

Experiment: 03

Aim: Create a Cryptocurrency using Python and perform mining in the Blockchain created.

Theory:

1. Blockchain Overview

Blockchain is a **distributed and decentralized ledger** that stores information in a series of linked blocks.

Each block contains:

- Transaction data
- Timestamp
- Previous block's hash
- Its own unique hash (digital fingerprint)

Once data is recorded in a blockchain, it becomes **immutable** because altering one block would require recalculating all subsequent blocks.

2. Mining

Mining is the process of:

1. Collecting pending transactions into a block.
2. Performing a computational puzzle (Proof-of-Work) to find a valid hash.
3. Adding the new block to the blockchain.

Broadcasting it to all connected peers.

Miners are rewarded with cryptocurrency for successfully mining a block.

3. Multi-Node Blockchain Network

In this lab, we simulate **three independent blockchain nodes** (5001, 5002, 5003).

Each node:

- Runs on a separate port.
- Maintains its own copy of the blockchain.
- Can connect with peers to share and validate blocks.

4. Consensus Mechanism

We use the **Longest Chain Rule**:

- If multiple versions of the chain exist, the **longest valid chain** is chosen.
- This ensures all nodes agree on a single transaction history.

5. Transactions & Mining Reward

Each transaction has:

- Sender
- Receiver
- Amount

When mining a block:

- Pending transactions are added to the block.
- A reward transaction is added automatically to pay the miner.

6. Chain Replacement

When /replace_chain is called:

1. Node requests chains from peers.
2. If it finds a longer and valid chain, it replaces its own.
3. This keeps the blockchain consistent across all nodes.

Implementation:

1. Add Transactions - invoke add_transactions() as a POST request.

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** http://127.0.0.1:5001/add_transaction
- Body:** Raw JSON (selected)
- JSON Body:**

```
1 "amount": 1000,
2 "receiver": "pooja",
3 "sender": "4208036bca004b56b2e456b4daff0059"
```
- Response Status:** 201 CREATED
- Response Body:**

```
1 {
2   "message": "This transaction will be added to Block 4"
3 }
```

2. mining - mine_block()

The screenshot shows a REST API testing interface with the following details:

- Method:** GET
- URL:** http://127.0.0.1:5001/mine_block
- Headers:** (6 items listed)
- Body:** (Empty JSON object)
- Response Status:** 200 OK
- Response Time:** 25 ms
- Response Size:** 621 B
- Response Content:**

```
2
{
  "index": 4,
  "message": "Congratulations, you just mined a block!",
  "previous_hash": "ed3cc6f96be22e405a6ed26538e3c0a71a79779d682eff6e7e713be9ec690104",
  "proof": 21391,
  "timestamp": "2026-01-27 11:26:58.840324",
  "transactions": [
    {
      "amount": 1,
      "receiver": "Richard",
      "sender": "4200036bca004b56b2e456b4daff0059"
    },
    {
      "amount": 1000,
      "receiver": "pooja",
      "sender": "4200036bca004b56b2e456b4daff0059"
    }
  ]
}
```

3.fetch the chain - get_chain(),

The screenshot shows a REST API testing interface with the following details:

- Method:** GET
- URL:** http://127.0.0.1:5001/get_chain
- Headers:** (6 items listed)
- Body:** (Empty JSON object)
- Response Status:** 200 OK
- Response Time:** 4 ms
- Response Size:** 533 B
- Response Content:**

```
{
  "chain": [
    {
      "index": 1,
      "previous_hash": "0",
      "proof": 1,
      "timestamp": "2026-01-27 11:09:25.029105",
      "transactions": []
    },
    {
      "index": 2,
      "previous_hash": "6d617bf1d8543b614960a1afcdbe1e1bb5f00e0e840c15c815761faffe79dec0",
      "proof": 533,
      "timestamp": "2026-01-27 11:12:17.003830",
      "transactions": []
    }
  ]
}
```

4. node - invoke connect_node()

The screenshot shows a Postman request to `http://127.0.0.1:5001/connect_node`. The request method is `POST`. The request body contains the following JSON:

```
1 {
2     "nodes": [
3         "http://127.0.0.1:5000",
4         "http://127.0.0.1:5002"
5     ]
6 }
```

The response status is `201 CREATED` with a response time of 5 ms and a response size of 325 B. The response body is:

```
1 {
2     "message": "All the nodes are now connected. The Hadcoin Blockchain now contains the following nodes:",
3     "total_nodes": [
4         "127.0.0.1:5002",
5         "127.0.0.1:5000"
6     ]
7 }
```

5. replace the longest chain - replace_chain()

The screenshot shows a Postman request to `http://127.0.0.1:5001/replace_chain`. The request method is `GET`. The response status is `200 OK` with a response time of 3 ms and a response size of 1.2 KB. The response body is:

```
1 {
2     "actual_chain": [
3         {
4             "index": 1,
5             "previous_hash": "0",
6             "proof": 1,
7             "timestamp": "2026-01-27 11:09:25.029105",
8             "transactions": []
9         },
10        {
11            "index": 2,
12            "previous_hash": "6d617bf1d8543b614960a1afcdbe1e1bb5f00e0e840c15c815761fafffe79dec0",
13            "proof": 533,
14            "timestamp": "2026-01-27 11:12:17.003830",
15            "transactions": [
16                {
17                    "amount": 1,
18                    "receiver": "Richard",
19                }
20            ]
21        }
22    ]
23 }
```

```

19           "sender": "4208036bca004b56b2e456b4daff0059"
20       }
21     ]
22   },
23   {
24     "index": 3,
25     "previous_hash": "1af95c07c0d568f49621eb2fd8759ff3c729423d818dae19a05b2083fd",
26     "proof": 45293,
27     "timestamp": "2026-01-27 11:19:00.005438",
28     "transactions": [
29       {
30         "amount": 1,
31         "receiver": "Richard",
32         "sender": "4208036bca004b56b2e456b4daff0059"
33     ],
34
35   "index": 4,
36   "previous_hash": "ed3cc6f96be22e405a6ed26538e3c0a71a79779d682eff6e7e713be9ec690104",
37   "proof": 21391,
38   "timestamp": "2026-01-27 11:26:58.840324",
39   "transactions": [
40     {
41       "amount": 1,
42       "receiver": "Richard",
43       "sender": "4208036bca004b56b2e456b4daff0059"
44     },
45     {
46       "amount": 1000,
47       "receiver": "pooja",
48       "sender": "4208036bca004b56b2e456b4daff0059"
49     },
50     {
51       "amount": 1,
52       "receiver": "Richard",
53       "sender": "4208036bca004b56b2e456b4daff0059"
54     }
55   ],
56   "message": "All good. The chain is the largest one."
57 }
58 ],
59 ],
60 "message": "All good. The chain is the largest one."
61 }

```

Conclusion: This experiment demonstrates the practical application of blockchain principles through the implementation of a simulated cryptocurrency. By modeling transactions, mining, and consensus across multiple nodes, it provides clear insight into how decentralized blockchain systems function. The experiment emphasizes the role of security, consensus mechanisms, and decentralization in maintaining a reliable and distributed ledger.