BLOCKCHAIN – Building the basic structure of the Blockchain

Link for the repository: https://github.com/CRag-01/IOT-Blockchain-RiceSupplyChain/tree/main/Building%20a%20Blockchain

Description and Screenshots:

Files:

i. **blockchain.py –** Defines the basic structure of the blockchain
ii. **app.py –** Flask framework for server connections and end points.
(Server running on PORT 5000)
iii. **notes.txt -** A glimpse about the terms and concepts used in the implementation.

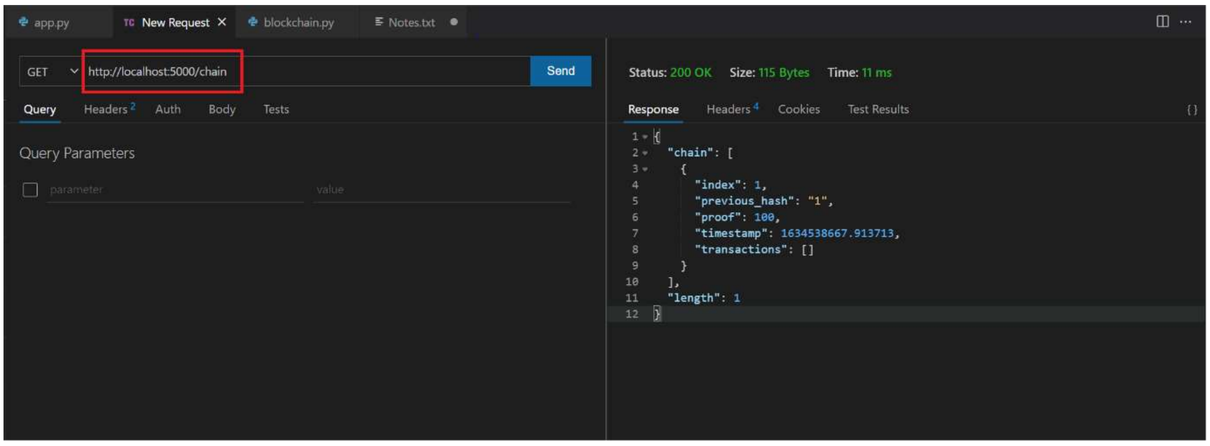
Here we have defined 5 end-points which are –

i. http://localhost:5000/mine **- to mine a new block – GET Method**
ii. http://localhost:5000/transactions/new **- to append a new transaction – POST Method**
iii. http://localhost:5000/chain **- to view our whole block chain**
iv. http://localhost:5000/nodes/register **- to register a new node running on a different machine/port and registering with our main network**
v. http://localhost:5000/nodes/resolve **- works using the consensus algorithm to resolve the longest chain problem, makes the longest chain as the current new blockchain.**

Testing with POSTMAN API Testing (Used the VS Code extension here -):

As soon as the server is started a new block is mined, which is the first block of the chain – namely the *GENESIS* Block.

