

mineBlock

127.0.0.1:5001/mineBlock

GET 127.0.0.1:5001/mineBlock

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

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```

1  {
2    "index": 2,
3    "message": "Se ha minado un bloque",
4    "previous hash": "cc3714bcc57b9d216fa718f4cb466355363f3ca760b48b7d0919f386",
5    "proof": 334,
6    "timestamp": "2022-06-04 21:42:23.340399",
7    "transactions": [
8      {
9        "amount": 1000,
10       "receiver": "Yo",
11       "sender": "21bce6b6e79845d69eee018d7d325da6"
12     }
13   ]
14 }
```

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

Overview

GET 127.0.0.1:5001/validateChain

GET 127.0.0.1:5001/mineBlock

GET 127.0.0.1:5001/validateChain

127.0.0.1:5001/validateChain

GET

127.0.0.1:5001/validateChain

ParamsAuthorizationHeaders (8)Body●Pre-request ScriptTestsSettings

Query Params

KEY	VAL
Key	Value

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

```
1 {
2   "Date": "2022-06-04 22:26:30.479742",
3   "Is valid": false
4 }
```

127.0.0.1:5001/validateChain

GET

127.0.0.1:5001/validateChain

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE
Key	Value

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

```
1 {
2   "Date": "2022-06-04 21:42:23.340399",
3   "Is valid": true
4 }
```

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/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 -
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 C

```

1  {
2    "amount": 1000,
3    "receiver": "21bce6b6e79845d69eee018d7d325da6",
4    "sender": "Yo"
5  }

```

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```

1  {
2    "message": "Transaccion añadida al bloque 4"
3  }

```

```

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

The screenshot shows a Postman interface with a GET request to `127.0.0.1:5000/ReplaceChain`. The response is a JSON object with the following structure:

```
11  },
12  {
13    "hash_bloque": "27430f6f7a270b8ee4b8bc8d06fac144f7c3ffd8360ae24d3b0da2df",
14    "index": 2,
15    "previous_hash": "fff6290a40006f05818f4e58ab1cefe53b77f721c72fa2694fd927c9",
16    "proof": 334,
17    "route": "127.0.0.1/5001",
18    "timestamp": "2022-06-06 17:52:36.051019",
19    "transactions": [
20      {
21        "amount": 1000,
22        "receiver": "Yo",
23        "sender": "6ee4e10ab6bb4d49b0dbda0ca41b5203"
24      }
25    ]
26  },
27 ],
28 "condition": "Su cadena era igual de larga que otra pero perdió por tiempo de minado, se ha reemplazado por la mas a
29 "message": "Se ha reemplazado tu cadena"
30
```

----- 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



```
40
41   },
42   {
43     "hash_bloque": "92002212d26af182e32c416f9c85fc018ccf751dc70a03184976a3a1",
44     "index": 4,
45     "previous_hash": "ae0710938dda0f57ae2a2c523465a2a2fdb6aa716ff537ab3662a001",
46     "proof": 64,
47     "route": "127.0.0.1/5001",
48     "timestamp": "2022-06-06 17:54:24.706306",
49     "transactions": [
50       {
51         "amount": 1000,
52         "receiver": "Yo",
53         "sender": "72eed513875f4b008e01c7c7c29eef92"
54       }
55     ]
56   }
57 ],
58 "condition": "Su cadena no es la mas larga, se ha reemplazado por la mas larga y valida",
59 "message": "Se ha reemplazado tu cadena"
60 }
```

----- Cadena prevalece

127.0.0.1:5000/ReplaceChain

GET127.0.0.1:5000/ReplaceChain

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettings

Query Params

KEY	VALUE
-----	-------

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

```
26      },
27      {
28        "hash_bloque": "d84ee521003bf171e75df2663cc2e9c0c9855747565281188a072c19",
29        "index": 3,
30        "previous_hash": "a7c69ecb86b489cbf7d5e7b329dfdae3edc5381d990b00682247db63",
31        "proof": 141,
32        "route": "127.0.0.1/5000",
33        "timestamp": "2022-06-06 18:19:06.166275",
34        "transactions": [
35          {
36            "amount": 1000,
37            "receiver": "Yo",
38            "sender": "120d838f30de41d89b382c05ddf1cb2e"
39          }
40        ]
41      }
42    ],
43    "condition": "Su cadena es valida",
44    "message": "Tu cadena prevalece"
45  }
```

---- cadena invalida

127.0.0.1:5001/ReplaceChain

GET 127.0.0.1:5001/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



```
11  },
12  {
13      "hash_bloque": "db6f302e78c83a7106f3b1901c4eb5a1d830e921a9bc626339a124ad",
14      "index": 2,
15      "previous_hash": "6e8d0c5819fe4c1babe10b19a1c61f649307cf4a61e0fc3e467700f2",
16      "proof": 334,
17      "route": "127.0.0.1/5000",
18      "timestamp": "2022-06-06 19:24:22.503487",
19      "transactions": [
20          {
21              "amount": 1000,
22              "receiver": "Yo",
23              "sender": "72302043492b4eeab79dc0323f1f4249"
24          }
25      ]
26  },
27 ],
28 "condition": "Su cadena no es valida, se ha reemplazado por la mas apropiada",
29 "message": "Se ha reemplazado tu cadena"
30
```

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



