Effective Identity Authentication Based on Multiattribute Centers for Secure Government Data Sharing

Now-a-days all countries are digitizing their citizens data and different organizations or departments may require this data, for example bank or electricity departments may want to access this data to know about citizen capability of repayment. So government will allow such different departments to access their citizen’s data. In all existing applications all citizen’s data will be maintained in a single centralized servers and then for DATA REQUESTER KEY authentication government will be using third party authentication servers.

Third party servers can be attack by attacker to steal all citizens private and public key to decrypt data and then all citizen’s data privacy will be at risk. Sometime some malicious employees of third party server managing employees can sale all citizen data to other companies or can alter citizen data and in such situations also citizen’s privacy will be at risk.

To overcome from above issue author of this paper employing different technologies such as IPFS and Blockchain decentralized data storage and by using this technologies we are not required to depend on third party servers and data owner data will be secured.

In propose work author using following entities

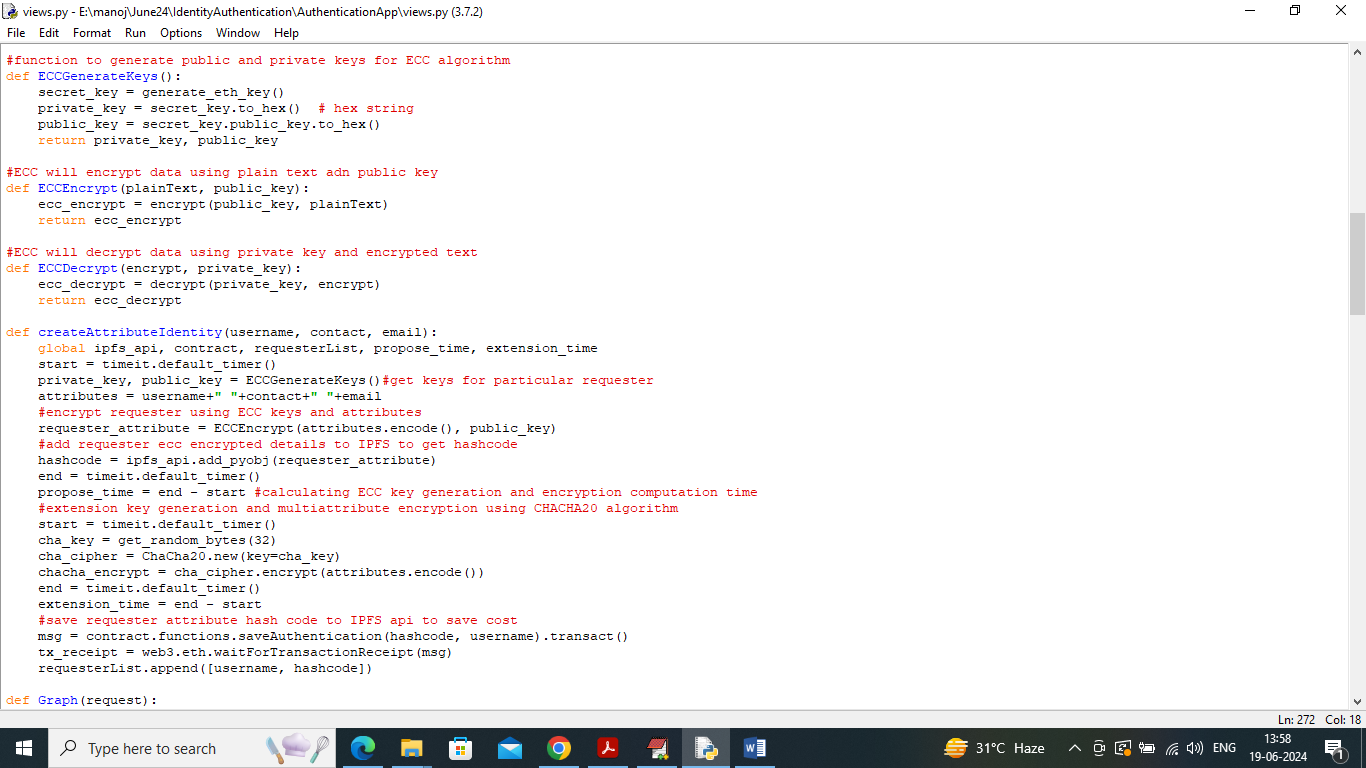
1. Data Requester: data requester will request for private keys from authentication server and it just required to generate keys. Requester will generate personal private keys using private keys sent by authentication server. Data Requester will encrypt multiple attributes (requester name, contact no, department name) using personal keys and then saved data in IPFS system. IPFS will stored requester data and then returned hash code and this hash code will get saved in Blockchain to verify requester.
2. Data Owner: data owner will connect to Blockchain and verify requester and if requester hash code available in Blockchain then only data owner will give access to requester.
3. IPFS: Blockchain charge heavy gas values for storage so if we store all requester attributes in Blockchain then government has to pay huge charges so author of this paper saving Request attributes details in IPFS and the hash code returned by IPFS for those attributes will get saved in Blockchain.
4. Blockchain: Blockchain smart contract just saved Requester hash code in Blockchain for future verification of Data Requester

To generate keys and for encryption author using ECC (Elliptic Curve Cryptography) and this algorithms key generation, initialization and encryption process we can read from base paper.