http://localhost:5000/chain **- To check the current block chain**



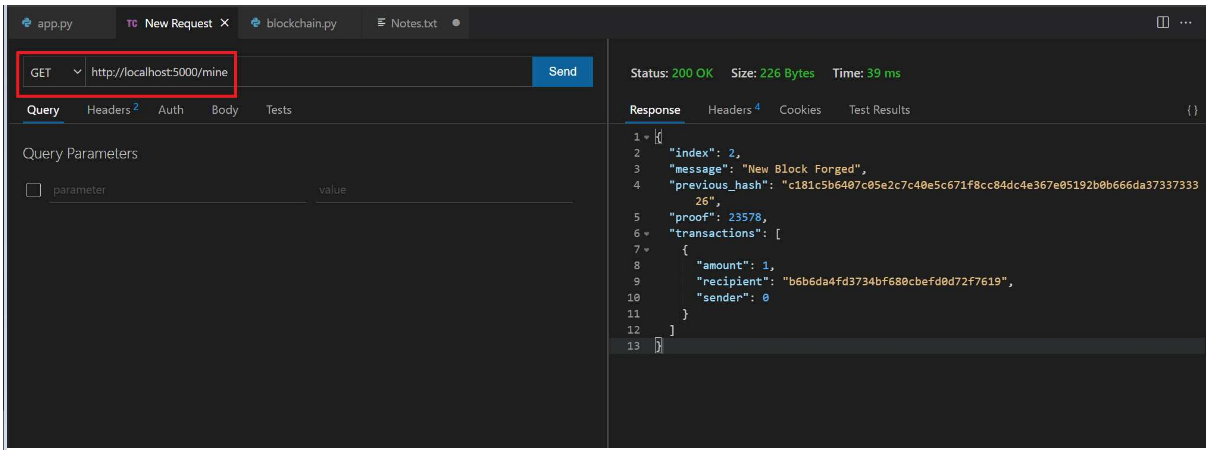
GET http://localhost:5000/chain

Status: 200 OK Size: 115 Bytes Time: 11 ms

```
{
  "chain": [
    {
      "index": 1,
      "previous_hash": "1",
      "proof": 100,
      "timestamp": 1634538667.913713,
      "transactions": []
    }
  ],
  "length": 1
}
```

Currently the chain contains only the genesis block which was automatically created when the server was started.

Now, lets mine a new block using the **GET** method of http://localhost:5000/mine endpoint -

