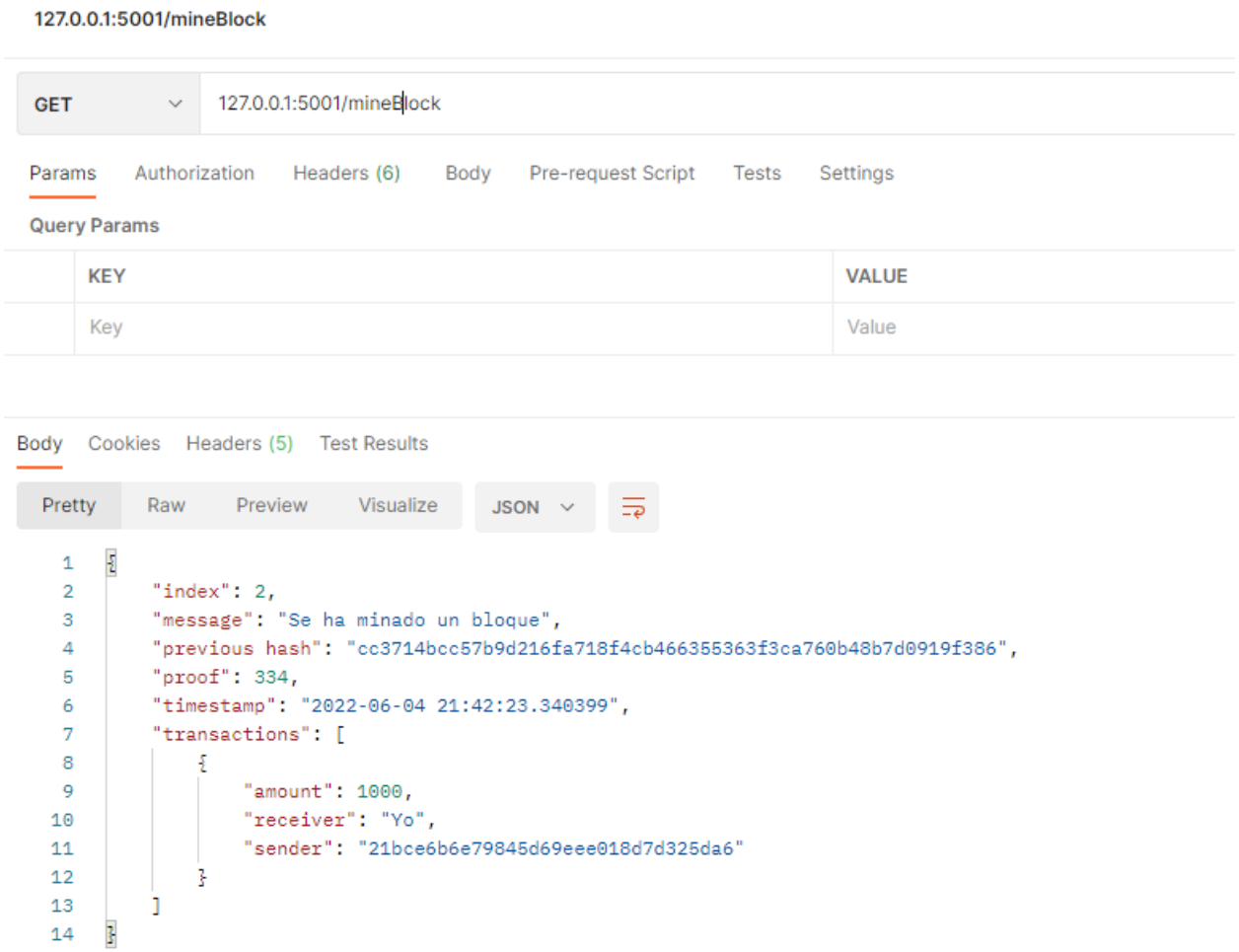
Reporte Scamcoin

Francisco Fuentes - Tamara Badilla

mineBlock



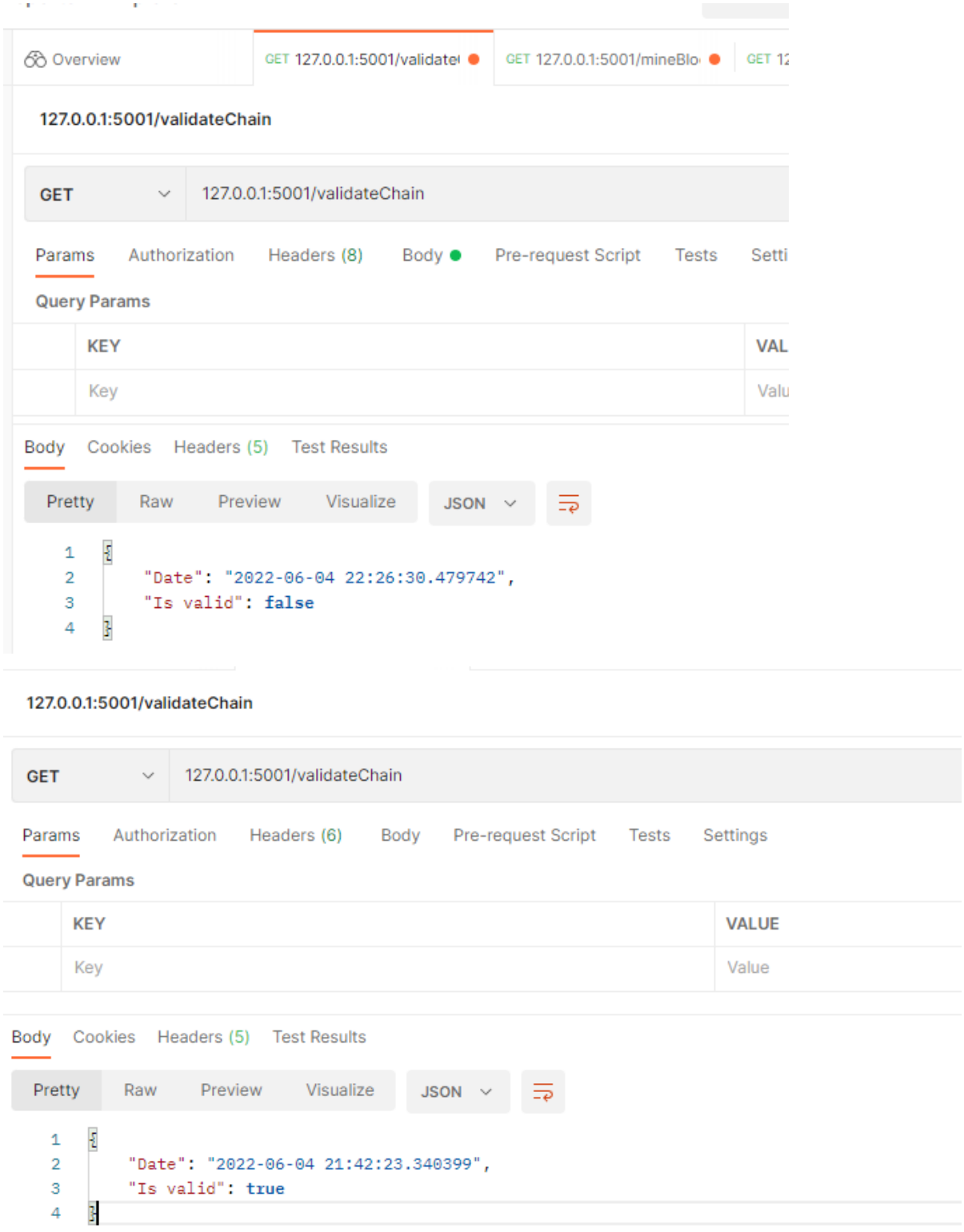
Desde la perspectiva de postman, se accede al request de GET mineBlock, y recibe la respuesta dada por el programa en formato JSON.

```
francizs@Francizs:/mnt/c/Users/fraci/OneDrive/Escritorio/Carpetas/SistDist/Unidad 2/re
po/scamcoin-3$ python3 nodeone.py
 * Serving Flask app 'nodeone' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5001 (Press CTRL+C to quit)
127.0.0.1 - - [04/Jun/2022 21:42:16] "GET /mineblock HTTP/1.1" 404 -
127.0.0.1 - - [04/Jun/2022 21:42:23] "GET /mineBlock HTTP/1.1" 200 -
```

Desde la perspectiva del node, envía la respuesta al request, junto con un código 200.

