Domain Claim Manual

Overview

The Domain Claims API provides the necessary services to administer and execute the functions of the Claim Factory, Claim and Policy Report Factory, Policy Report smarts contracts.

Claim Factory

The Claim Factory smart contract has the functions of creating a smart claim contract (Claim). A smart contract is created for each claim, recording its information on the blockchain. Each Claim smart contract can perform query and status change functions.

Policy Report Factory

The Policy Report Factory smart contract has the functions of creating smart contract reports (Policy Report). A smart contract is created for each report, recording its information on the blockchain. Each Policy Report smart contract can perform query and status change functions.

Smart Contract Administration

The first smart contract administrator is the contract owner. When the smart contract is published on the BFT network, the address of the publisher becomes the owner and has the ability add other admins.
To manage smart contracts it is necessary to have a BFT wallet. With the address of that wallet the contract owner can add that wallet as a smart contract administrator.

owner	The address of the wallet that publishes the smart contract on the network. This address is inserted in all smart contract roles at the time of publication.
Wallet	Wallets allow users to store BFT and interact with smart contracts on the BFT network.
address	The address is an unique identifiers derived from public keys. Private Key -> Public Key -> BFT Address
smart contract	A smart contract is the application code published on the BFT blockchain network. Executes the business rules defined for claims and reports.

Administrative functions

The roles in a smart contract define the roles performed by each entity. The claim and award smart contracts have the roles: Insurance Company and Admin.

Admin role

- Responsible for controlling the functions of **adding** and **removing** other admins.

API Method	Smart Contract Function
POST/role/admin	`addAdmin`

DELETE/role/admin	`removeAdmin`

- Responsible for controlling the functions of **adding** and **removing** insurance companies.

API Method	Smart Contract Function
POST/role/insurance-company	`addInsuranceCompany`
DELETE/role/insurance-company	`removeInsuranceCompany`

- Responsible for **pausing** / **activating** the smart contract when necessary

API Method	Smart Contract Function
POST/lifecycle/pause	`pause`
POST/lifecycle/unpause	`unpause`

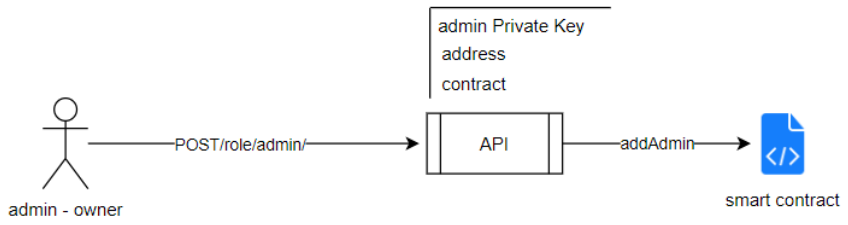
Insurance Company role

This role gives permission to create claims or reports, change the status of both and carry out consultations.

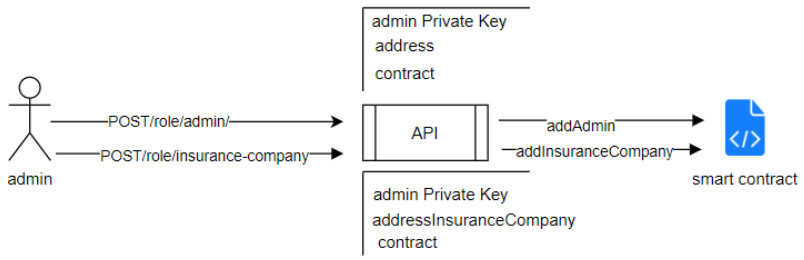
API Method	Smart Contract Function
POST/insurance-claim	`generateClaim`
PUT/insurance-claim	`updateStatusClaim`
POST/insurance-claim/consult-insuranceCompany-claims	`consultInsuranceCompanyClaims`
POST/insurance-claim/consult-claims	`consultApolice`; `consultVehicle`; `consultVehicleLicence`; `consultConductor`; `consultOcurrence`; `consultOcurrenceReport`; `consultThirdParty`; `consultClaimInformation`; `consultClaimInsurance`
POST/policy-report	`generateReport`
PUT/policy-report	`updateStatusReport`
POST/policy-report/consult-InsuranceCompany-policies-report	`consultInsuranceCompanyPoliciesReport`
POSt/policy-report/consult-policy-report	`consultReportInsuranceCompany`; `consultReportOcurrenceData`; `consultReportAssistanceData`; `consultReportTechnicalReport`; `consultReport`;

