Bhartiya Vidya Bhavan's
Sardar Patel Institute of Technology, Mumbai-400058
Department of Computer Science and Engineering
OEIT1:Blockchain Technology and Applications

Lab-7A: Blockchain and Cybersecurity
Part-I
Develop a blockchain application for Cybersecurity

Name : Adwait Purao **UID : 2021300101**
Objective: Develop a blockchain application for Cybersecurity

Outcomes: After successful completion of lab students should be able to

- Implement an Ethereum private blockchain
- Build two-factor authentication (2FA) using Blockchain
- Write a smart contract using Solidity Language
- Compile and run the 2FA using Ethereum Blockchain
- Use REST API and Flask microframework

System Requirements:
PC (C2D, 8GB RAM, 100GB HDD space and NIC),Ubuntu Linux 14.04/20.04
Internet connectivity,Python Cryptography and Pycrypto,Nodejs, Truffle,Ganache-cli
, solidity,REST API

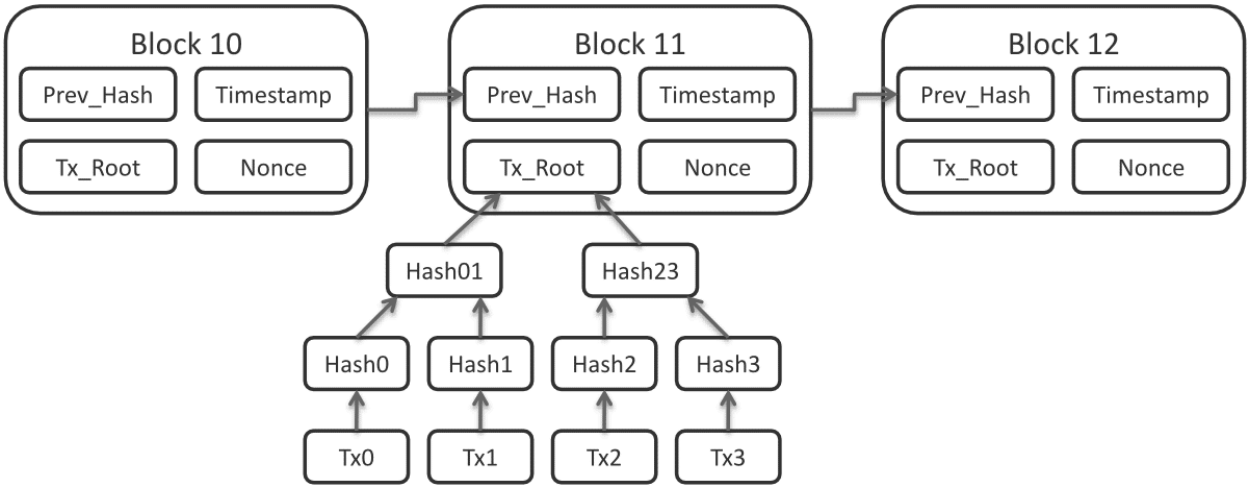


Figure-1: Blockchain Implementation

Part-I: Two-Factor Authentication with Blockchain

Two-factor authentication (2FA) provides an added layer to the existing credential-based system protection as a solution to this drastically growing problem.