---

validateChain



Desde la perspectiva de postman, se genera el request de GET validateChain, y recibe en un JSON si la cadena era o no válida.

```
francizs@Francizs:/mnt/c/Users/fraci/OneDrive/Escritorio/Carpetas/SistDist/Unidad 2/repo/scamcoin-3$ python3 nodeone.py
 * Serving Flask app 'nodeone' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5001 (Press CTRL+C to quit)
127.0.0.1 - - [04/Jun/2022 21:42:16] "GET /mineblock HTTP/1.1" 404 -
127.0.0.1 - - [04/Jun/2022 21:42:23] "GET /mineBlock HTTP/1.1" 200 -
127.0.0.1 - - [04/Jun/2022 21:44:19] "GET /validateChain HTTP/1.1" 200 -
```

Desde la perspectiva del node, envía la respuesta al request, junto con un código 200.

---

addTransaction

127.0.0.1:5001/addTransaction

POST 127.0.0.1:5001/addTransaction

Params Authorization Headers (8) Body Pre-request Script

none form-data x-www-form-urlencoded raw binary

```
{
    "amount": 1000,
    "receiver": "21bce6b6e79845d69eee018d7d325da6",
    "sender": "Yo"
}
```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

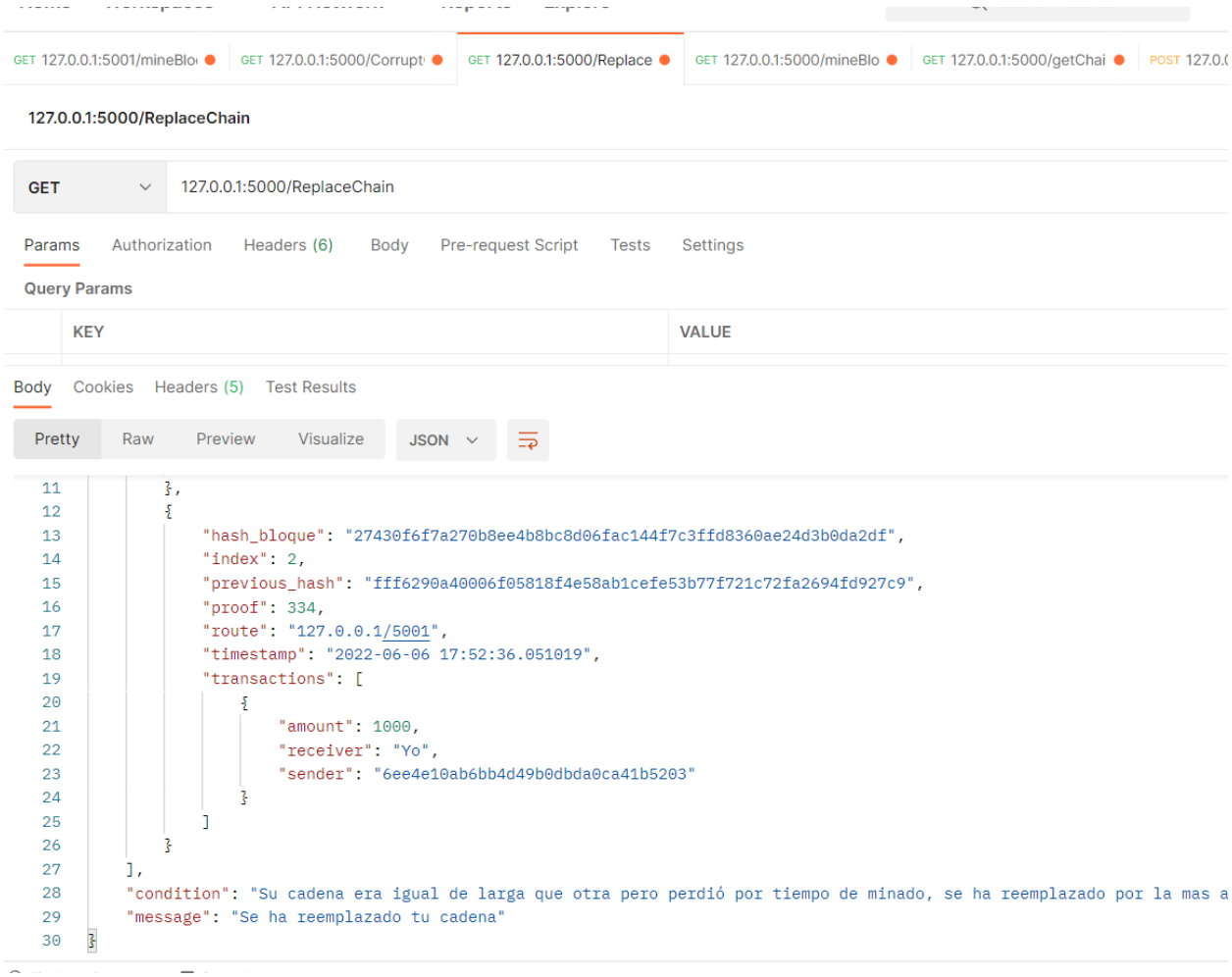
```
{
    "message": "Transaccion añadida al bloque 4"
}
```

```
127.0.0.1 - - [04/Jun/2022 21:51:46] "POST /addTransaction HTTP/1.1" 201 -
```