Create Claim and Reports

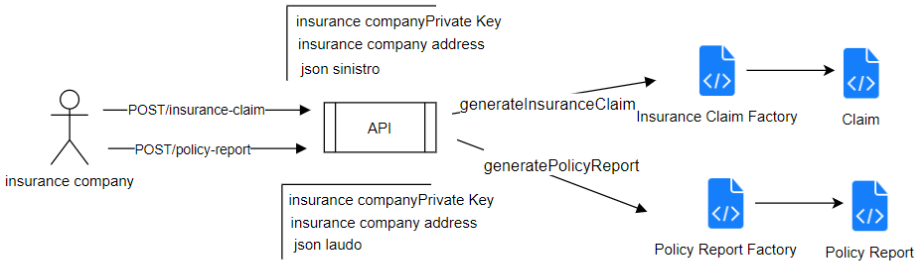
After the publication of the smart contract on the BFT network, the owner of the contract that was automatically added to the role admin, chooses who will be the administrator of that smart contract and adds the address to the role admin.
As the API Claim has both the Smart Claim Insurance contract and the Policy Report, the administrative functions have one more parameter to select which smart contract is being called. The parameter `contract` must be filled with: `insurance-claim` or `policy-report`



This admin role can control who is added or removed from both the admin role and the insurance company role. Then the admin adds the insurers to the role insurance company. Every insurer must have a BFT wallet in order for the address of that wallet to be added to the role insurance company by the admin.



Insurance companies, already with the permission, when sending the parameters for the creation of a claim or report, it is created and registered in the blockchain.



Run a method using swagger

On the swagger page you can select a GET / POST / PUT method and click the **try it out** button.
GET methods can be performed by entering the parameters in the required fields.
POST / PUT methods can be executed by editing the body with the necessary parameters.
After editing, click **Execute** and the method will be executed.

Method	Description
POST	The HTTP method `POST` it is used to create resources or to send information that will be processed. For example, creating a claim or report, consulting the claim and report information.
PUT	The HTTP method `PUT` is used to update an existing resource. For example, changing the status.
GET	The HTTP method `GET` is used to query existing resources. For example, query for roles.

Methods

Insurance Claim Factory

Insurance Claim Factory methods need to be signed by addresses that have the permission of insurers within the smart contract. This signature is called a private key. The smart contract admin first needs to add the insurer's address to the Insuranse Company role, from then on the insurer will be allowed to create claims and consult them using their private key.

POST/insurance-claim

Through this method it is possible to create and register a claim on the blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- insurance Company - Address of insurance companies participating in this claim.
- Fields:

Json exemplo

```
{

"privateKey":"A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2"
,
        "insuranceCompany":{
                "wallets":[
                        "0x09c15427Fed859ed46AFFB996bCd62f3b9180137"
                        ]
        },
        "apoliceData": {
                "apolice":"123456",
                "start_validity":"2020-01-01",
                "end_validity":"2020-01-02",
                "apolice_status":"1"
        },
```

```
        "claimInformation":{
                "claim": "123456",
                "claimStatus":"1"
        },
        "conductorData":{
                "name_complete":"client",
                "date_of_birth":"2020-01-01",
                "marital_status":"1",
                "cnpj_cpf":"11111111111",
                "gender":"1",
                "profession":"dev",
                "cnh":"1234567890",
                "category_cnh":"b",
                "date_validate_cnh":"2020-03-01"
        },
        "ocurrenceData":{
                "claim_number": "123456",
                "date_occurrence":"2020-10-01",
                "time_occurrence":"13:00",
                "place_occurrence":"place",
                "police_report":"678767",
                "protocol_police_report":"1",
                "conductor_guilty_occurrence":"1",
                "ocurrence_description": "111111",
                "victims":"1",
                "damage_victims":"1",
                "damage_vehicle":"1"
        },
        "thirdPartyData":{
                "involvement_3rd":"2",
                "how_many_involvement":"3",
                "licenses_plates": ["aaa1234"],
                "damage_caused":"none"
        },
        "vehicleData":{
                "vehicle_type":"auto",
        "maker": "ford",
        "model":"ka",
        "numer_slides":"5",
        "year_manufacture":"2010",
        "year_model":"2011",
        "license_plate":"1",
        "chassis":"11111A1111",
        "renavam":"11111N1111",
        "fuel":"1"
        }
}
```

body * required

object

(body)

Para gerar o sinistro é necessário preencher os campos

Edit Value | Model

```
},
"apoliceData": {
        "apolice":"123456",
        "start_validity":"2020-01-01",
        "end_validity":"2020-01-02",
        "apolice_status":"1"
},
"claimInformation":{
        "claim": "123456",
        "claimStatus":"1"
},
"conductorData":{
        "name_complete":"client",
        "date_of_birth":"2020-01-01",
        "marital_status":"1",
        "cnpj_cpf":"11111111111",
        "gender":"1",
        "profession":"dev",
        "cnh":"1234567890",
        "category_cnh":"b",
        "date_validate_cnh":"2020-03-01"
```

Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-7017",
  "link": "https://rinkeby.etherscan.io/tx/0x6f1ad3833a1dadd11542a9f506ef1ac07a11575eafefadd1e983b2ba03433c43"
}
```

POST/insurance-claim/consult-insuranceCompany-claims

Through this method it is possible to list all claims and insurance companies (claim owners) registered on the Blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- insurance Company - Address of insurance companies participating in this claim.

```
{
  "privateKey": "565RT090EC0B1DA54EC68E34234D9E3561F36E26B00A788E2D223CF478EDFE11",
  "insuranceCompanyAddress": "0xD8c9193b73d43c3d0BcBFf53DF7F35B27CE9fbB7"
}
```

Response

- data - lists *contract address insurance claim* and *address insurance company*

[[contract address insurance claim], [address insurance company]]

```
{
  "success": true,
  "code": "100-1000",
  "message": "sucess",
  "data": [
    [
      "0xD46faB4Eb466d5dB22B7F8E771Ec8eBC2555B9bF",
      "0x641ADDAE3bc7B508607C2F4DB4F3f6a74efd0E97",
      "0xA453b3785F32Be74eB66d01ED9d457ed524d2ffE",
      "0xA294de3c0e65d55E536082a1D088d26220aE6355",
      "0x45bc104Fb8ed013e85A74Ef0d7ebeFb5F6b49AEB",
      "0x303e9dd05b07D22e7DB8355AD972E6f42165E458"
    ],
    [
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137"
    ]
  ]
}
```


Insurance Claim

Insurance Claim methods must be signed by addresses that have the permission of insurers within the smart contract. This signature is called a private key. The smart contract admin first needs to add the insurer's address to the Insuranse Company role, from then on the insurer will be allowed to change the status and consult claims using his private key.POST/insurance-claim/consult-claims

Through this method it is possible to return all information of the Claim registered in the Blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- contract address - Address of the Insurance Claim smart contract that will have the information retrieved.