Extension 1: in this project author just using simple citizen details which will not sufficient to verify citizen authentication so as extension we are taking citizen details along with address proof document upload which can help in authenticating citizen as genuine or fake.

Extension 2: In propose paper author using ECC algorithm which is very much heavy in computation and required heavy storage in IPFS and to reduce this computation cost we are employing another lightweight key generation and encryption algorithm called CHACHA20 as extension 2.

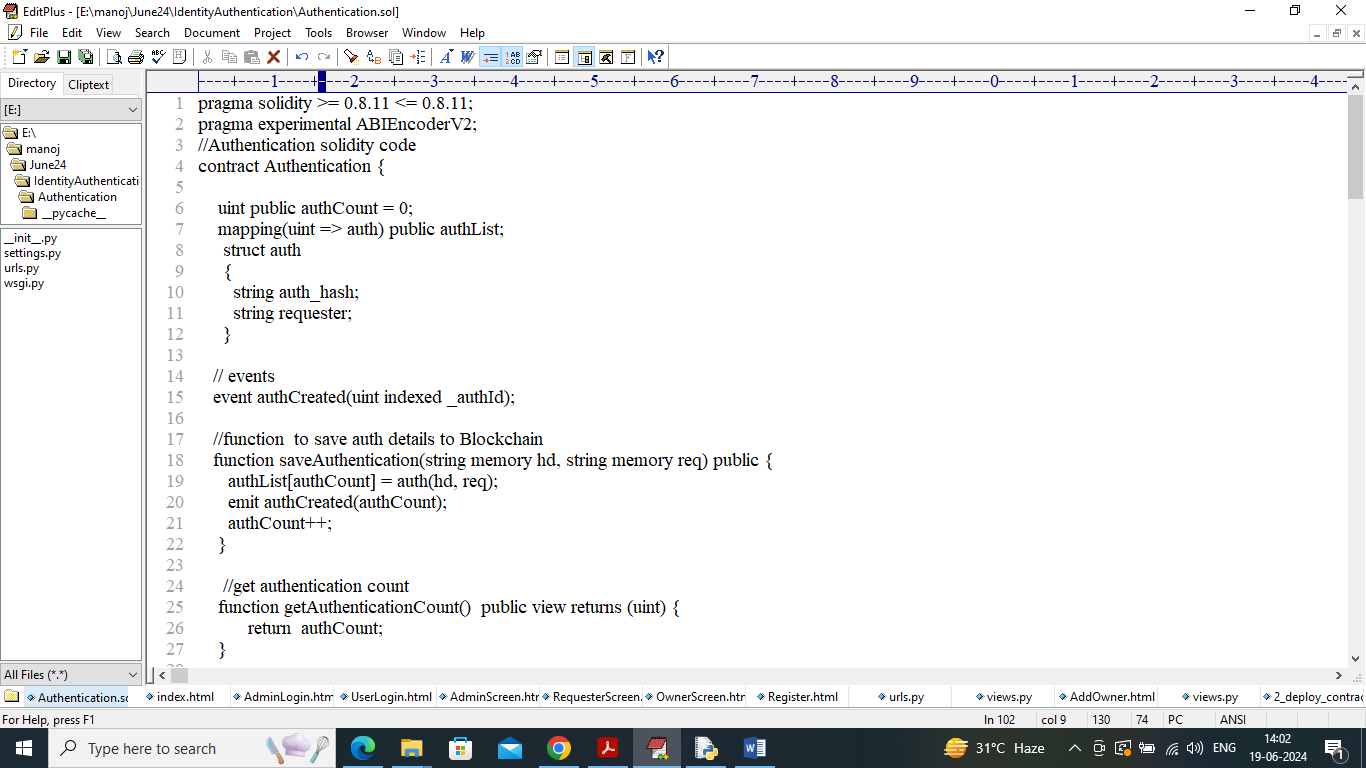
In below screen showing code for key generation and encryption using both ECC and CHACHA20.