Desde Postman se envía un JSON tipo {“amount”: int, “receiver”: String, “sender”: String}, desde el código se recibe ese JSON, y se agrega la transacción, junto con un mensaje de respuesta.

ReplaceChain

----- misma longitud, distinto tiempo de minado



------ No es la cadena más larga

127.0.0.1:5000/ReplaceChain

GET 127.0.0.1:5000/ReplaceChain

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Query Params

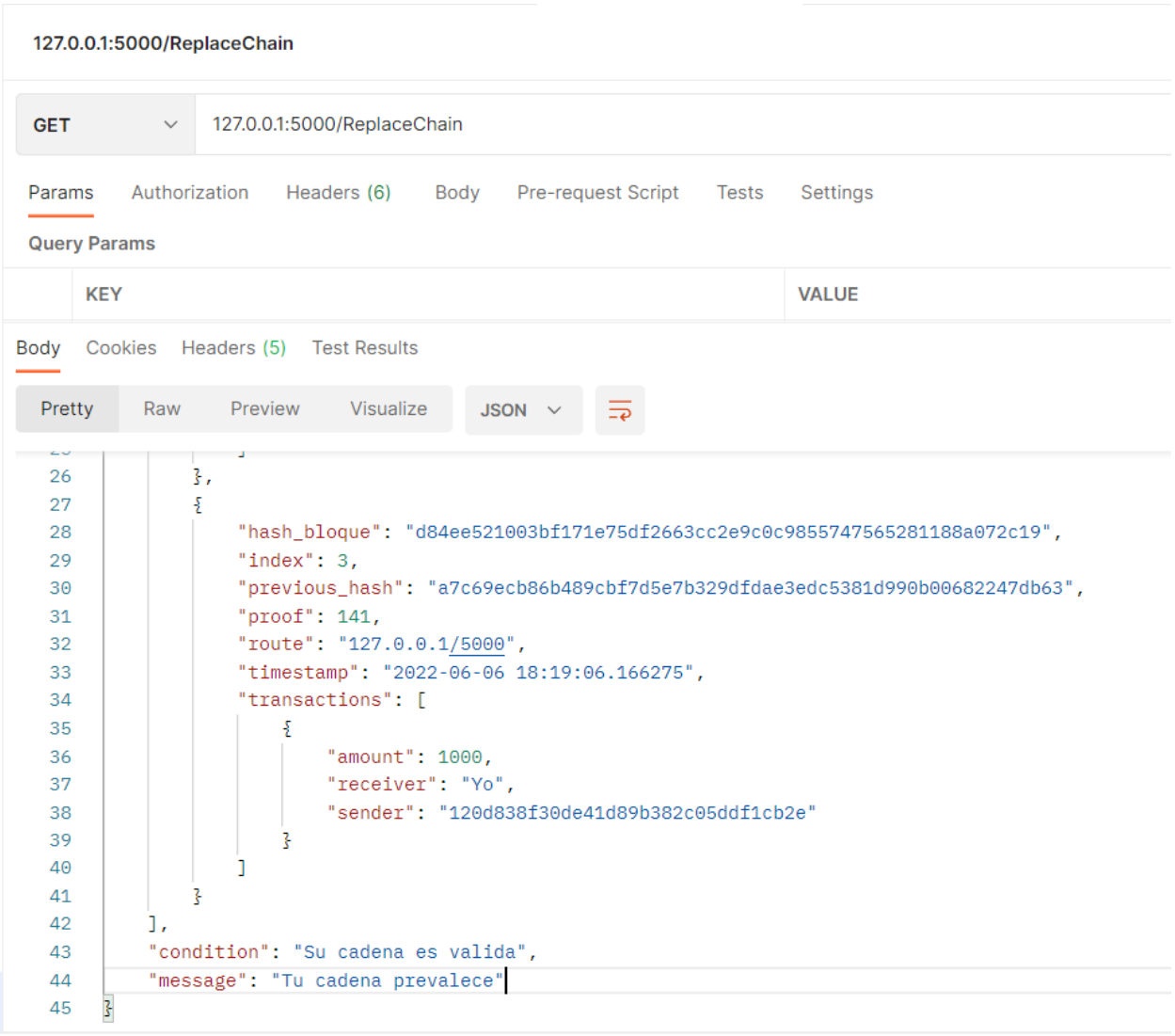
KEY	VALUE

Body Cookies Headers (5) Test Results

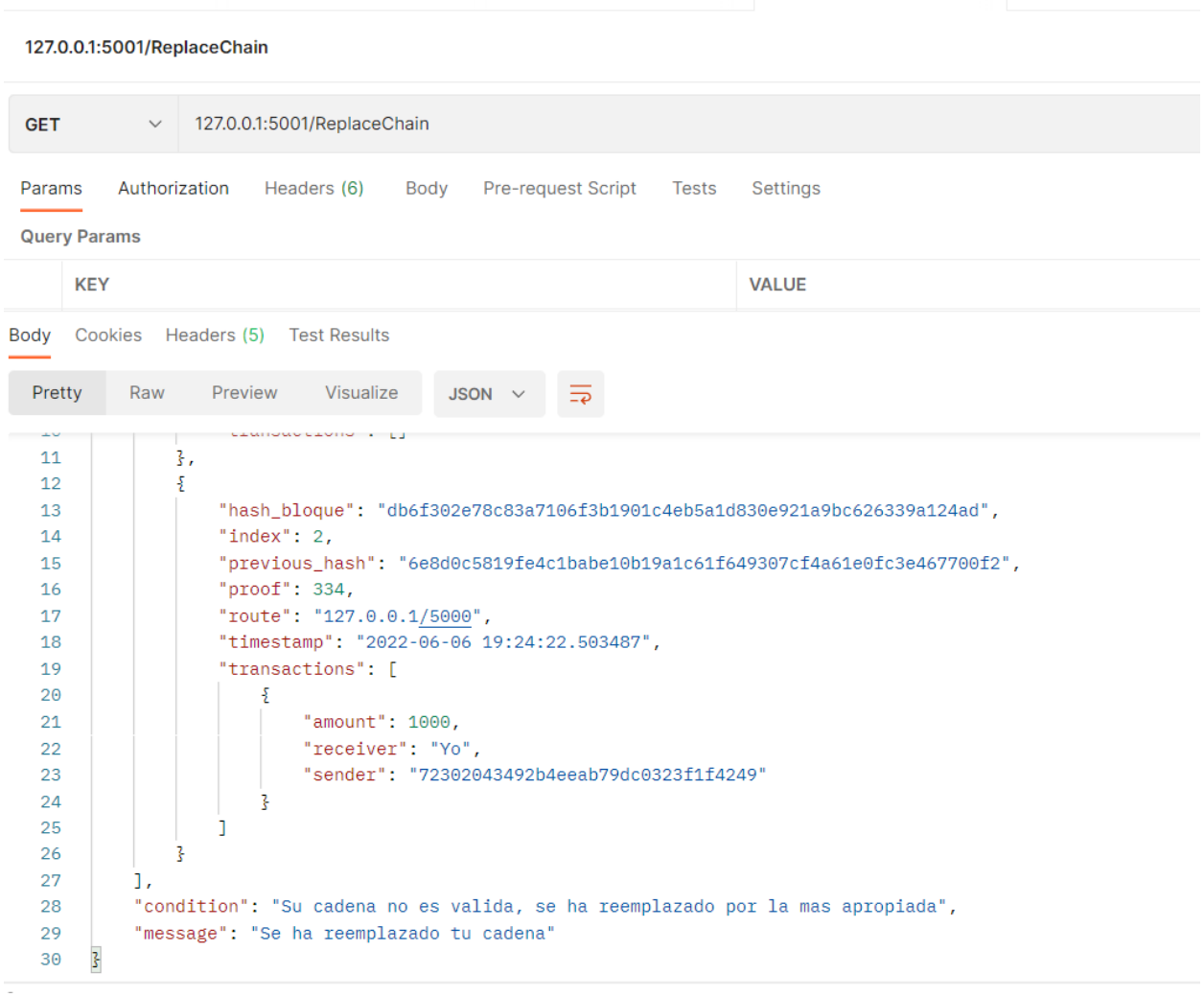
Pretty Raw Preview Visualize JSON


```
        },
        {
            "hash_bloque": "92002212d26af182e32c416f9c85fc018ccf751dc70a03184976a3a1",
            "index": 4,
            "previous_hash": "ae0710938dda0f57ae2a2c523465a2a2fdb6aa716ff537ab3662a001",
            "proof": 64,
            "route": "127.0.0.1/5001",
            "timestamp": "2022-06-06 17:54:24.706306",
            "transactions": [
                {
                    "amount": 1000,
                    "receiver": "Yo",
                    "sender": "72eed513875f4b008e01c7c7c29eef92"
                }
            ]
        }
    ],
    "condition": "Su cadena no es la mas larga, se ha reemplazado por la mas larga y valida",
    "message": "Se ha reemplazado tu cadena"
}
```