```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contractAddress": "0x303e9dd05b07D22e7DB8355AD972E6f42165E458"
}
```


Response

Example


```
{
  "success": true,
  "code": "100-1000",
  "message": "sucess",
  "data": {
```

```
"apoliceData": [
  {
    "_hex": "0x01e240"
  },
  "2020-01-01",
  "2020-01-02",
  {
    "_hex": "0x01"
  }
],
"claimInformation": [
  {
    "_hex": "0x01e240"
  },
  "23"
],
"conductorData": [
  "client",
  "2020-01-01",
  {
    "_hex": "0x01"
  },
  {
    "_hex": "0x02964619c7"
  },
  {
    "_hex": "0x01"
  },
  "dev",
  {
    "_hex": "0x499602d2"
  },
  "b",
  "2020-03-01"
],
"occurenceData": [
  {
    "_hex": "0x01e240"
  },
  "2020-10-01",
  "13:00",
  "place"
],
"occurenceReport": [
  {
    "_hex": "0x0a5b6f"
  },
  {
    "_hex": "0x01"
  },
  {
    "_hex": "0x01"
  },
  {
```

```
        "_hex": "0x01b207"
      },
      {
        "_hex": "0x01"
      },
      "1",
      "1"
    ],
    "thirdPartyData": [
      {
        "_hex": "0x02"
      },
      {
        "_hex": "0x03"
      },
      [
        "aaa1234"
      ],
      "none"
    ],
    "vehicleData": [
      "auto",
      "ford",
      "ka",
      {
        "_hex": "0x05"
      },
      {
        "_hex": "0x07da"
      },
      {
        "_hex": "0x07db"
      },
      {
        "_hex": "0x01"
      }
    ],
    "vehicleLicense": [
      "1",
      "11111A1111",
      "11111N1111"
    ]
  }
}
```

```
{
  "success": true,
  "code": "100-1000",
  "message": "sucess",
  "data": {
    "apoliceData": [
      {
        "_hex": "0x01e240"
      },
      "2020-01-01",
      "2020-01-02",
      {
        "_hex": "0x01"
      }
    ],
    "claimInformation": [
      {
        "_hex": "0x01e240"
      },
      "23"
    ],
    "conductorData": [
      "client",
      "2020-01-01",
      {
        "_hex": "0x01"
      },
```


PUT/insurance-claim

Through this method it is possible to change the value of the claim status on the blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- contract address - Address of the Insurance Claim smart contract that will have the information retrieved.
- status - New status


```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contractAddress": "0x303e9dd05b07D22e7DB8355AD972E6f42165E458",
  "status": "23"
}
```


Response

- link - Blockexplorer link to the transaction


```
{
  "success": true,
  "code": "100-3011",
  "link": "https://rinkeby.etherscan.io/tx/0xc1ce4f1a235d0fb054a0de0a4a90797dba5e348f46629086664705aff384e97d"
}
```


Policy Report Factory

Policy Report Factory methods must be signed by addresses that have the permission of insurers within the smart contract. This signature is called a private key. The smart contract admin first needs to add the insurer's address to the Insuranse Company role, from then on the insurer will be allowed to create reports and consult them using his private key.

POST/policy-report

Through this method it is possible to create and register a report on the blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- insurance Company - Addresses of insurance companies participating in this claim.
- Fields:

Json example

```
{

"privateKey":"A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2"
,
        "insuranceCompany": {
        "wallets": ["0x09c15427Fed859ed46AFFB996bCd62f3b9180137"]
    },
    "occurrenceData": {
        "claimNumber": "321123",
        "dateOcurrence": "1585084539000",
        "timeOcurrence": "1585084539000",
        "placeOcurrence": "Avenida Paulista,1392",
        "policeReport": "1",
        "protocolPoliceReport": "321344",
        "conductorGuiltyOccurrence": "1",
        "ocurrenceDescription": "3221333211",
        "victims": "1",
        "damageVictims": "Houve danos das vitimas descrição",
        "damageVehicle": "Relato dos danos causados no veiculo"
    },
    "assistanceData": {
        "cnpjCpf": "99999999999",
        "nameComplete": "Assistencia tecnica"
    },
        "technicalReport": {
        "reportNumber": "252415",
        "reportStatus": "2",
        "damageComponents": ["2", "1", "32"]
    },
    "manpower": "Descrição da mão de obra",
    "technicalAdvice": "Relato do parecer técnico"
}
```