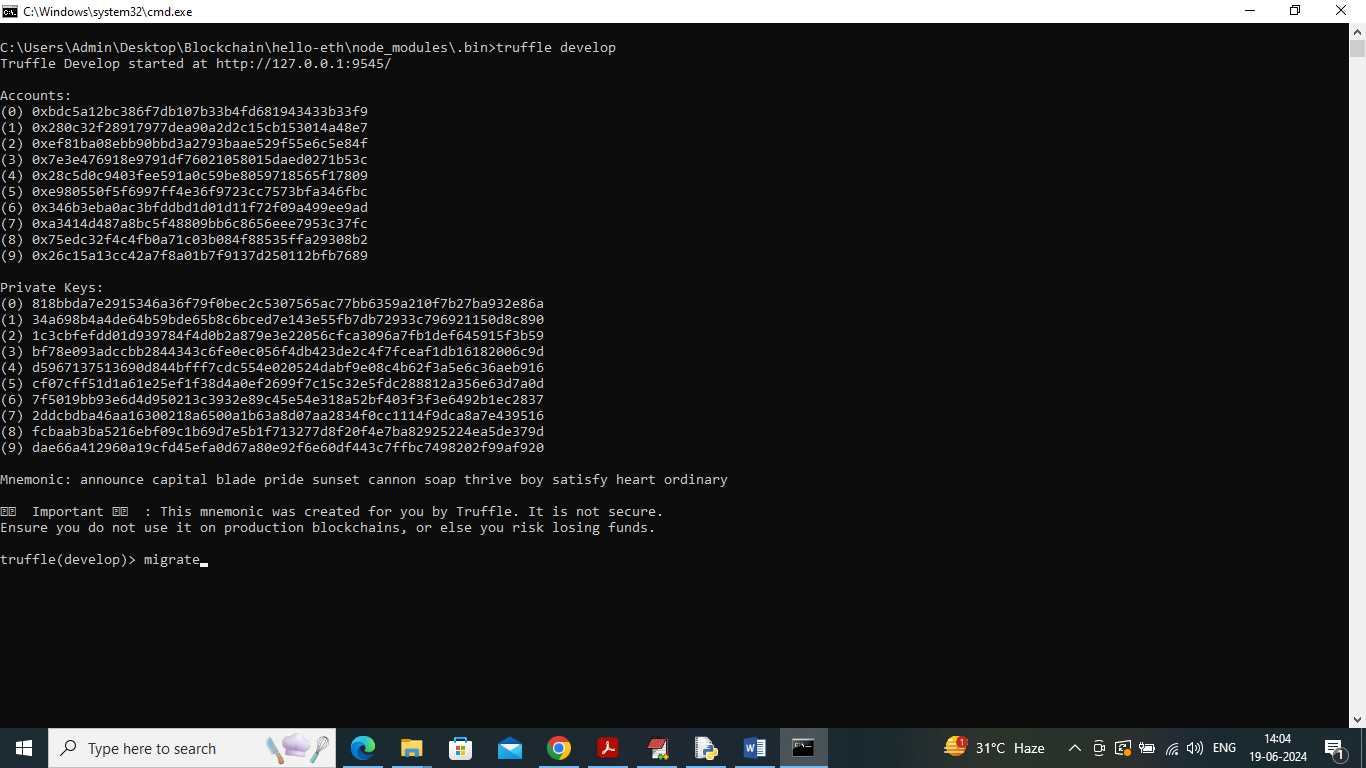
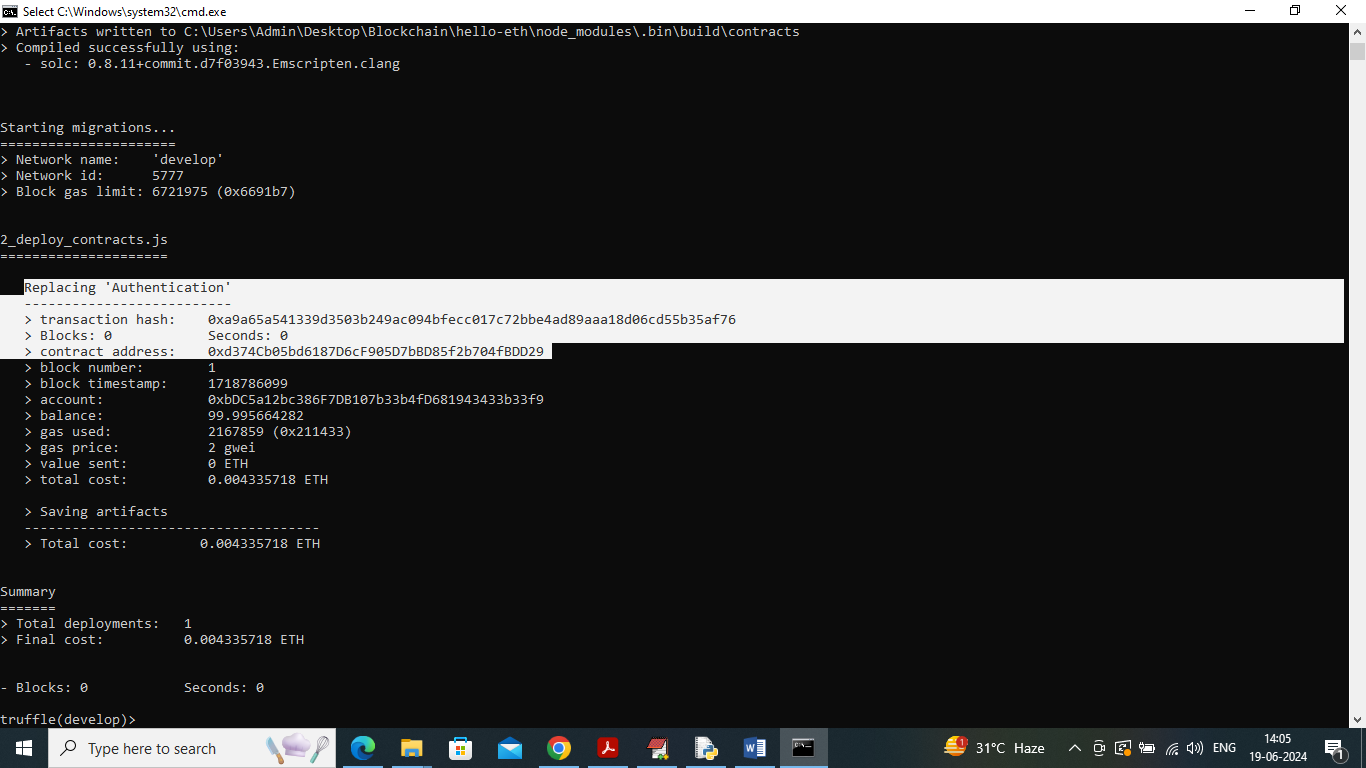
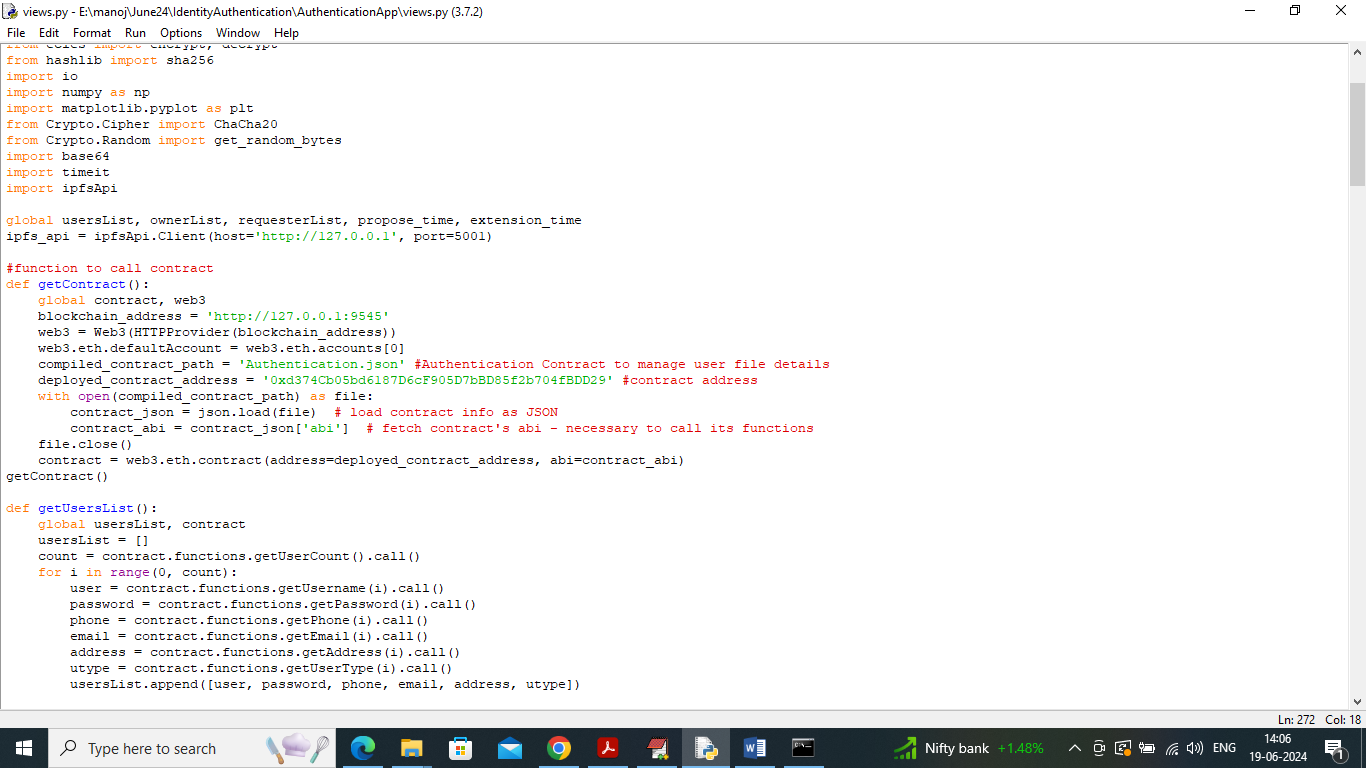
In above screen read red colour comments to know about attributes encryption and key generation using ECC and extension algorithm.