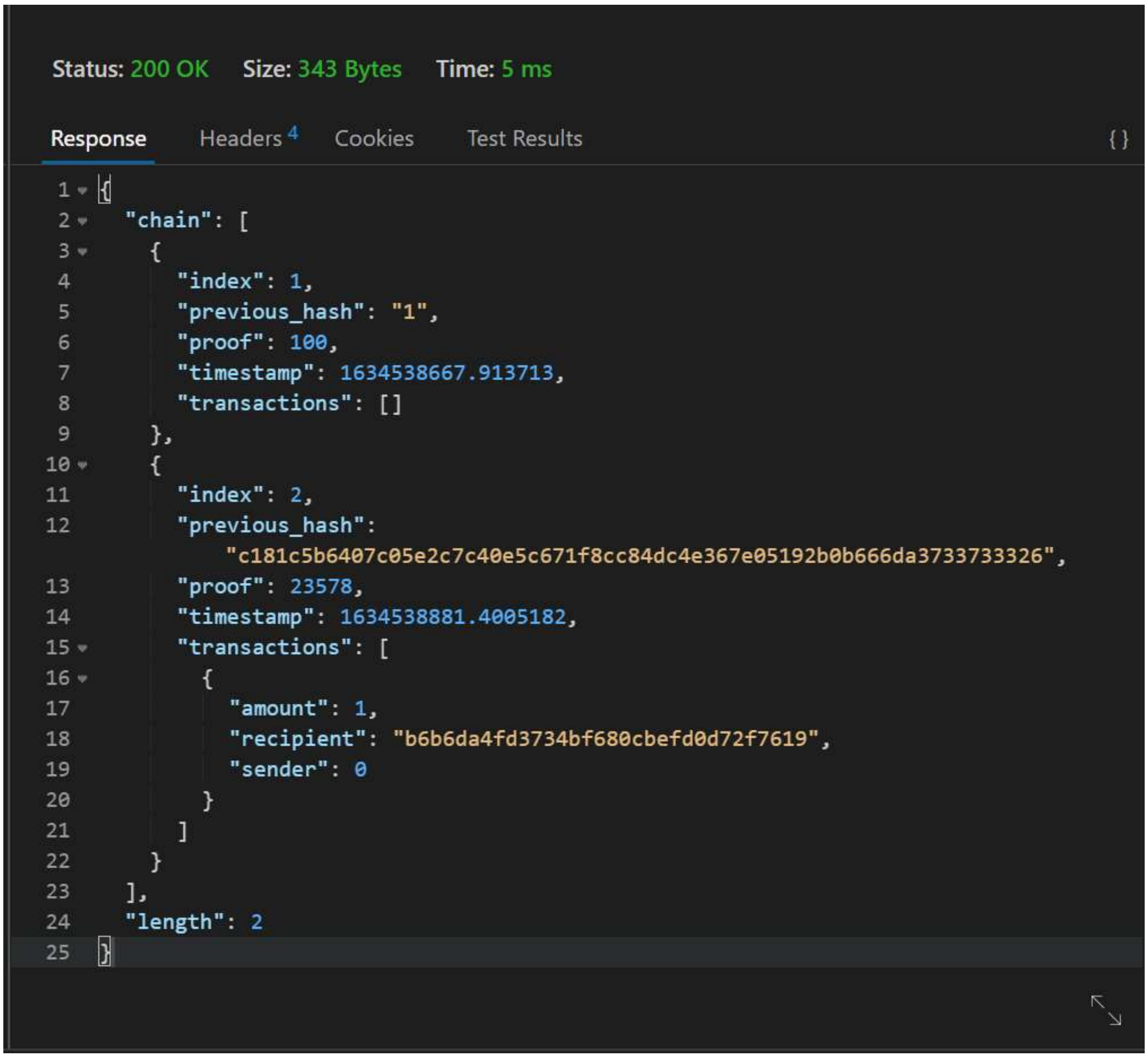


GET http://localhost:5000/mine

Status: 200 OK Size: 226 Bytes Time: 39 ms

```
{
  "index": 2,
  "message": "New Block Forged",
  "previous_hash": "c181c5b6407c05e2c7c40e5c671f8cc84dc4e367e05192b0b666da37337333
     26",
  "proof": 23578,
  "transactions": [
    {
      "amount": 1,
      "recipient": "b6b6da4fd3734bf680cbefd0d72f7619",
      "sender": 0
    }
  ]
}
```

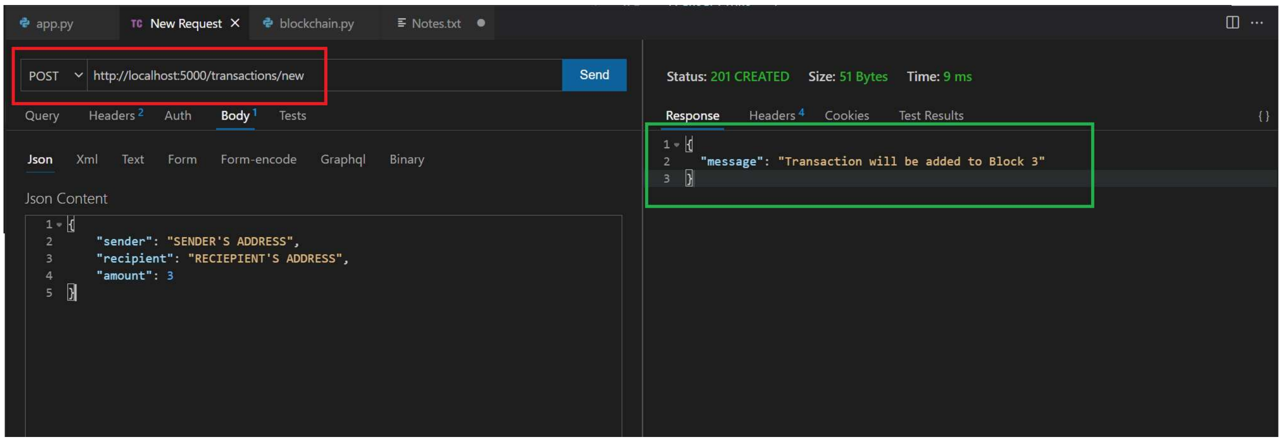
This newly mined block is now appended to the chain –



Status: 200 OK Size: 343 Bytes Time: 5 ms

```
{
  "chain": [
    {
      "index": 1,
      "previous_hash": "1",
      "proof": 100,
      "timestamp": 1634538667.913713,
      "transactions": []
    },
    {
      "index": 2,
      "previous_hash":
          "c181c5b6407c05e2c7c40e5c671f8cc84dc4e367e05192b0b666da3733733326",
      "proof": 23578,
      "timestamp": 1634538881.4005182,
      "transactions": [
        {
          "amount": 1,
          "recipient": "b6b6da4fd3734bf680cbefd0d72f7619",
          "sender": 0
        }
      ]
    }
  ],
  "length": 2
}
```

Adding a new transaction using - http://localhost:5000/transactions/new endpoint -



```
POST  http://localhost:5000/transactions/new
```