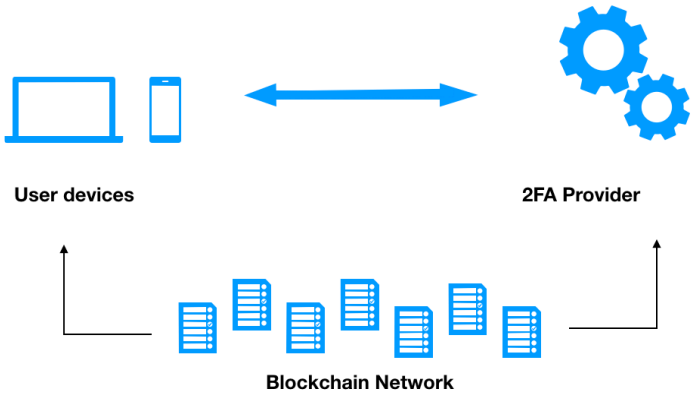


Figure-2: Blockchain-Based 2FA

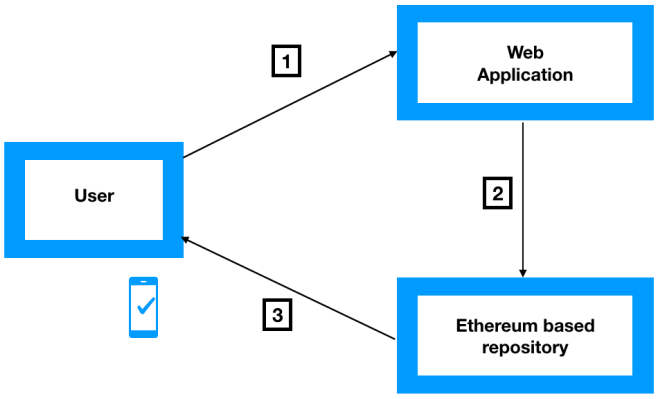


Figure-3:Solution Architecture

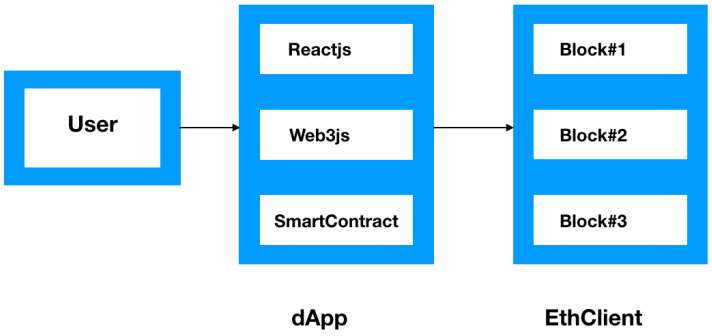


Figure-4: Ethereum based 2FA

Procedure:

[1]Create a directory under BTA

$cd ~

$mkdir BTA/lab7a-2FA

$cd BTA/lab7a-2FA

[2] Clone or download the Ethereum-2FA

$ git clone https://github.com/hoxxep/Ethereum-2FA

$cd Ethereum-2FA

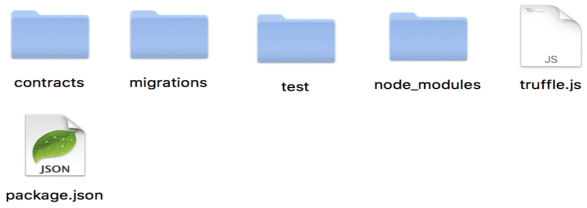


Figure-5: Ethereum-2FA Directory and Files

The files in the preceding screenshot are explained as follows:

contracts: This folder includes our smart contract, TwoFactorAuth.sol

migrations: This folder consists of migration files to deploy the contract to the blockchain

test: This folder consists of server.js, which is responsible for event authentication in our contract

node_modules: This folder includes all the libraries

truffle.js: This configuration file consists of a set of configurations to connect to the blockchain

package.json: This is where we specify a configuration of our projects, such as name and scripts

[3] Install Nodejs

Refer to [2] installation of Nodejs on Ubuntu Linux

How To Install Node.js on Ubuntu 20.04 | DigitalOcean

#Recommended method No. 3

#Check Nodejs version

$node -v

#Turning up Ethereum

#Install ganache-cli

$npm install -g ganache-cli

Components

The following are the three core components of this project, shown in the following diagram:

- A blockchain network (which we will develop with the Ganache CLI)
- A smart contract
- A server communicating with the blockchain