```
1  {
2    "block": {
3      "hash_bloque": "b393cc3ab99a4759068abfa3455937e593359d9be9c4b4666c2719b1",
4      "index": 1,
5      "previous_hash": "0",
6      "proof": 1,
7      "route": "127.0.0.1/5003",
8      "timestamp": "2022-06-04 22:03:11.386081",
9      "transactions": []
10   },
11   "chain": [
12     {
13       "hash_bloque": "b393cc3ab99a4759068abfa3455937e593359d9be9c4b4666c2719b1",
14       "index": 1,
15       "previous_hash": "0",
16       "proof": 1,
17       "route": "127.0.0.1/5003",
18       "timestamp": "2022-06-04 22:03:11.386081",
19       "transactions": []
20     }
21   ],
22   "length of chain": 1,
23   "mensaje": "Cadena corrupta"
24 }
```

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

DisconnectNode

127.0.0.1:5001/DisconnectNode

POST

127.0.0.1:5001/DisconnectNode

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

```
1  {
2    ... "nodes":["http://127.0.0.1:5001",
3    ... "http://127.0.0.1:5002",
4    ... "http://127.0.0.1:5003"]
5  }
```

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

JSON

```
1  {
2    "message": "Red de nodos actualizada",
3    "total nodes": []
4  }
```

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

127.0.0.1:5001/connectNode

POST

127.0.0.1:5001/connectNode

Params

Authorization

Headers (8)

Body

Pre-request Script

none

form-data

x-www-form-urlencoded

raw

binary

1

[

2

"nodes": ["http://127.0.0.1:5001",

3

"http://127.0.0.1:5002",

4

"http://127.0.0.1:5003"]

5

]

Body

Cookies

Headers (5)

Test Results

Pretty

Raw

Preview

Visualize

JSON

1

[

2

"message": "Red de nodos actualizada",

3

"total nodes": [

4

"127.0.0.1:5003",

5

"127.0.0.1:5002",

6

"127.0.0.1:5001"

7

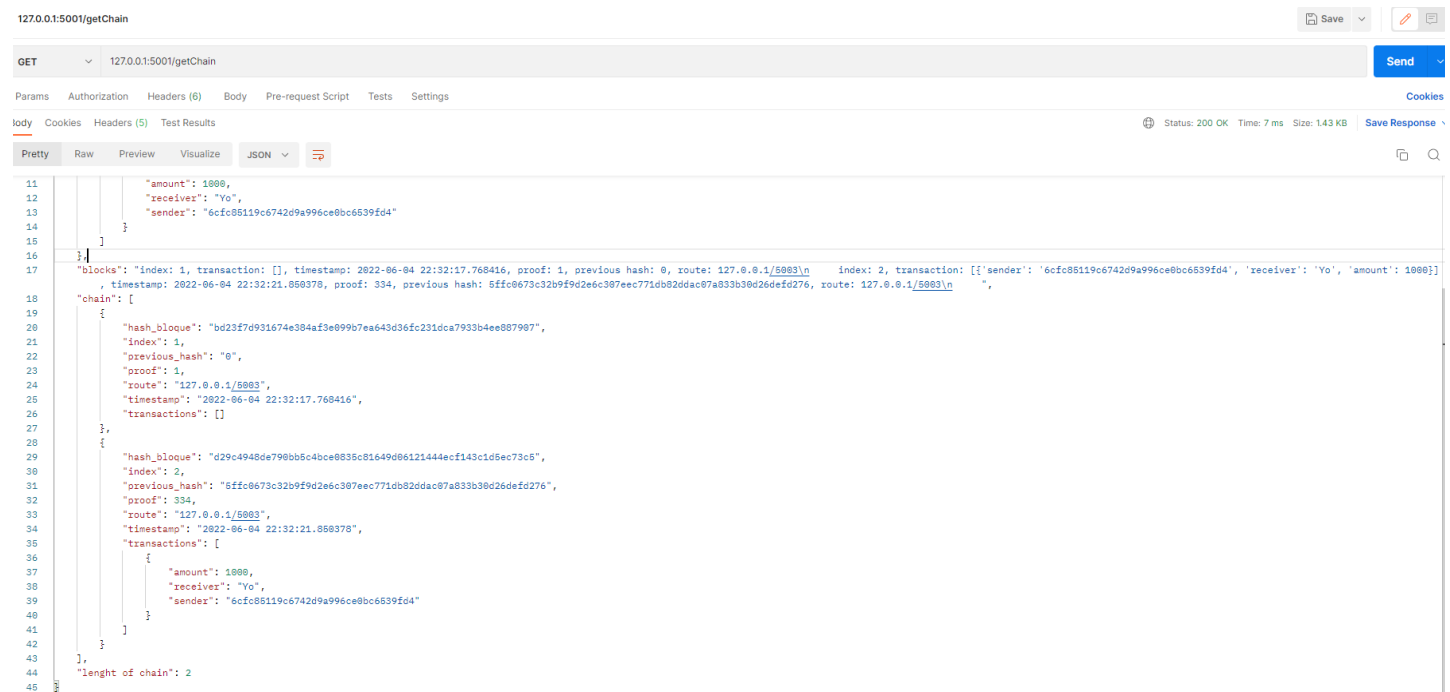
]

8

]

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.