Bhartiya Vidya Bhavan's
Sardar Patel Institute of Technology, Mumbai-400058
Department of Electronics and Telecommunication Engineering
IT424:Blockchain Technology and Applications

Lab-5: Blockchain Programming-III
Develop a blockchain application in Python

Objective: Develop a blockchain application in Python

Outcomes: After successful completion of lab students should be able to
Implement a public blockchain
Build a simple application
Use REST API and Flask microframework

System Requirements:
PC (C2D, 4GB RAM, 100GB HDD space and NIC)
Ubuntu Linux 14.04/20.04
Internet connectivity
Python Cryptography and Pycrypto
REST API
Flask Framework

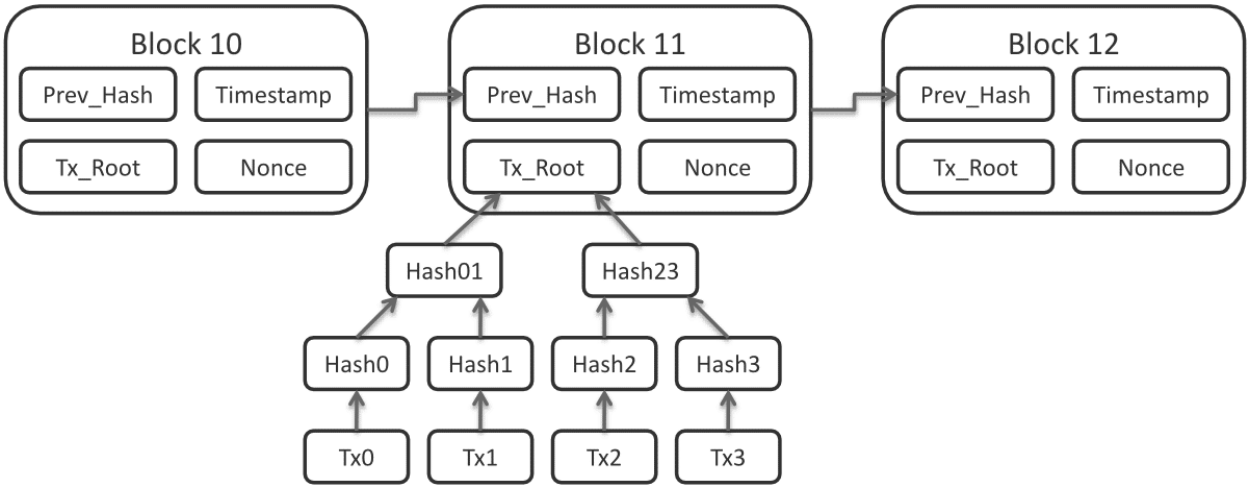


Figure-1: Blockchain Implementation

Refer to the step-by-step tutorial by Satvik Kansal [1]
Develop a blockchain application from scratch in Python

Procedure:

[1] Clone it from git

$ git clone https://github.com/satwikkansal/python_blockchain_app.git

Command Prompt

```
Microsoft Windows [Version 10.0.19044.1586]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Nupur Gupte>D:

D:\>cd dev

D:\dev>git clone https://github.com/satwikkansal/python_blockchain_app.git
Cloning into 'python_blockchain_app'...
remote: Enumerating objects: 146, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 146 (delta 0), reused 1 (delta 0), pack-reused 143
Receiving objects: 100% (146/146), 223.96 KiB | 1.48 MiB/s, done.
Resolving deltas: 100% (67/67), done.
```