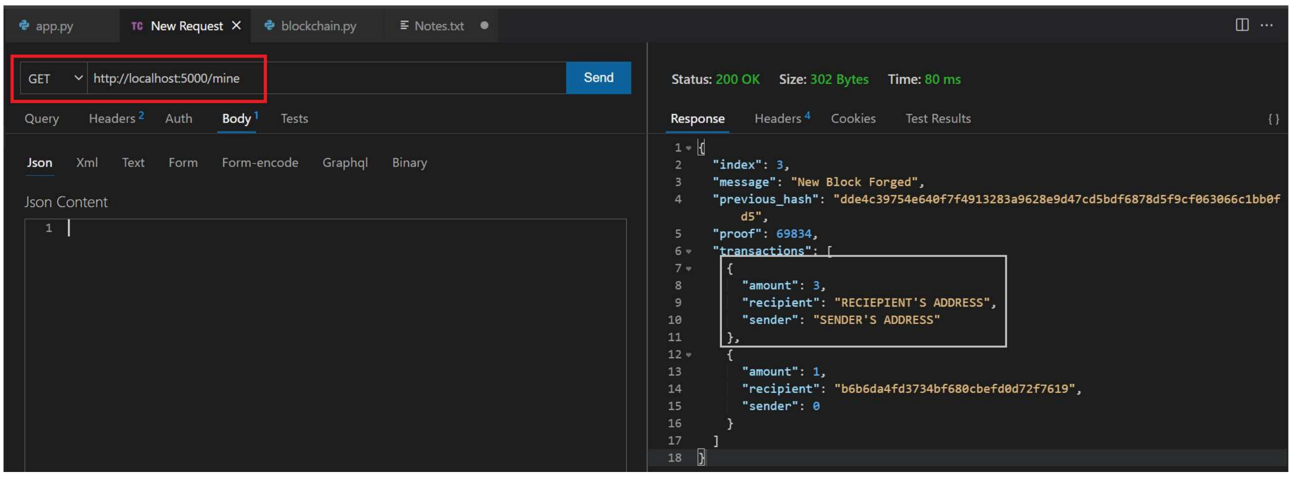
Json Content

```
{
    "sender": "SENDER'S ADDRESS",
    "recipient": "RECIEPIENT'S ADDRESS",
    "amount": 3
}
```

Status: 201 CREATED Size: 51 Bytes Time: 9 ms

```
{
  "message": "Transaction will be added to Block 3"
}
```

These transactions would be added to the new blocked which will be mined further –



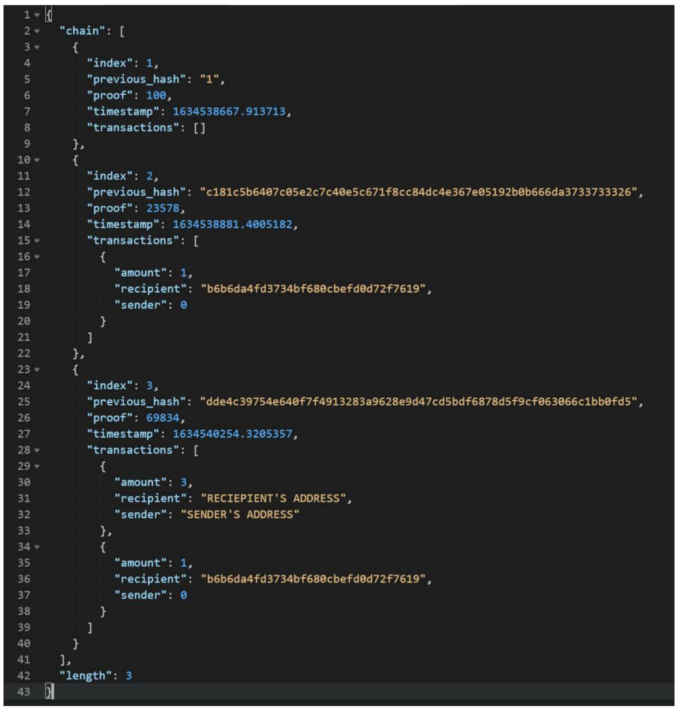
```
GET  http://localhost:5000/mine
```

Json Content

Status: 200 OK Size: 302 Bytes Time: 80 ms

```
{
  "index": 3,
  "message": "New Block Forged",
  "previous_hash": "dde4c39754e640f7f4913283a9628e9d47cd5bdf6878d5f9cf063066c1bb0f
     d5",
  "proof": 69834,
  "transactions": [
    {
      "amount": 3,
      "recipient": "RECIEPIENT'S ADDRESS",
      "sender": "SENDER'S ADDRESS"
    },
    {
      "amount": 1,
      "recipient": "b6b6da4fd3734bf680cbefd0d72f7619",
      "sender": 0
    }
  ]
}
```