----- Cadena prevalece



---- cadena invalida



Desde Postman se envía un GET replaceChain, después de esto, en el código se comparan las cadenas, y devuelve, según el caso, un mensaje diferente.

CorruptChain

127.0.0.1:5001/CorruptChain

GET 127.0.0.1:5001/CorruptChain

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

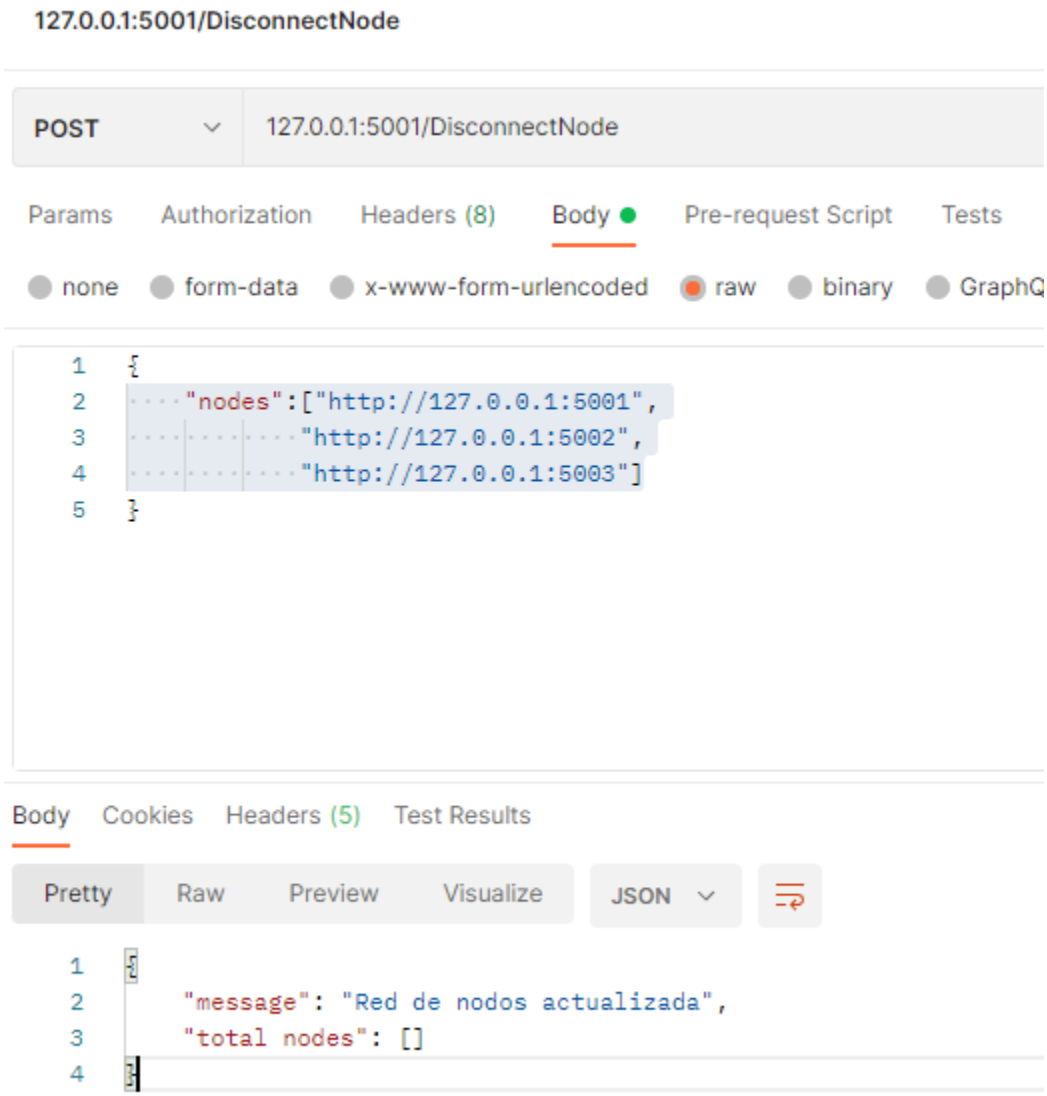
Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON


```
{
    "block": {
        "hash_bloque": "b393cc3ab99a4759068abfa3455937e593359d9be9c4b4666c2719b1",
        "index": 1,
        "previous_hash": "0",
        "proof": 1,
        "route": "127.0.0.1/5003",
        "timestamp": "2022-06-04 22:03:11.386081",
        "transactions": []
    },
    "chain": [
        {
            "hash_bloque": "b393cc3ab99a4759068abfa3455937e593359d9be9c4b4666c2719b1",
            "index": 1,
            "previous_hash": "0",
            "proof": 1,
            "route": "127.0.0.1/5003",
            "timestamp": "2022-06-04 22:03:11.386081",
            "transactions": []
        }
    ],
    "lenght of chain": 1,
    "mensaje": "Cadena corrupta"
}
```