```
{
        "privateKey":"A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
        "mutual": {
        "wallets": ["0x09c15427Fed859ed46AFFB996bCd62f3b9180137"]
    },
    "occurrenceData": {
        "claimNumber": "321123",
        "dateOcurrence": "1585084539000",
        "timeOcurrence": "1585084539000",
        "placeOcurrence": "Avenida Paulista,1392",
        "policeReport": "1",
        "protocolPoliceReport": "321344",
        "conductorGuiltyOccurrence": "1",
        "ocurrenceDescription": "3221333211",
        "victims": "1",
        "damageVictims": "Houve danos das vitimas descrição",
        "damageVehicle": "Relato dos danos causados no veiculo"
    },
    "assistanceData": {
        "cnpjCpf": "99999999999",
```


Response

- link - Blockexplorer link for the transaction


```
{
  "success": true,
  "code": "100-2100",
  "link": "https://rinkeby.etherscan.io/tx/0x21df8028610fe0459f0bcfe33996d4dce2564cba4aa83f48a799c813b92c9791"
}
```


POST/policy-report/consult-InsuranceCompany-policies-report

Through this method it is possible to list all the reports and insurers (owners of the reports) registered in the Blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- insurance Company - Addresses of insurance companies participating in this claim.


```
{
  "privateKey": "5654C091WC0B1DA54EC68E34234D9E3561F36E26B00A788E2D223CF478EDFE11",
  "insuranceCompanyAddress": "0xD8c9193b73d43c3d0BcBFf53DF7F35B27CE9fbB7"
}
```


Response

- data - lista *contract address policy report* e *address insurance company*

[[contract address policy report], [address insurance company]]

```
{
  "success": true,
  "code": "100-1000",
  "message": "sucess",
  "data": [
    [
      "0xD46faB4Eb466d5dB22B7F8E771Ec8eBC2555B9bF",
      "0x641ADDAE3bc7B508607C2F4DB4F3f6a74efd0E97",
      "0xA453b3785F32Be74eB66d01ED9d457ed524d2ffE",
      "0xA294de3c0e65d55E536082a1D088d26220aE6355",
      "0x45bc104Fb8ed013e85A74Ef0d7ebeFb5F6b49AEB",
      "0x303e9dd05b07D22e7DB8355AD972E6f42165E458"
    ],
    [
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137",
      "0x09c15427Fed859ed46AFFB996bCd62f3b9180137"
    ]
  ]
}
```


Policy Report

Policy Report methods must be signed by addresses that have the permission of insurers within the smart contract. This signature is called a private key. The smart contract admin first needs to add the insurer's address to the Insurance Company role, from then on the insurer will be allowed to change the status and consult the reports using his private key.

POST/policy-report/consult-policy-report

Through this method it is possible to return all information from the reports registered in the Blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- contract address - policy report contract address that wants to retrieve the information..


```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contractAddress": "0x303e9dd05b07D22e7DB8355AD972E6f42165E458"
}
```


Response

Example