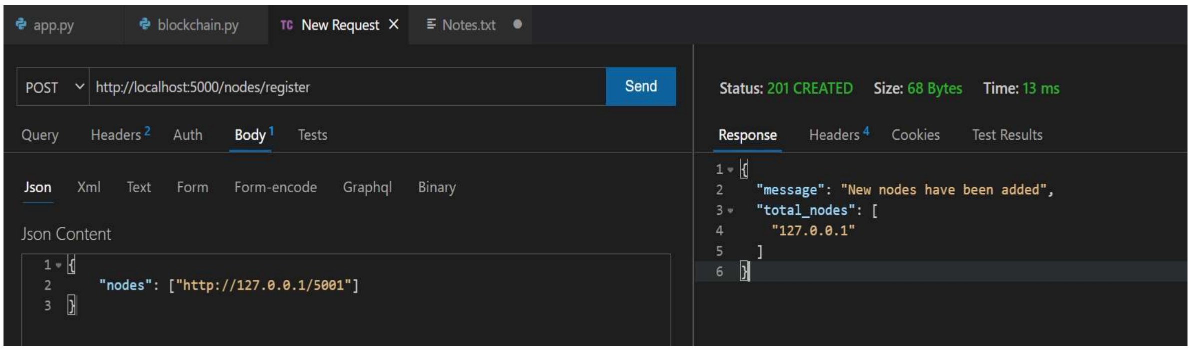
And the chain will look like –



```
{
  "chain": [
    {
      "index": 1,
      "previous_hash": "1",
      "proof": 100,
      "timestamp": 1634538667.913713,
      "transactions": []
    },
    {
      "index": 2,
      "previous_hash": "c181c5b6407c05e2c7c40e5c671f8cc84dc4e367e05192b0b666da3733733326",
      "proof": 23578,
      "timestamp": 1634538881.4005182,
      "transactions": [
        {
          "amount": 1,
          "recipient": "b6b6da4fd3734bf680cbefd0d72f7619",
          "sender": 0
        }
      ]
    },
    {
      "index": 3,
      "previous_hash": "dde4c39754e640f7f4913283a9628e9d47cd5bdf6878d5f9cf063066c1bb0fd5",
      "proof": 69834,
      "timestamp": 1634540254.3205357,
      "transactions": [
        {
          "amount": 3,
          "recipient": "RECIEPIENT'S ADDRESS",
          "sender": "SENDER'S ADDRESS"
        },
        {
          "amount": 1,
          "recipient": "b6b6da4fd3734bf680cbefd0d72f7619",
          "sender": 0
        }
      ]
    }
  ],
  "length": 3
}
```

Now consider, **two nodes, node1 and node2** running on **port 5000 and port 5001.**

- Registering node2 in node1 using - http://localhost:5000/nodes/register



POST http://localhost:5000/nodes/register Send

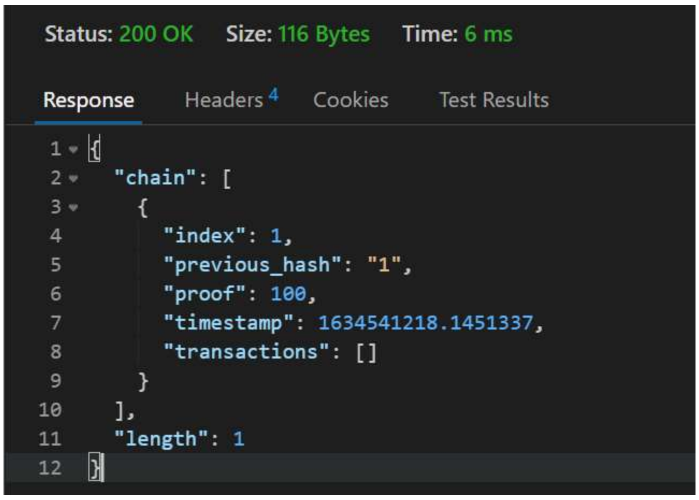
```
{
    "nodes": ["http://127.0.0.1/5001"]
}
```

Status: 201 CREATED Size: 68 Bytes Time: 13 ms

```
{
  "message": "New nodes have been added",
  "total_nodes": [
    "127.0.0.1"
  ]
}
```

Now Node2 running on port 5001 is also a part of this network.