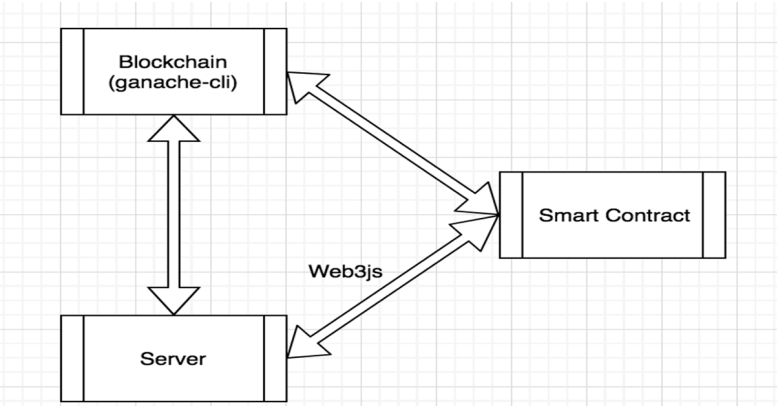


Figure-5: Components of Blockchain-Based 2FA

$ganache-cli

```
user — node /usr/local/bin/ganache-cli — 80×41
Last login: Sun May 13 15:57:44 on ttys001
MacBook-Pro-Macbook:~ user$ ganache-cli
Ganache CLI v6.1.0 (ganache-core: 2.1.0)

Available Accounts
==================
(0) 0x9d92766c6ff285295164d29bfebceb9e88d95f21
(1) 0x7f70815e09840bdfdce8ab24fa0e6e7f46f68b45
(2) 0x2eceef0ad7da38e35e5d5fdb7d4e25510ba788d1
(3) 0x9a2b3c9c032bc34f7bd50be93872db82136e2e8a
(4) 0xcc95b95055c03c61dc406f7247e9dab60f20820d
(5) 0x9178a368f01b6fd21bda5030884c7cd4e7d73bed
(6) 0xd0914248a466e54c83cd8df1ef8b14b69b077627
(7) 0xd77e444e49e0d15c3d995d56cfb95d54d078df7f
(8) 0xec4f6e31fae1963ee59fc9082b11e3dae0c7f6f3
(9) 0x7ed265366670ff176b334dfb2ff011566e906753

Private Keys
==================
(0) 401e2344354aa597d81f0c987f717612e571597e8a9d6bbe5da54f4368a92e9a
(1) 57f92aee8eede3c53a81110debd12e8fee43fc15bfce3c56472232f5e89b687e
(2) 62037c947171f49897a456df1aff3385cf1ca46cbab3c5e13a5e06279f0b8d34
(3) 704a14e48f9e8e294309eda5aed92d8891a8fcda770013c5fb9ccf77d31acb04
(4) 2081626376ca37cba7fd6c5c11c074114506a0797c9ee140855b3476bc02bcd3
(5) ac6756b661f27a486b39a693b8884018cb12b765dd5dc6889ca9f92760e5853f
(6) 32d0606eaf5a826e30d2ffc8adf490417d84629e1e5543e120a1e086ea3f2707
(7) 74b31aff959260ab32044c1879a7a94c69cd9c8f6607aca1e226ddba398fa231
(8) 8007b7ab1b206f3cda425e48812fa8c28e07aac8493693e4ed9dd04fdc358848
(9) 1b78a1b49339bb579399908ba91d785473ddc0e18a3ab99db9cd954280ac8192

HD Wallet
==================
Mnemonic:      desert vacuum wide apology gown afford place bar quarter short et
ernal teach
Base HD Path:  m/44'/60'/0'/0/{account_index}
```