```
{
  "success": true,
  "code": "100-1000",
  "result": {
    "report": [
```

```
      "Descrição da mão de obra",
      "Relato do parecer técnico"
    ],
    "insuranceCompany": [
      "0xD8c9193b73d43c3d0BcBFf53DF7F35B27CE9fbB7"
    ],
    "occurence": [
      "123456",
      {
        "_hex": "0x01710e66a078"
      },
      {
        "_hex": "0x01710e66a078"
      },
      "Avenida Paulista,1392",
      "1",
      "321344",
      "1",
      "3221333211",
      "1",
      "Houve danos das vitimas descrição",
      "Relato dos danos causados no veiculo"
    ],
    "assistanceData": [
      "99999999999",
      "Assistencia tecnica"
    ],
    "technicalReport": [
      "252415",
      "2",
      [
        "2",
        "1",
        "32"
      ]
    ]
  }
}
```


PUT/policy-report

Through this method it is possible to change the value of the status of the report on the blockchain. You must be on the role Insurance Company.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance Company". This private key must be from an insurance company
- contract address - Policy Report smart contract address with information to be retrieve
- status - New status

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contractAddress": "0x303e9dd05b07D22e7DB8355AD972E6f42165E458",
  "status": "23"
}
```

Response

- link - Blockexplorer link for the transaction

```
{
  "success": true,
  "code": "100-3021",
  "link": "https://rinkeby.etherscan.io/tx/0xeb0ca9411fa2d554682bb166fa50305404a33a170b4eb0004476a6737c563b2d"
}
```

Lifecycle

The so-called lifecycle methods must be signed by an address that has admin permission within the smart contract. This subscription is called a private key. A smart contract admin first needs to add the address to the role admin, from there they will be allowed to pause or activate the Insurance Claim Factory and Policy Report Factory smart contracts using their private key.

POST/lifecycle/pause

Through this method, it is possible to pause the Insurance Claim and Policy Report smart contracts. You must be in the Admin role.

Request

- private Key - Whoever carries out this transaction needs to be in the "admin" role. This private key must be from an administrator of the selected smart contract
- contract - put the name of the contract*: insurance-claim* or *policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contract": "policy-report"
}
```

Response

```
{
  "success": true,
  "code": "",
  "description": "insurance-claim paused"
}
```

POST/lifecycle/unpause

Through this method it is possible to activate the Insurance Claims and Policy Report smart contracts. You must be in the Admin role.

Request

- private Key - Whoever carries out this transaction needs to be in the "admin" role. This private key must be from an administrator of the selected smart contract
- contract - Use contract name: *insurance-claim* or *policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "contract": "policy-report"
}
```

Response

```
{
  "success": true,
  "code": "",
  "description": "policy-report unpaused"
}
```

GET/lifecycle/paused/{contract}

Through this method, it is possible to check the status of the Policy Factory smart contract.

Request

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

Name	Description
contract * required *(path)*	insurance-claim

Response

```
{
  "success": true,
  "code": "",
  "description": "insurance-claim is paused",
  "data": false
}
```

Roles

The methods called roles need to be signed by address that have admin permission within the smart contract. This signature is called a private key. A smart contract admin first needs to add the address in the role admin, from there they will be allowed to add or remove addresses from the Smart Claim Insurance Factory and Policy Report Factory contracts using their private key.

GET/role/admin

Through this method, it is possible to check if an address is an administrator in the Insurance Claim Factory or Policy Report Factory smart contract.

Request

- contract - *Use contract name: insurance-claim or policy-report*
- address - Wallet address to be consulted

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

contract * required *(path)*	insurance-claim
address * required string *(path)*	0x71fC0e20F2DA5853CE9174A54093f5b29

Response

- isAdmin is true if the address is in the role admin, otherwise it is false.
- contract - name of the smart contract requested.

```
{
  "success": true,
  "code": "100-7017",
  "result": {
    "isAdmin": false,
    "contract": "policy-report"
  }
}
```

POST/role/admin

Through this method, it is possible to add an administrator to the Insurance Claim Factory or Policy Report Factory smart contract. You must be in the Admin role.