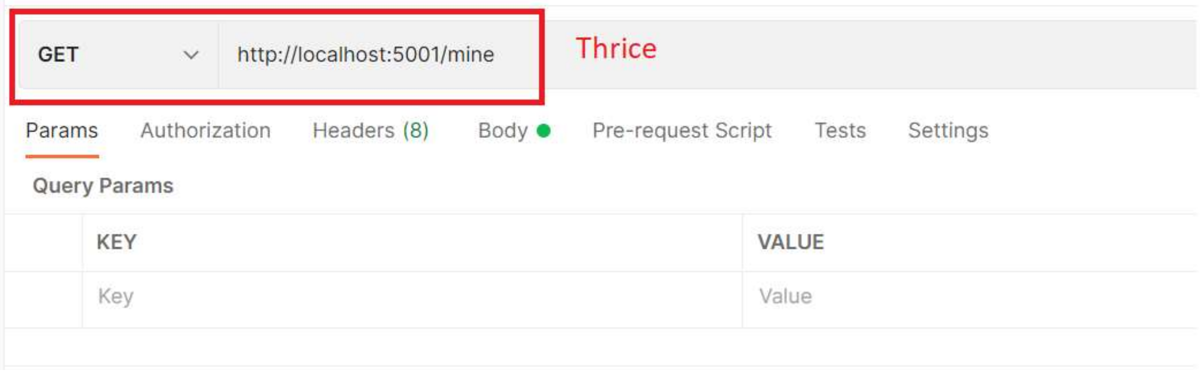
As we've *restarted* the server, node 1's chain just has one block - the genesis block.



Status: 200 OK Size: 116 Bytes Time: 6 ms

```
{
  "chain": [
    {
      "index": 1,
      "previous_hash": "1",
      "proof": 100,
      "timestamp": 1634541218.1451337,
      "transactions": []
    }
  ],
  "length": 1
}
```

On **Node2(5001 Port),** Let us mine a few blocks such that, the length of the chain in node2 is greater than that in node1.



GET http://localhost:5001/mine

Thrice

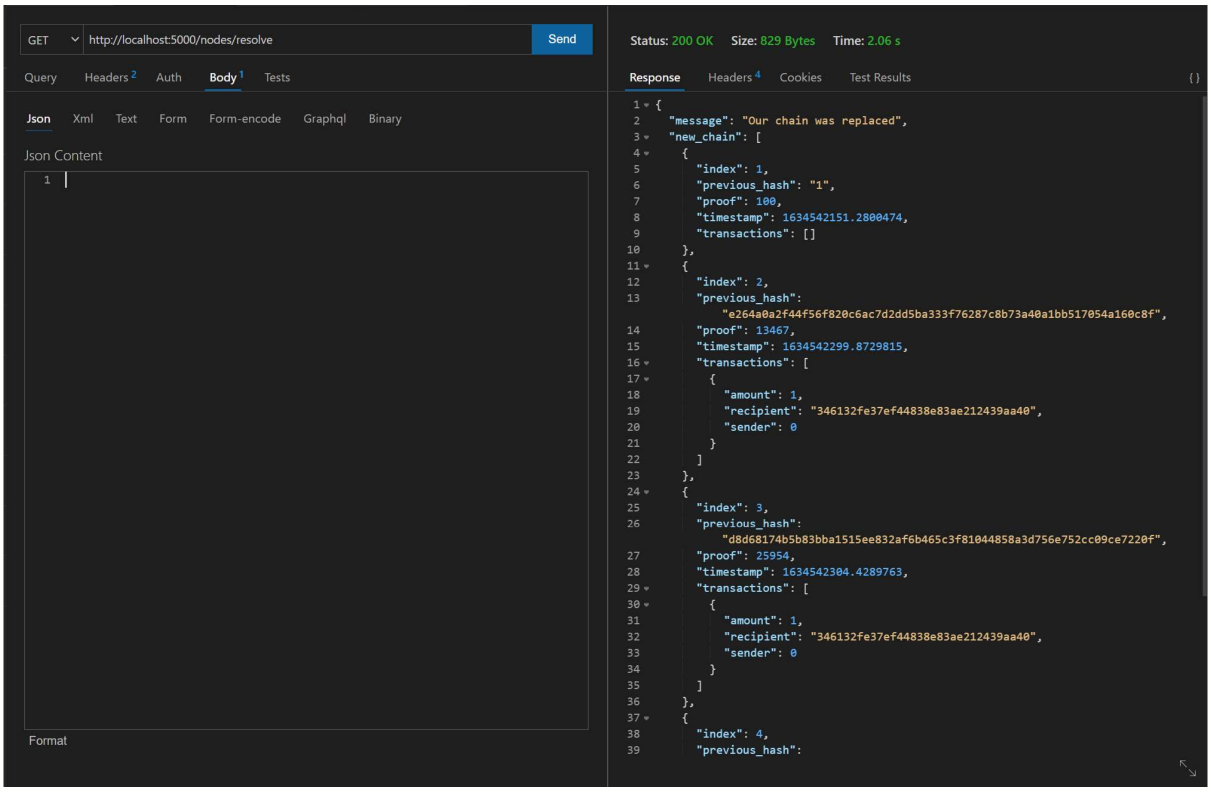
Here, 3 new blocks were mined on node2 and this is how it looks after -