In propose paper main idea of using Blockchain is to avoid data tamper and leak without any attacks so Blockchain has inbuilt support for decentralized data storage (Blockchain record single data storage or transaction at multiple nodes) storage and verification. Blockchain store each record as block/transaction and associate each record with unique hash code and if any attacker or employee change data at any node then it will result into different hash code and verification get failed. So data in Blockchain cannot be alter in any manner.

To store and retrieve record in Blockchain we need to utilize smart contracts code which can be designed using SOLIDITY programming and in below screen showing SMART Contract code to manage REQUESTER and Data Owner details.



In above smart contract code defined functions to save and get data and now we need to deploy above contract in Blockchain ETHEREUM using below steps

1. First go inside ‘hello-eth/node-modules/bin’ folder and then find and double click on ‘runBlockchain.bat’ file to get below page
2. 
3. In above screen type command as ‘migrate’ and press enter key to get below page
4. 
5. In above screen in white colour text can see ‘Authentication’ contract deployed and we got contract address also and this address need to specify in python code to call contract to save and get data
6. 
7. In above screen read red colour comments to know about contract calling using contract address. In above screens can see contract deployed and running and let it run.

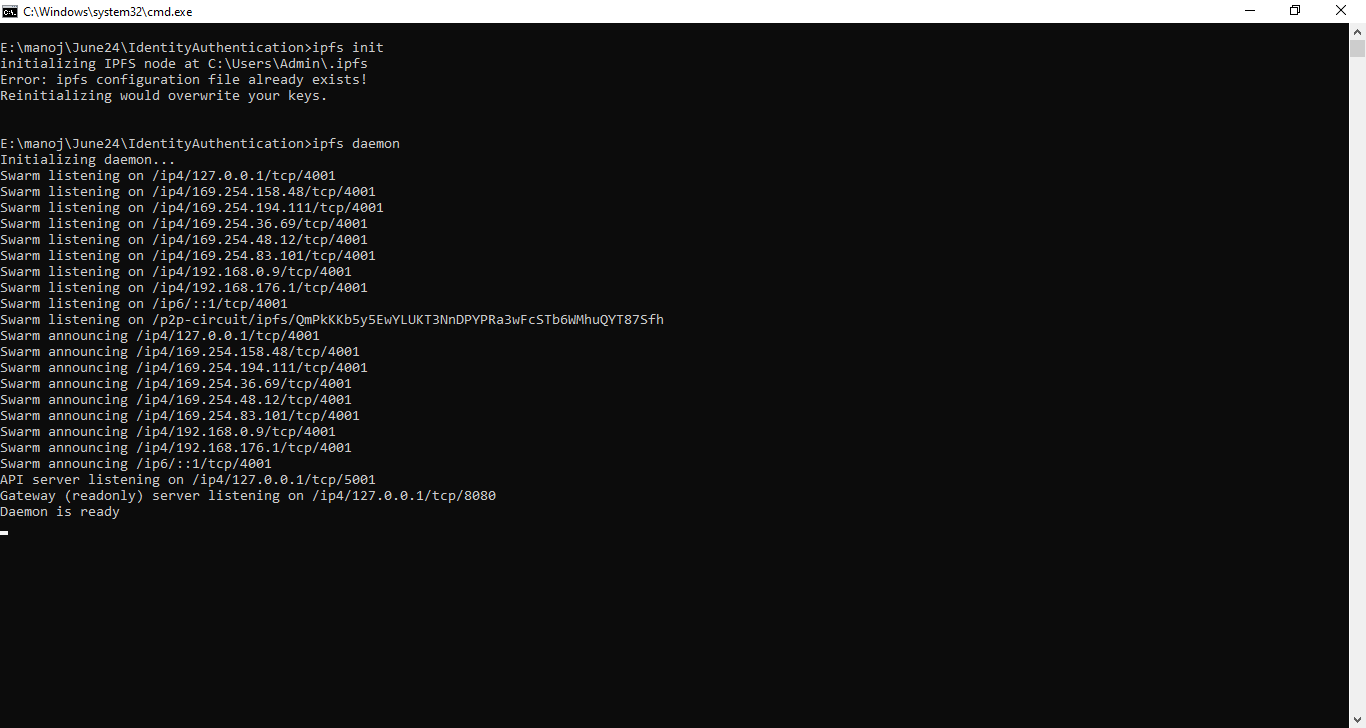
Modules Implementation

To implement this project we have designed following modules

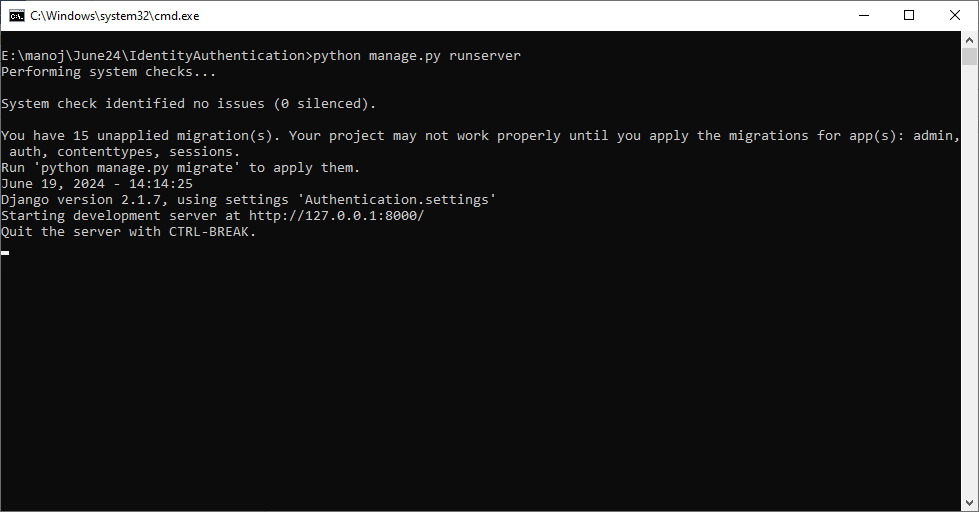
1. Admin Module: admin can login to system using username and password as ‘admin and admin’. After login admin will add all ‘Data Owner or citizens’ details. Admin will record only those owners whose has supported Address proof documents to upload. Admin can view all registered Data Owner details and can download their documents
2. User Register: using this module ‘Data Owner and Data Requester’ will register with the application. All requester attributes like Name, Contact number, address will get encrypted and saved in IPFS and Blockchain for future verification to access data.
3. Data Owner Login: data owner can login to system and then can see list of Requester along with their generated IPFS hash code which stored in Blockchain. Data owner can retrieve all hash code to verify their identity and then allow them to access data
4. Requester Login: Requester login to system and then Blockchain will verify Requester identity and upon successful identify verification then application allow ‘Requester User’ to access Data Owner details.