Request

- private Key - Whoever carries out this transaction needs to be in the "admin" role. This private key must be from an administrator of the selected smart contract
- address - Wallet address that will be added as an administrator
- contract - *Use contract name: insurance-claim or policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "address": "0x71fC0e20F2DA5853CE9174A54093f5b2918cCfa2",
  "contract": "policy-report"
}
```

Response

- link - Blockexplorer link for the transaction ⚠️

DELETE/role/admin

Through this method, it is possible to remove an administrator in the Smart Claim Insurance Factory or Policy Report Factory smart contract. You must be in the Admin role.

Request

- private Key - Whoever carries out this transaction needs to be in the "admin" role. This private key must be from an administrator of the selected smart contract
- address - Wallet address that will be added as an administrator
- contract - *Use contract name: insurance-claim or policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "address": "0x71fC0e20F2DA5853CE9174A54093f5b2918cCfa2",
  "contract": "policy-report"
}
```

Response

- link - Blockexplorer link for the transaction ⚠️

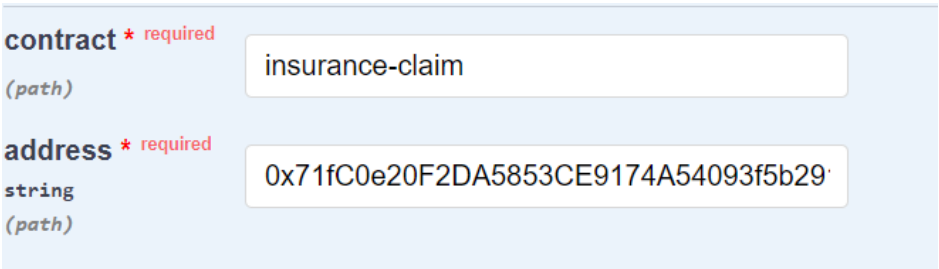
GET/role/insurance-company

Through this method it is possible to check if an address is in the insurance role in the smart contract Insurance Claim or Policy Report.

Request

- contract - *Use contract name: insurance-claim or policy-report*
- address - Wallet address to be consulted

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`



Response

- isInsuranceCompany is true if the address is in the role insurance company, otherwise it is false.
- contract - name of the smart contract to be consulted.

```
{
  "success": true,
  "code": "100-7017",
  "result": {
    "isInsuranceCompany": true,
    "contract": "insurance-claim"
  }
}
```

POST/role/insurance-company

Through this method it is possible to add an insurance company to the Smart Contract Insurance Claim Factory or Policy Report Factory. You must be in the Admin role.

Request

- private Key - Whoever carries out this transaction must be in the role "insurance-company". This private key must be from an administrator of the selected smart contract
- address - Wallet address that will be added as an administrator
- contract - *Use contract name: insurance-claim or policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "address": "0x71fC0e20F2DA5853CE9174A54093f5b2918cCfa2",
  "contract": "policy-report"
}
```

Response

- link - Blockexplorer link for the transaction ⚠️

DELETE/role/insurance-company

Through this method it is possible to remove an insurer from the Smart Contract Insurance Claim Factory or Policy Report Factory. You must be in the Admin role.

Requisição

- private Key - Whoever carries out this transaction must be in the role "insurance-company". This private key must be from an administrator of the selected smart contract
- address - Wallet address that will be added as an administrator
- contract - Use contract name*: insurance-claim or policy-report*

The parameter `contract` must be filled with: `insurance-claim` or `policy-report`