Chain on Node2:

```
{
    "chain": [
        {
            "index": 1,
            "previous_hash": "1",
            "proof": 100,
            "timestamp": 1634540995.5530374,
            "transactions": []
        },
        {
            "index": 2,
            "previous_hash": "7513cea26a0913bc4bb469d36e91ee62d6fa8f3166d3653241316
23a875d9010",
            "proof": 54479,
            "timestamp": 1634541627.0620131,
            "transactions": [
                {
                    "amount": 1,
                    "recipient": "cde1a6895feb44e593c141b2661ac9e5",
                    "sender": 0
                }
            ]
        },
        {
            "index": 3,
            "previous_hash": "c80a4e95cc03eb1ad61b535f863af41ce9918613bfc381a74d570
7ae32df60df",
            "proof": 116876,
            "timestamp": 1634541631.8442833,
            "transactions": [
                {
                    "amount": 1,
                    "recipient": "cde1a6895feb44e593c141b2661ac9e5",
                    "sender": 0
                }
            ]
        },
        {
            "index": 4,
            "previous_hash": "f6e0a4c21aa4edf332e132e9afae51fa8a0e595ceb4defa6ae04d
d92cfdbd99a",
            "proof": 137176,
            "timestamp": 1634541633.3526533,
            "transactions": [
                {
                    "amount": 1,
                    "recipient": "cde1a6895feb44e593c141b2661ac9e5",
                    "sender": 0
                }
                ]
            }
        ],
        "length": 4
}
```

Now we can see that the length of the block running on node2 is greater than that which is running on node1. By the consensus algorithm, the chain with the greatest length should prevail.

So, to resolve this we use the - http://localhost:5000/nodes/resolve **endpoint -**



```
{
  "message": "Our chain was replaced",
  "new_chain": [
    {
      "index": 1,
      "previous_hash": "1",
      "proof": 100,
      "timestamp": 1634542151.2800474,
      "transactions": []
    },
    {
      "index": 2,
      "previous_hash":
        "e264a0a2f44f56f820c6ac7d2dd5ba333f76287c8b73a40a1bb517054a160c8f",
      "proof": 13467,
      "timestamp": 1634542299.8729815,
      "transactions": [
        {
          "amount": 1,
          "recipient": "346132fe37ef44838e83ae212439aa40",
          "sender": 0
        }
      ]
    },
    {
      "index": 3,
      "previous_hash":
        "d8d68174b5b83bba1515ee832af6b465c3f81044858a3d756e752cc09ce7220f",
      "proof": 25954,
      "timestamp": 1634542304.4289763,
      "transactions": [
        {
          "amount": 1,
          "recipient": "346132fe37ef44838e83ae212439aa40",
          "sender": 0
        }
      ]
    },
    {
      "index": 4,
      "previous_hash":
```

We can see how the longest chain running on a different machine (here, port) on the same network replaces the shorter chain.

With this, we have seen how the basic implementation of a blockchain is done.