Desde Postman se llama a GET corruptChain para corromper una cadena y probar el validateChain.

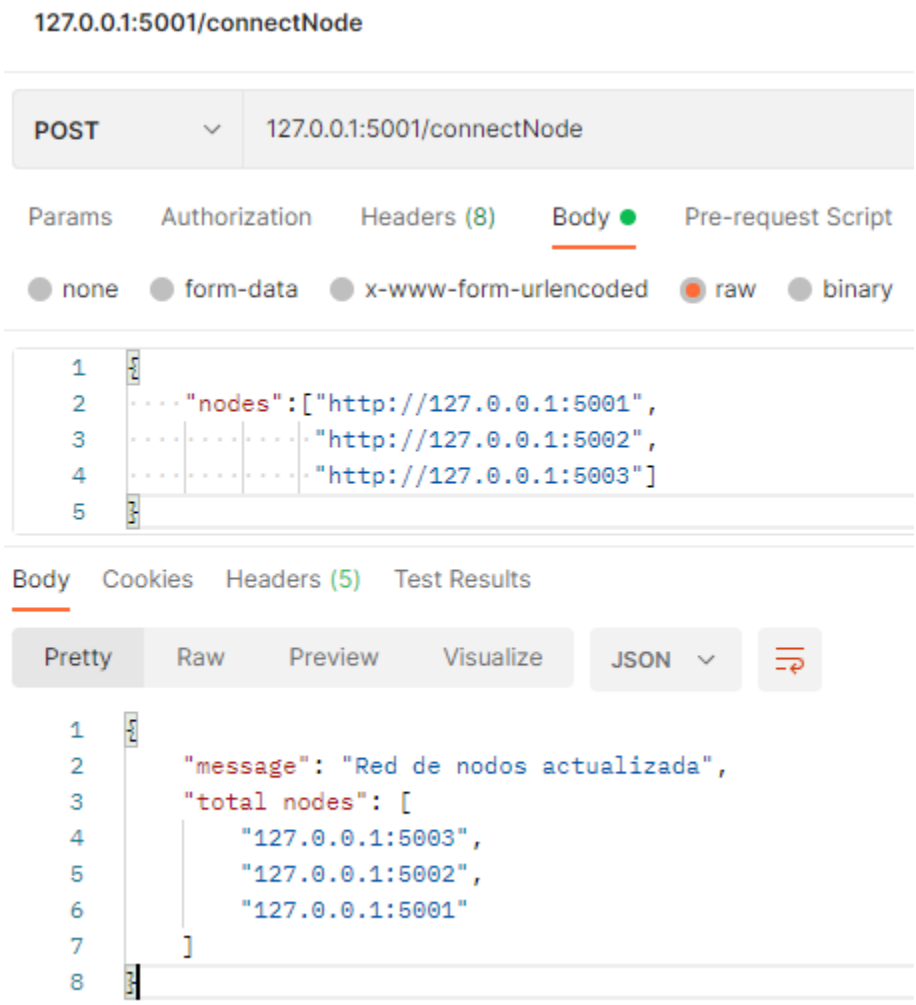
DisconnectNode



Desde Postman se envían los nodos que se quieran desconectar en un JSON tipo {“nodes: lista de nodos} en un POST a connectNode, el código luego los desconecta y envía un mensaje junto con la cantidad de nodos conectados.