Figure-7: Ganache-cli

Turning up the smart contract

$cd Ethereum-2FA

$truffle test ./test/server.js

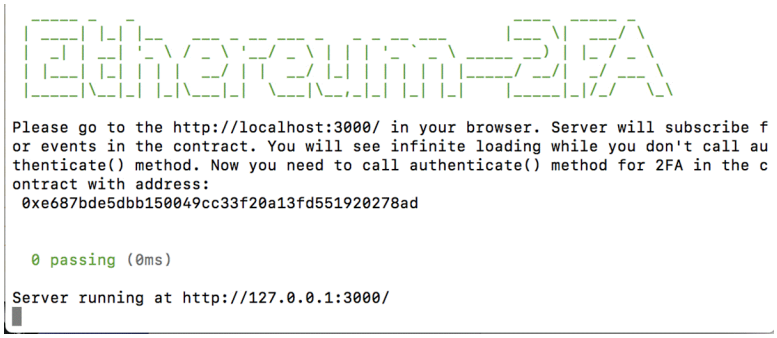
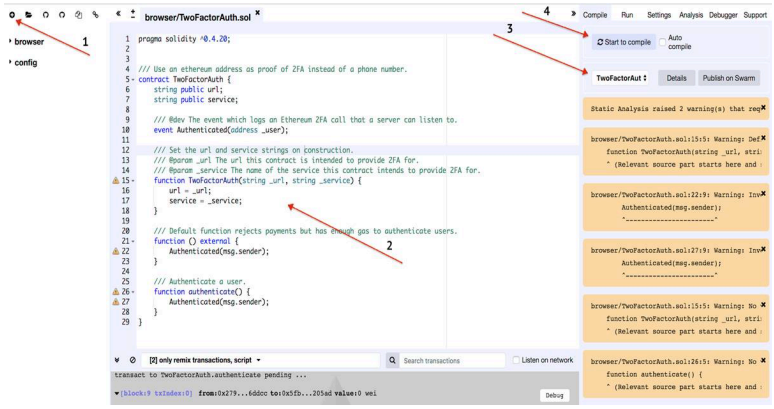