SCREEN SHOTS

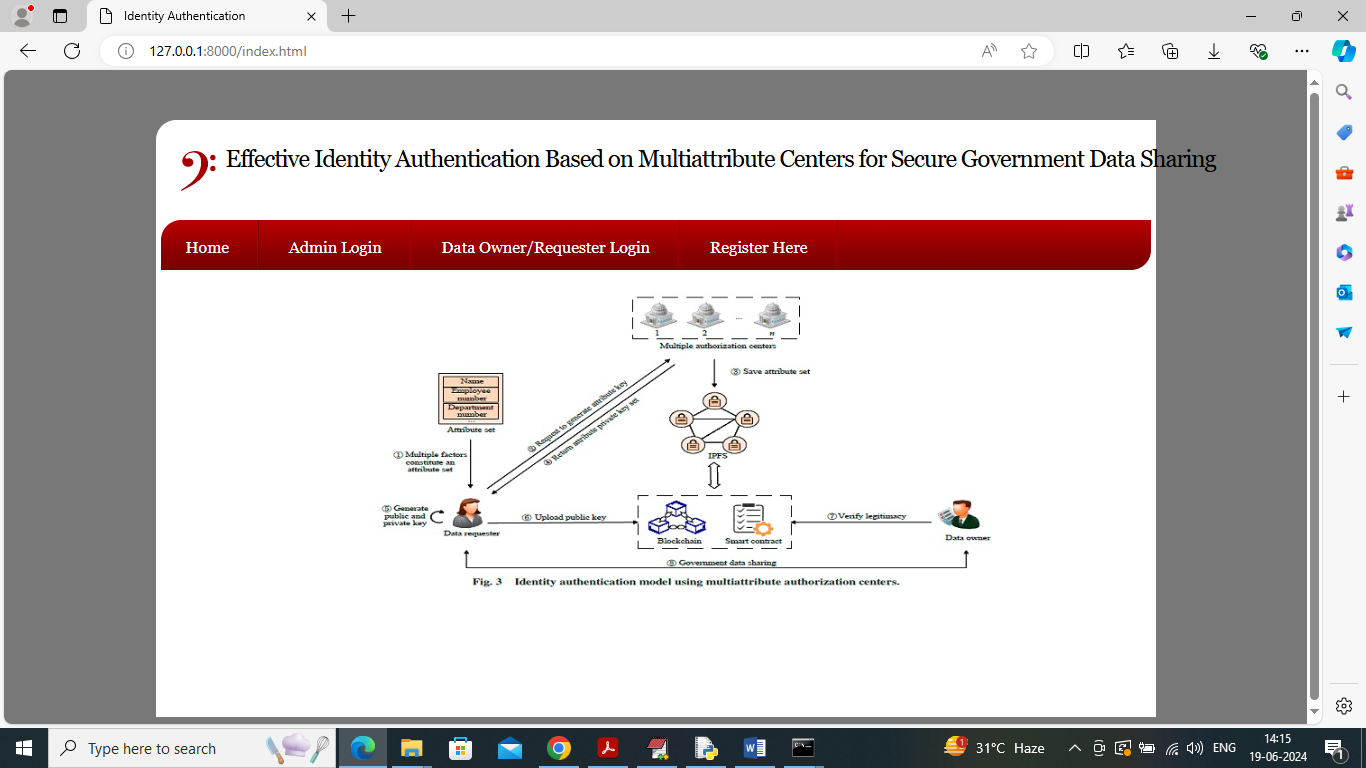
Double click on ‘Start\_IPFS.bat’ file to start IPFS server and get below page