[2] Install the dependencies,

$ cd python_blockchain_app

```
D:\dev>cd python_blockchain_app
```

$ pip install -r requirements.txt

```
D:\dev\python_blockchain_app>pip install -r requirements.txt
Requirement already satisfied: Flask~=1.1 in c:\python310\lib\site-packages (from -r requirements.txt (line 1)) (1.1.4)
Requirement already satisfied: requests~=2.22 in c:\python310\lib\site-packages (from -r requirements.txt (line 2)) (2.27.1)
Requirement already satisfied: Jinja2<3.0,>=2.10.1 in c:\python310\lib\site-packages (from Flask~=1.1->-r requirements.txt (line 1)) (2.11.3)
Requirement already satisfied: Werkzeug<2.0,>=0.15 in c:\python310\lib\site-packages (from Flask~=1.1->-r requirements.txt (line 1)) (1.0.1)
Requirement already satisfied: click<8.0,>=5.1 in c:\python310\lib\site-packages (from Flask~=1.1->-r requirements.txt (line 1)) (7.1.2)
Requirement already satisfied: itsdangerous<2.0,>=0.24 in c:\python310\lib\site-packages (from Flask~=1.1->-r requirements.txt (line 1)) (1.1.0)

Requirement already satisfied: certifi>=2017.4.17 in c:\python310\lib\site-packages (from requests~=2.22->-r requirements.txt (line 2)) (2021.10
.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in c:\python310\lib\site-packages (from requests~=2.22->-r requirements.txt (line 2)) (
2.0.12)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\python310\lib\site-packages (from requests~=2.22->-r requirements.txt (line 2)) (1.26
.8)
Requirement already satisfied: idna<4,>=2.5 in c:\python310\lib\site-packages (from requests~=2.22->-r requirements.txt (line 2)) (3.3)
Requirement already satisfied: MarkupSafe>=0.23 in c:\python310\lib\site-packages (from Jinja2<3.0,>=2.10.1->Flask~=1.1->-r requirements.txt (li
ne 1)) (2.0.1)
```

[3] Start a blockchain node server

Windows users can follow this:
#https://flask.palletsprojects.com/en/1.1.x/cli/#application-discovery

$ export FLASK_APP=node_server.py

$ flask run --port 8000

```
D:\dev\python_blockchain_app>set FLASK_APP=node_server.py

D:\dev\python_blockchain_app>flask run --port 8000
 * Serving Flask app "node_server.py"
 * 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://127.0.0.1:8000/ (Press CTRL+C to quit)
```

[4] Run the application on a different terminal session:

$ python run_app.py

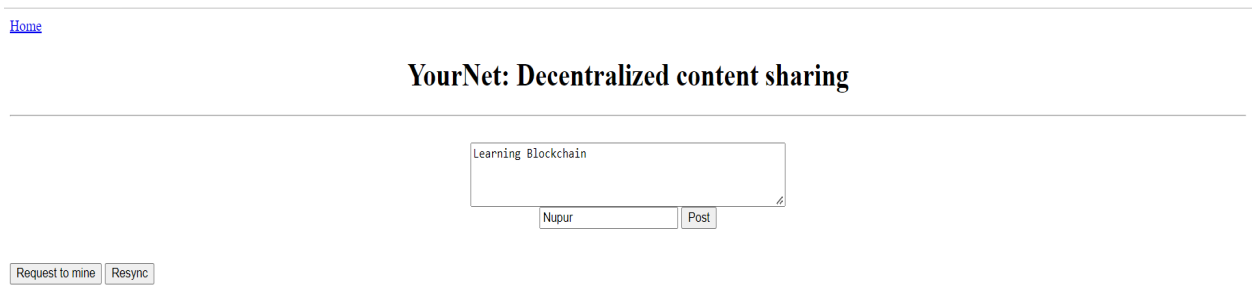
The application should be up and running at http://localhost:5000

```
D:\dev\python_blockchain_app>python run_app.py
 * Serving Flask app "app" (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: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 280-946-206
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

[5] Explore the steps given in the tutorial and complete it. Refer [1]

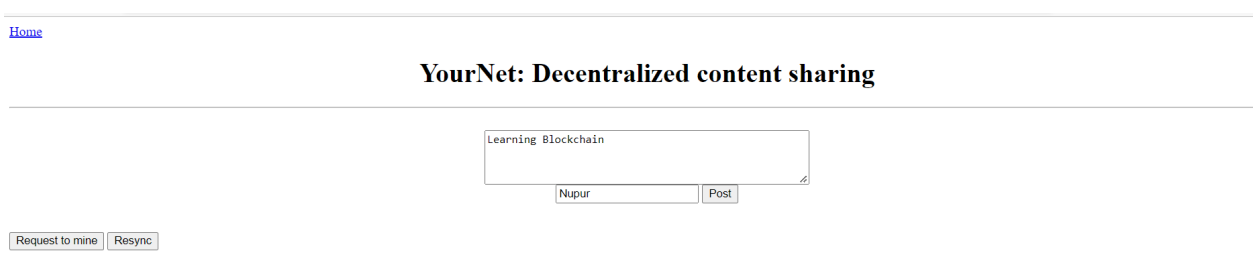
Add screenshots with a brief description.

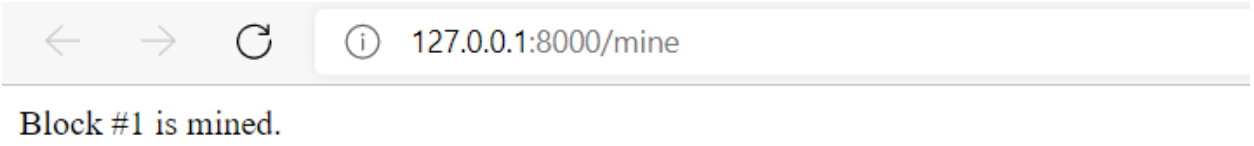
Posting content



when post some content the data gets added to pending transactions, we submit a new transaction

Request to mine





When mine is requested the pending transactions to the blockchain by adding them to the block and figuring out Proof Of Work.

Resyncing with chain for updated data

Resync displays the transactions and authors currently present in the blockchain

GET request

Returns a JSON of the blockchain, we this we can see each block's structure which contains timestamp, hash, previous hash, transactions, index, nonce

http://127.0.0.1:8000/chain

Save

GET http://127.0.0.1:8000/chain Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results Status: 200 OK Time: 37 ms Size: 1 KB Save Response

Pretty Raw Preview Visualize JSON