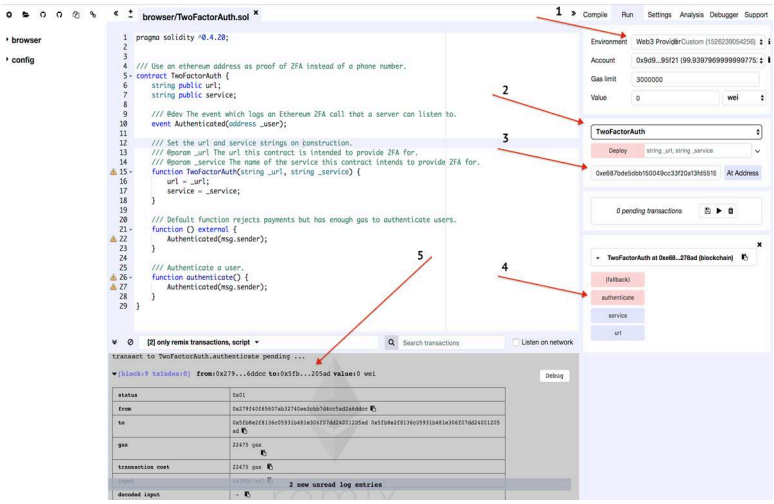
Figure-8:Ethereum-2FA

Open Google Chrome and access the localhost on port 3000

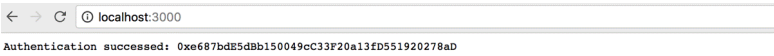
#Testing and verification



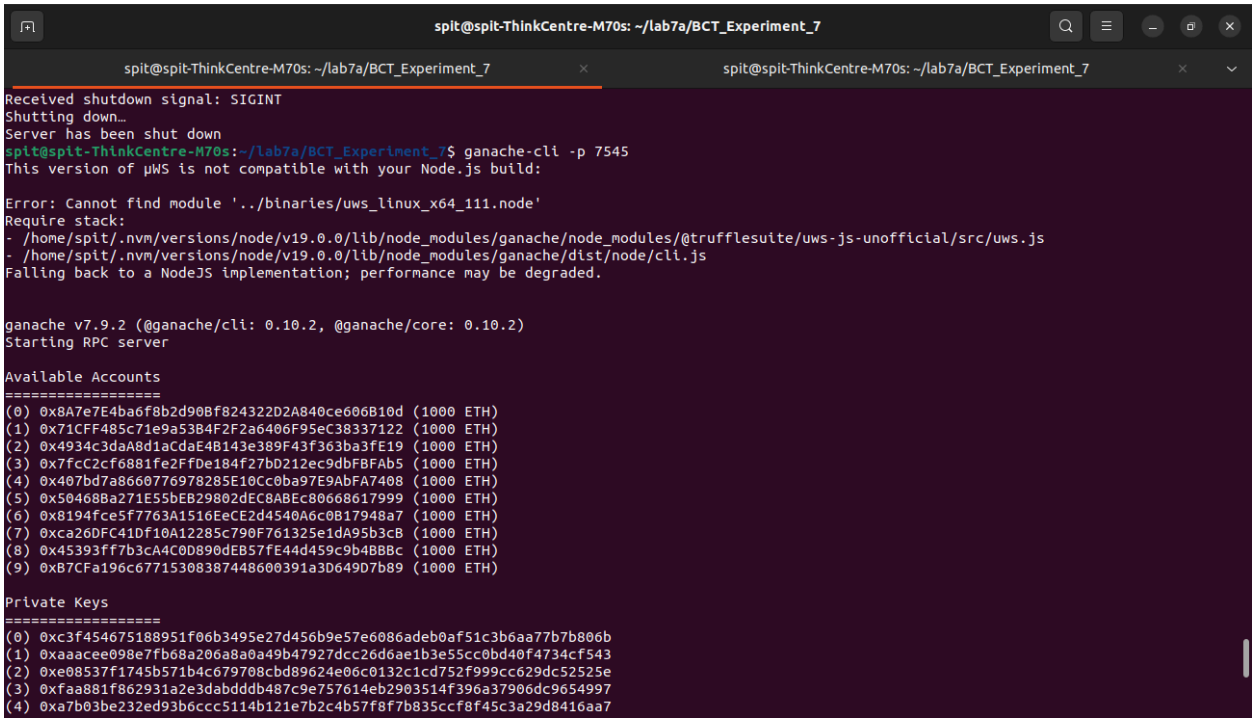
#

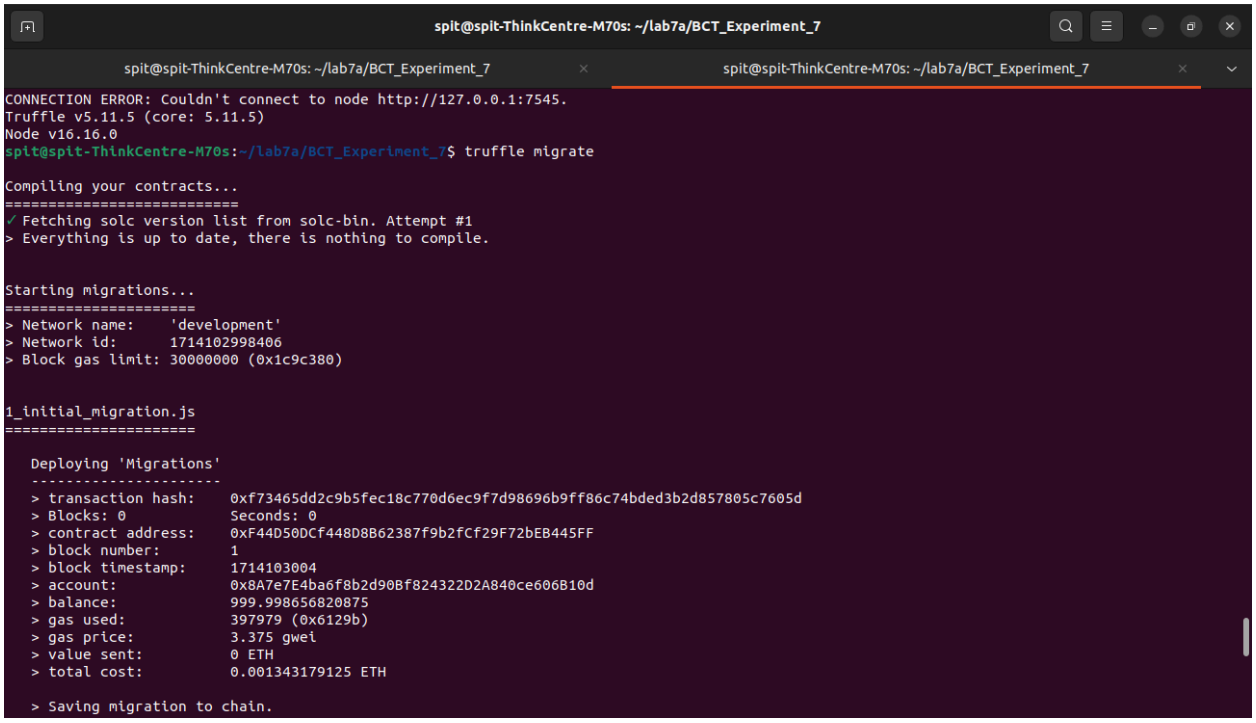


Successful authentication:

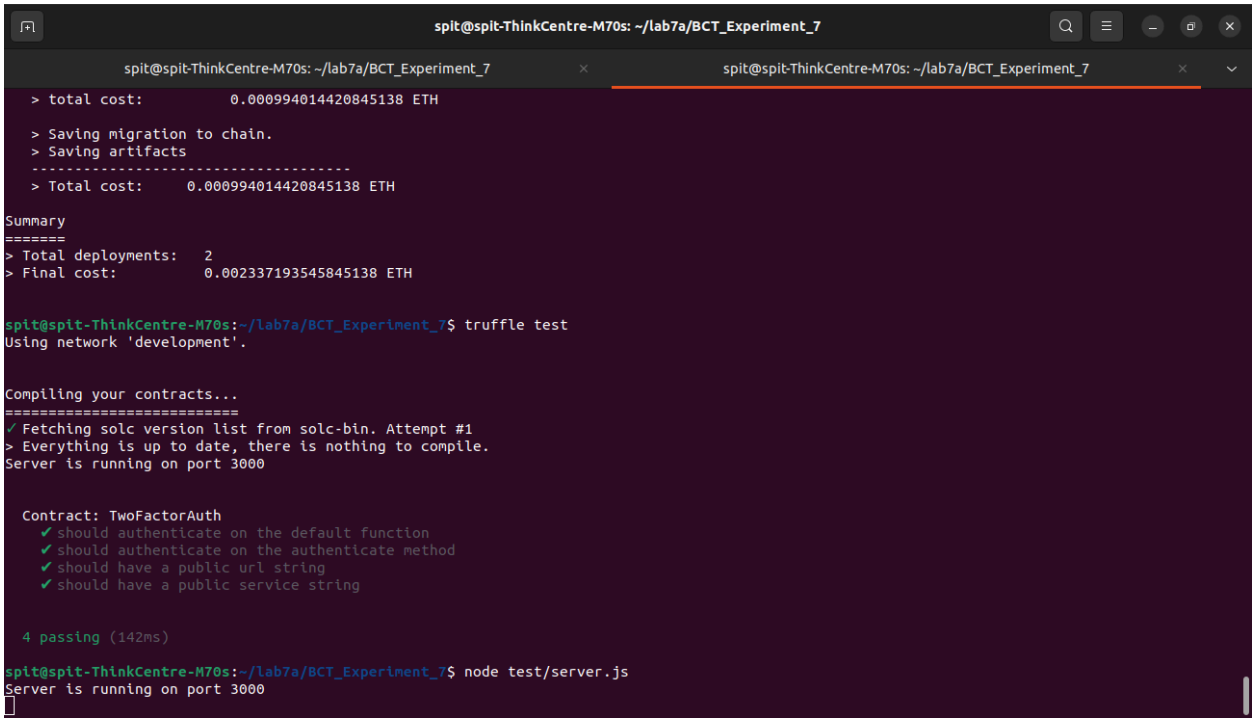


Add your system screenshots with a brief description.


spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
Received shutdown signal: SIGINT
Shutting down…
Server has been shut down
spit@spit-ThinkCentre-M70s:~/lab7a/BCT_Experiment_7$ ganache-cli -p 7545
This version of µWS is not compatible with your Node.js build:
Error: Cannot find module '../binaries/uws_linux_x64_111.node'
Require stack:
- /home/spit/.nvm/versions/node/v19.0.0/lib/node_modules/ganache/node_modules/@trufflesuite/uws-js-unofficial/src/uws.js
- /home/spit/.nvm/versions/node/v19.0.0/lib/node_modules/ganache/dist/node/cli.js
Falling back to a NodeJS implementation; performance may be degraded.
ganache v7.9.2 (@ganache/cli: 0.10.2, @ganache/core: 0.10.2)
Starting RPC server
Available Accounts
==================
(0) 0x8A7e7E4ba6f8b2d90Bf824322D2A840ce606B10d (1000 ETH)
(1) 0x71CFF485c71e9a53B4F2F2a6406F95eC38337122 (1000 ETH)
(2) 0x4934c3daA8d1aCdaE4B143e389F43f363ba3fE19 (1000 ETH)
(3) 0x7fcC2cf6881fe2FfDe184f27bD212ec9dbFBFAb5 (1000 ETH)
(4) 0x407bd7a8660776978285E10Cc0ba97E9AbFA7408 (1000 ETH)
(5) 0x50468Ba271E55bEB29802dEC8ABEc80668617999 (1000 ETH)
(6) 0x8194fce5f7763A1516EeCE2d4540A6c0B17948a7 (1000 ETH)
(7) 0xca26DFC41Df10A12285c790F761325e1dA95b3cB (1000 ETH)
(8) 0x45393ff7b3cA4C0D890dEB57fE44d459c9b4BBBc (1000 ETH)
(9) 0xB7CFa196c6771530838744860039 1a3D649D7b89 (1000 ETH)
Private Keys
==================
(0) 0xc3f454675188951f06b3495e27d456b9e57e6086adeb0af51c3b6aa77b7b806b
(1) 0xaaacee098e7fb68a206a8a0a49b47927dcc26d6ae1b3e55cc0bd40f4734cf543
(2) 0xe08537f1745b571b4c679708cbd89624e06c0132c1cd752f999cc629dc52525e
(3) 0xfaa881f862931a2e3dabdddb487c9e757614eb2903514f396a37906dc9654997
(4) 0xa7b03be232ed93b6ccc5114b121e7b2c4b57f8f7b835ccf8f45c3a29d8416aa7


spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
spit@spit-ThinkCentre-M70s: ~/lab7a/BCT_Experiment_7
CONNECTION ERROR: Couldn't connect to node http://127.0.0.1:7545.
Truffle v5.11.5 (core: 5.11.5)
Node v16.16.0
spit@spit-ThinkCentre-M70s:~/lab7a/BCT_Experiment_7$ truffle migrate
Compiling your contracts...
===========================
✓ Fetching solc version list from solc-bin. Attempt #1
> Everything is up to date, there is nothing to compile.
Starting migrations...
======================
> Network name: 'development'
> Network id: 1714102998406
> Block gas limit: 30000000 (0x1c9c380)
1_initial_migration.js
======================
Deploying 'Migrations'

> transaction hash: 0xf73465dd2c9b5fec18c770d6ec9f7d98696b9ff86c74bded3b2d857805c7605d
> Blocks: 0   Seconds: 0
> contract address: 0xF44D50DCf448D8B62387f9b2fCf29F72bEB445FF
> block number: 1
> block timestamp: 1714103004
> account: 0x8A7e7E4ba6f8b2d90Bf824322D2A840ce606B10d
> balance: 999.998656820875
> gas used: 397979 (0x6129b)
> gas price: 3.375 gwei
> value sent: 0 ETH
> total cost: 0.001343179125 ETH
> Saving migration to chain.



Authenticated : 0x71CFF485c71e9a53B4F2F2a6406F95eC38337122

Conclusion:

This lab experiment provided valuable hands-on experience in utilizing blockchain technology for cybersecurity enhancement. Through implementing an Ethereum private blockchain, integrating two-factor authentication (2FA), writing smart contracts in Solidity, and executing these functionalities on the Ethereum blockchain, I gained practical skills in deploying secure and decentralized solutions. Additionally, utilizing REST API and Flask microframework facilitated seamless interaction with external applications, emphasizing the importance of interoperability. Overall, this experience equipped me with foundational knowledge and practical skills to innovate and contribute effectively to cybersecurity practices.

References:

[1] Two-factor authentication through an Ethereum contract.

https://github.com/hoxxep/Ethereum-2FA

[2] How To Install Node.js on Ubuntu 20.04

How To Install Node.js on Ubuntu 20.04 | DigitalOcean