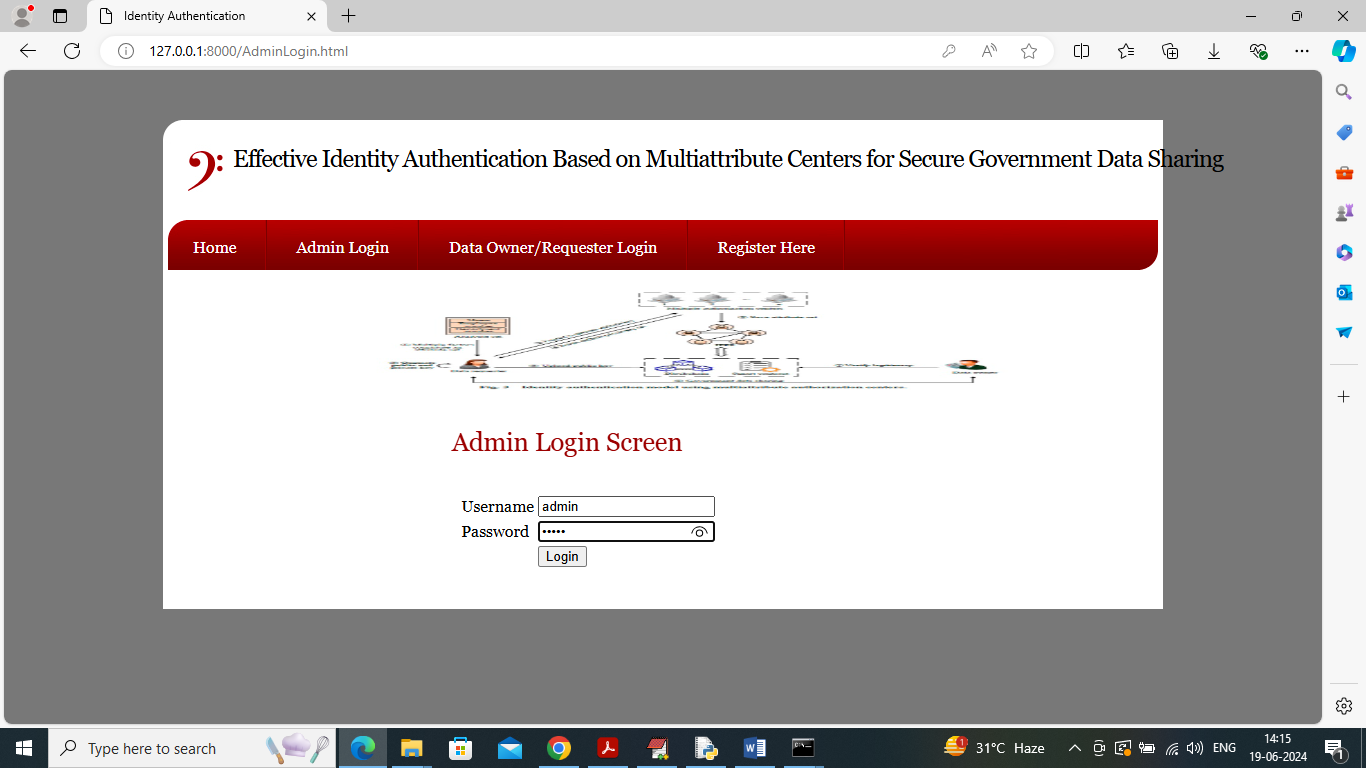
In above screen IPFS server started and now double click on ‘runServer.bat’ file to start python web server and get below page



