one has yet bought a policy.

Initialize Person\_1 with PersonalProfile:

* *Compile* PersonalProfile.
* Select an account (different from the previous one) and *Deploy* the contract.
* Fill the function *InsertPersonalInfo*: name, surname, birth, licenceID, vehicle (*license plate*). This function records all the information about a person, as if it were an official and uncorruptible document. Click on transact.
* Now if you click on *viewPersonalInfo*, you can see all the information.
* Try to use the function *Update History*: as you can see it returns an error, since the individual can just initialize his state but can’t update it, only the police can, when a car accident occurs.
* If you click on *howManyAccident* you can see that we have 0.
* Now copy the InsurerContract address from the symbol (the one circled In red) 
* Past it in the function *pushToInsurer* and click on transact, in this way the Person\_1 provides all his personal information to the insurance company (be sure that the account is the one that you used to deploy the PersonalContract). This information - ideally - should be useful for estimating the risk premium, but in our simulation for simplicity we have assumed a homogeneous risk premium for all individuals and it amounts to 5 ether.
* Go to the InsurerContract, in *insured\_accounts* select 0 (that is the index of the first person who passed his own information), as you can see now there is one address that is the one of the account of the Person\_1. Paste this address in the *viewInsured* function and as you can see now you have all the information about Person\_1, pay attention at the line 5: *bool: false*, it means that the person has not already payed the risk premium and so it is not insured.
* Go back to PersonalProfile and in the function *payAndgetInsurance*, paste the same address as before (the one of the InsurerContract), select *Value* equal to 5 ether and click on *Transact*, Through this procedure the Person\_1 pays is risk premium in order to get the insurance policy (be sure that the account is the one that you used to deploy the PersonalContract). **NOTE**: here we hypothesize that the risk premium is 5 and equal for everyone.
* Go to InsurerContract, click on *Balance* and as you can see now there are 5 ether, this means that the policy has been payed. Click on insured\_accounts, there is the address/account of Person\_1, paste this address in the *viewInsured* function and as you can see now the line 5: *bool: true*, it means that the person has payed the risk premium.

Initialized Person\_2 with PersonalProfile:

* Select a different account from the two previous ones
* Repeat the same procedure as Person\_1 with different personal information and then pay the risk premium to be insured.
* Check if everything worked: go to *InsurerContract*, in the function *insured\_accounts* select 1 (the index of the second person that passed the information), copy that address and paste it in *viewInsured,* now you have all the information about Person\_2 and the line 5 is *bool: true*.
* Check also the balance of the Insurer, it is 10 Ether, that are the two risk premia.

**NOTE**: Both Person\_1 and Person\_2 for each transaction paysa little amount of ether, but it is a necessary cost to buy the insurance policy.

Now suppose that Person\_1 did not give precedence to Person\_2 and caused an accident.

Since it is fault of Person\_1 while Person\_2 has no fault, the latter will be eligible to get a refund while the former should pay with his own money the damage.

**NOTE**: In Italy, although the regulation takes into account several variables, the new regulatory procedure implies that if there are no cases of disability exceeding 9%, in most case the direct reimbursement procedure is applied (whether or not there is agreement between the parties), that is, the person with no fault can apply to his own insurer and, if eligible, he will be refund from it. Then it will be up to his insurance company to assert its rights against the insurance company of the person who caused the accident, in order to get the money back.

In our simulation we apply these principles, therefore Person\_2 is eligible to get the refund from his own Insurer, while Person\_1 is not eligible for any refund.

Initialize PoliceReport: this smart contract ,as we have already said, is an official source and it is authorized by the police:

* *Compile* the PoliceReport
* Select a different account that this time represents the police and click on *Deploy*.
* Select the function *updateReport* and insert all the information:

address\_to\_push: address of the PersonalProfile of Person\_1 ; first\_vehicle: the license plates of Person\_1 car;

second\_vehicle: the license plates of Person\_2 car;

fault: the ID LICENSE/NAME of Person\_1, since in this case it’s his fault;

description: just a brief description of the accident

damage: the estimated amount of damage of the car ‘without fault’, let’s select for this case 2 ether.

* Now, if you click on *howManyReports* it returns the number 1 and if you select 0 in the function *viewReport* you can see all the information about that report.
* Repeat the same procedure, but now in the field address\_to\_push select the address of the PersonalProfile of Person\_2.

**NOTE**: In this way the police updates both the history of PersonalProfile of Person\_1 and Person\_2, even thought in this simulation we have not included the computation of the risk premium, information about ‘history’ would be very useful for the insurer in order to compute it.

The police pays a little amount of ether in order to update the history of both person, but we assume that are essential costs that falls under the Public Administration.

Go to the personalprofile of Person\_2 (select the right account!)

* Click on *howManyAccident*, as you can see now you have 1, and if in the function *getAccident* you write 0 (the index of the first accident) it returns all the information about the first car accident.
* In the function *getRefund* copy paste from  both the address of the InsurerContract and Person\_2 PersonalProfile address, in the index select the index of the accident for which you want to request a refund, in this case is 0, then click on transact.
* Now click on *MyBalance*, as you can see now it amounts at 2 Ether! That is the amount of the damage written in the PoliceReport.

Before of providing the reimbursement, the insurer has checked two conditions related to Person\_2 that should be True:

1. he must have paid the risk premium and therefore he should be insured
2. with regard to the accident, It’s not his fault.

If these two conditions are satisfied the Insurer automatically provide the refund.

* Now if you go in the InsurerContract and click on Balance, it is 8 ether not anymore 10 ehter, it means that the Insurer took the money from its *Blance* and paied its insured.
* In *RefundQueue* select 0 and click, it returns and address, copy and paste that address in the function *ShowRefundRequest* and you can observe the information about the refund that that person requested.

Go to the Personalprofile of Person\_1 (select the right account!):

* Click on *howManyAccident*, as you can see now you have 1, and if in the function *getAccident* you write 0 it returns all the information about the first car accident.
* In the function *getRefund* copy paste from  both the address of the InsurerContract and Person\_1 PersonalProfile address, in the index select the index of the accident for which you want to request a refund - in this case is 0 – click on transact.
* This time it’s different from the previous procedure, indeed nothing happened, because the Person\_1 – since it’s his fault – is not entitled to get a refund.
* If you click on *MyBalance* you have 0 and if you go to the InsurerContract you can observe that anything changed. The insurer Balance is still 8 ether.

NOTE: Both people in order to request the refund pays a little amount of ether, but we decided that the cost of notifying the accident is at their expense. It represents a sort of effort that even in practice they should have taken to notify the accident to their insurance company.

The simulation is over. As you can see everything works as in the real word but it is based on SmartContract and blockchain System that ensures to have a faster, more efficient and uncorruptible procedure.