```
{
    "length": 3,
    "chain": [
        {
            "index": 0,
            "transactions": [],
            "timestamp": 0,
            "previous_hash": "0",
            "nonce": 0,
            "hash": "6dbf23122cb5046cc5c0c1b245c75f8e43c59ca8ffeac292715e5078e631d0c9"
        },
        {
            "index": 1,
            "transactions": [
                {
                    "author": "Nupur",
                    "content": "Learning blockchain",
                    "timestamp": 1647796111.820961
                }
            ],
            "timestamp": 1647796220.3337712,
            "previous_hash": "6dbf23122cb5046cc5c0c1b245c75f8e43c59ca8ffeac292715e5078e631d0c9",
            "nonce": 38,
            "hash": "00d4b67b1b5415d0fad9e32843b592b14c45d53eed5feaae02fae0fa53686d72"
```


Working with multiple nodes

We can work with multiple nodes by running the flask app on the desired ports and then registering the node with the already running node. Here we registered the nodes 8001 and 8002 with already running port 8000. Hence we have added peers in this decentralized system

http://127.0.0.1:8001/register_with

POST http://127.0.0.1:8001/register_with

Params Authorization Headers (10) Body Pre-request Script Tests Settings

Query Params

	KEY	VALUE	DESCRIPTION
	Key	Value	Description

Body Cookies Headers (4) Test Results Status: 200 OK Time: 68 ms

Pretty Raw Preview Visualize HTML

```
1  Registration successful
```

http://127.0.0.1:8002/register_with Save

POST http://127.0.0.1:8002/register_with Send

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

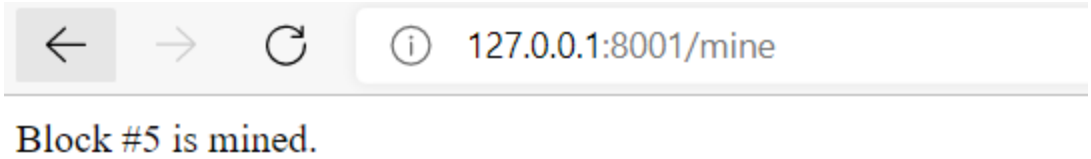
none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1  {"node_address": "http://127.0.0.1:8000"}
```

Body Cookies Headers (4) Test Results Status: 200 OK Time: 51 ms Size: 176 B Save Response