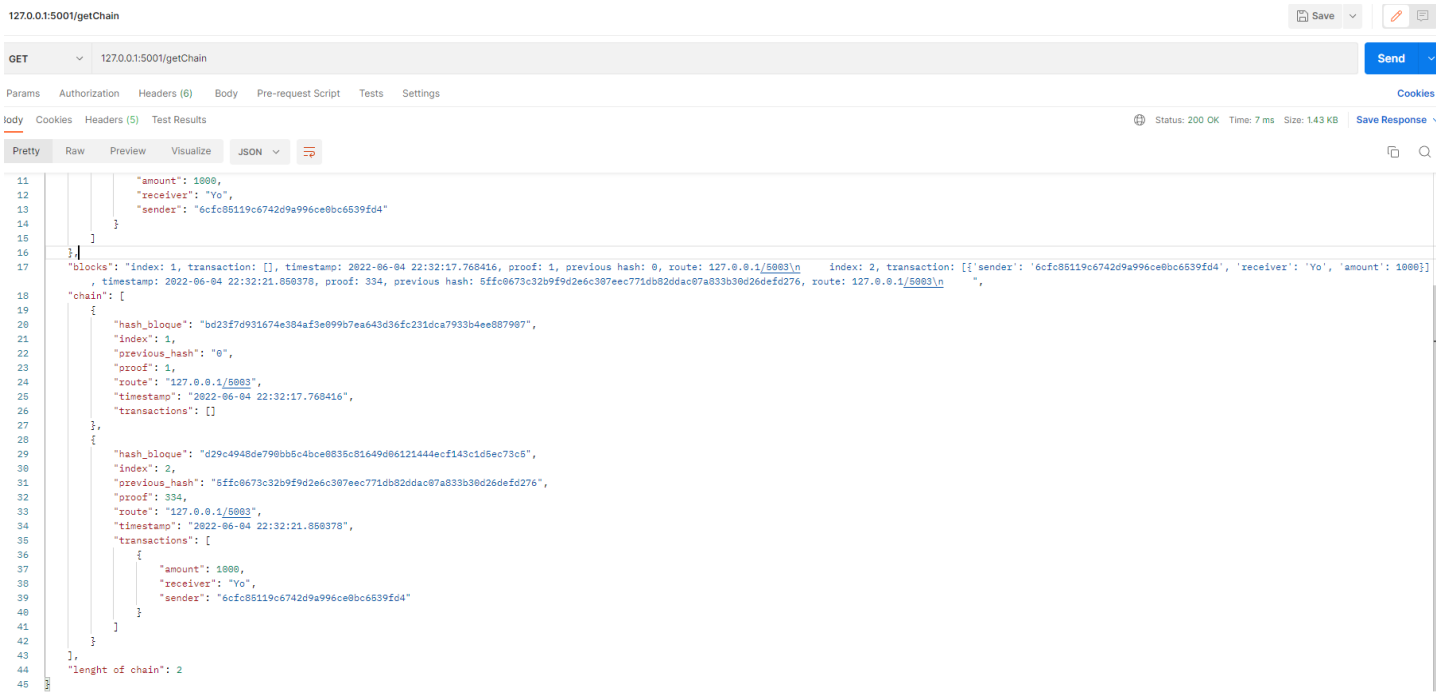
---

connectNode



Desde Postman se envían los nodos que se quieran conectar en un JSON tipo {“nodes: lista de nodos} en un POST a connectNode, el código luego los conecta y envía un mensaje junto con la cantidad de nodos conectados.

getChain



Desde Postman se realiza un GET a getChain, donde se recibe información sobre la cadena y los bloques en sí.