```
{
  "privateKey": "A9AC29CBEB110215AE6D5AF5D8731A848160A3A7DABEF198B832A38D4B4385E2",
  "address": "0x71fC0e20F2DA5853CE9174A54093f5b2918cCfa2",
  "contract": "policy-report"
}
```

Response

- link - Blockexplorer link for the transaction ⚠️

Campos sinistro e laudo

˅ insurence claim

Category	Variable	Description	Type	Example
privateKey		Private key of Signer (msg.sender)	`string`	A9AC29CBEB110215AE6D5AF5D
mutual				
	`wallets`	Numbers of wallets of each insurance company belonging to the claim.	`string`	[0x09c15427Fed859ed46AFFB996b
apoliceData				
	`apolice`	Policy identificatio n (internal)	`uint256`	123456

	`start_validity`	Effective date	`uint256`	01/01/2020
	`end_validity`	Effective end date	`uint256`	02/01/2020
	`apolice_status`	Policy status	`uint256`	1
claimInformation				
	`claim`	Claim identificatio n (internal)	`string`	123456
	`claimStatus`	Claim status	`uint256`	1
conductorData				
	name_complete	Driver name	`string`	client
	date_of_birth	Date of birth	`string`	01/01/2020
	marital_status	marital status	`uint256`	1
	cnpj_cpf	CPF / CNPJ	`string`	11111111111
	gender	Gender	`uint256`	1
	profession	Profession	`uint256`	dev
	cnh	Driver license	`uint256`	1234567890
	category_cnh	First license emission	`string`	b
	date_validate_cnh	License expiration date	`string`	01/03/2020
ocurrenceData				
	claim_number	Claim number (internal)	`uint256`	123456
	date_occurrence	Date of occurrence of the claim	`string`	01/10/2020

	time_occurrence	Time of occurrence of the claim	`string`	13
	place_occurrence	Place of occurrence of the claim	`string`	place
	police_report	Police report	`uint256`	678767
	protocol_police_report	Police report protocol	`uint256`	1
	conductor_guilty_occurrence	Driver at the time of occurrence	`uint256`	1
	ocurrence_description	Occurrence description	`string`	adc
	victims	Number of victims	`uint256`	1
	damage_victims	Description of the damage caused to the victims	`uint256`	1
	damage_vehicle	Description of the damage caused to the vehicle	`uint256`	1
thirdPartyData				
	involvement_3rd		`uint256`	2
	how_many_involvement		`uint256`	3
	licenses_plates		`string[]`	[aaa1234]
	damage_caused		`string`	none
vehicleData				
	vehicle_type	Vehicle type	`uint256`	1
	maker	Maker	`uint256`	4
	model	Model	`uint256`	19

	numer_slides		uint256	5
	year_manufacture	Year of manufacture	uint256	2010
	year_model	Model Year	uint256	2011
	license_plate	License plate	string	1
	chassis	Chassis	string	11111A1111
	renavam	Renavan	string	11111N1111
	fuel	Fuel type	uint256	1

policy report

Categoria	Variable	Description	Type	Example
privateKey		Private key of Signer (msg.sender)	string	A9AC29CBEB110215AE6D5
mutual				
	wallets	Numbers of wallets of each insurance company belonging to the claim.	string	[0x09c15427Fed859ed46AFF
occurrenceData				
	claimNumber	Claim identificatio n (internal)	string	321123
	dateOcurrence	Date of occurrence	uint256	1585084539000
	timeOcurrence	Time of occurrence	uint256	1585084539000
	placeOcurrence	Date of occurrence	string	Avenida Paulista1392
	policeReport	Police report	uint256	121
	protocolPoliceReport	Police report protocol	uint256	321344

	conductorGuiltyOccurrence	Driver at the time of occurrence	`uint256`	1
	ocurrenceDescription	Occurrence description	`string`	dsad11
	victims	Number of victims	`uint256`	1
	damageVictims	Description of the damage caused to the victims	`string`	Houve danos das vitimas descr
	damageVehicle	Description of the damage caused to the vehicle	`string`	Relato dos danos causados no
assistanceData				
	cnpjCpf	CPF/CNPJ	`string`	99999999999
	nameComplete	Full name of technical assistance	`string`	Assistencia tecnica
technicalReport				
	reportNumber	Report identification (internal)	`string`	252415
	reportStatus	Report status	`uint256`	2
	damageComponents	Damaged components	`string[]`	[2, 1, 32]
	manpower	Description of labor	`string`	Descrição da mão de obra
	technicalAdvice	Technical opinion report	`string`	Relato do parecer técnico