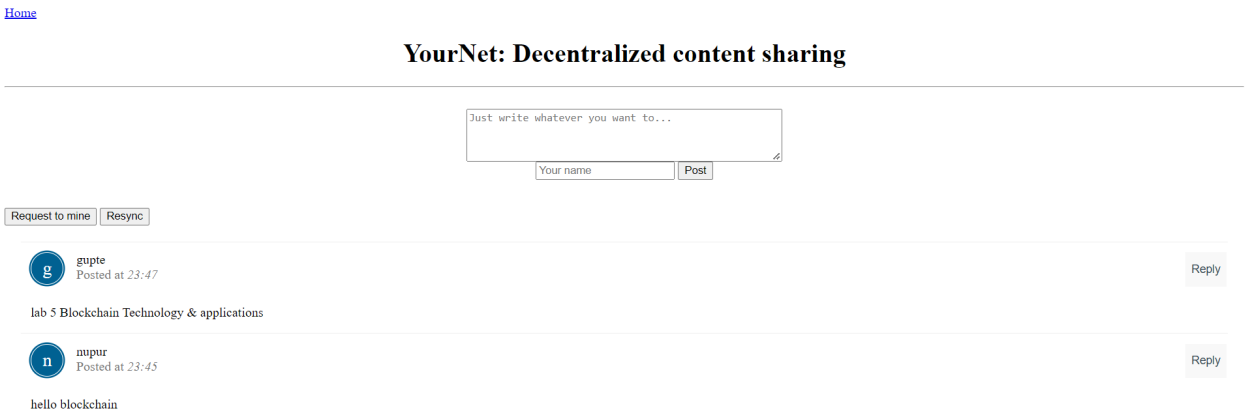
Pretty Raw Preview Visualize HTML

```
1  Registration successful
```



Block #5 is mined.

Thus the new blocks can now participate in the mining process



http://127.0.0.1:8001/chain

Save

GET http://127.0.0.1:8001/chain Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (4) Test Results Status: 200 OK Time: 15 ms Size: 1.02 KB Save Response

Pretty Raw Preview Visualize JSON


```
14            "transactions": [
15                {
16                    "author": "nupur",
17                    "content": "hello blockchain",
18                    "timestamp": 1647800116.094758
19                }
20            ],
21            "timestamp": 1647800118.2588053,
22            "previous_hash": "6dbf23122cb5046cc5c0c1b245c75f8e43c59ca8ffeac292715e5078e631d0c9",
23            "nonce": 210,
24            "hash": "00274519690caa3f9e4483c2c877c73f2d5db427290e98ad9cdf6df9cca9aa8a"
25        },
26        {
27            "index": 2,
28            "transactions": [
29                {
30                    "author": "gupte",
31                    "content": "lab 5 Blockchain Technology & applications",
32                    "timestamp": 1647800255.325827
33                }
34            ],
35            "timestamp": 1647800257.3741848,
36            "previous_hash": "00274519690caa3f9e4483c2c877c73f2d5db427290e98ad9cdf6df9cca9aa8a",
37            "nonce": 57,
```


http://127.0.0.1:8002/chain

Save

GET http://127.0.0.1:8002/chain Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	•••	Bulk Edit

Body Cookies Headers (4) Test Results Status: 200 OK Time: 15 ms Size: 1.05 KB Save Response

Pretty Raw Preview Visualize JSON


```
14            "transactions": [
15                {
16                    "author": "nupur",
17                    "content": "hello blockchain",
18                    "timestamp": 1647800116.094758
19                }
20            ],
21            "timestamp": 1647800118.2588053,
22            "previous_hash": "6dbf23122cb5046cc5c0c1b245c75f8e43c59ca8ffeac292715e5078e631d0c9",
23            "nonce": 210,
24            "hash": "00274519690caa3f9e4483c2c877c73f2d5db427290e98ad9cdf6df9cca9aa8a"
25        },
26        {
27            "index": 2,
28            "transactions": [
29                {
30                    "author": "gupte",
31                    "content": "lab 5 Blockchain Technology & applications",
32                    "timestamp": 1647800255.325827
33                }
34            ],
```


We can also view the current blockchain with the GET request through this new nodes

Conclusion:

Thus learned the concept of Blockchain. I learned about the structure of each block. Blockchain is distributed, every node has a copy of the blockchain. It is also decentralized. In this experiment, I saw a decentralized application where multiple nodes can contribute to the blockchain and its mining process.

References:

[1] Develop a blockchain application from scratch in Python

https://gist.github.com/satwikkansal/4a857cad2797b9d199547a752933a715#develop-a-blockchain-application-from-scratch-in-python