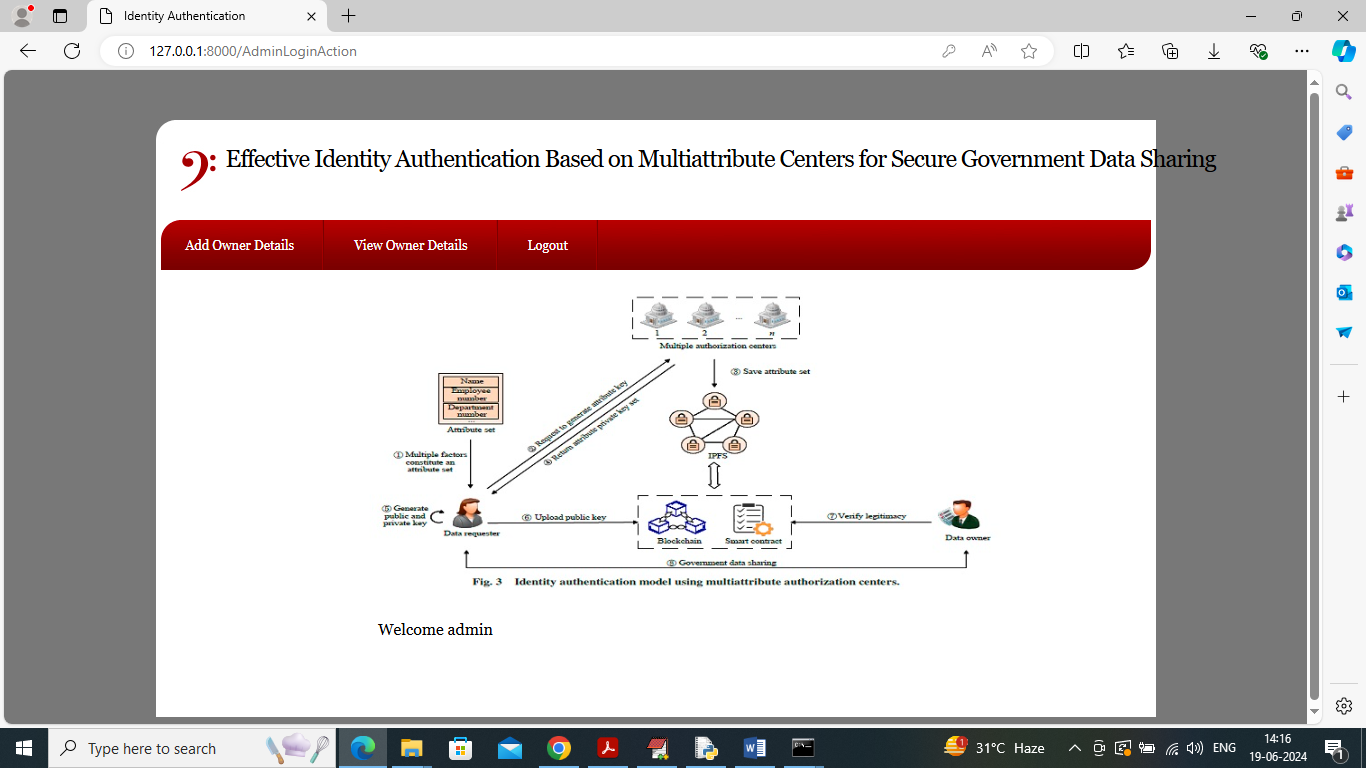
In above screen python server started and now open browser and enter URL as <http://127.0.0.1:8000/index.html> and press enter key to get below page



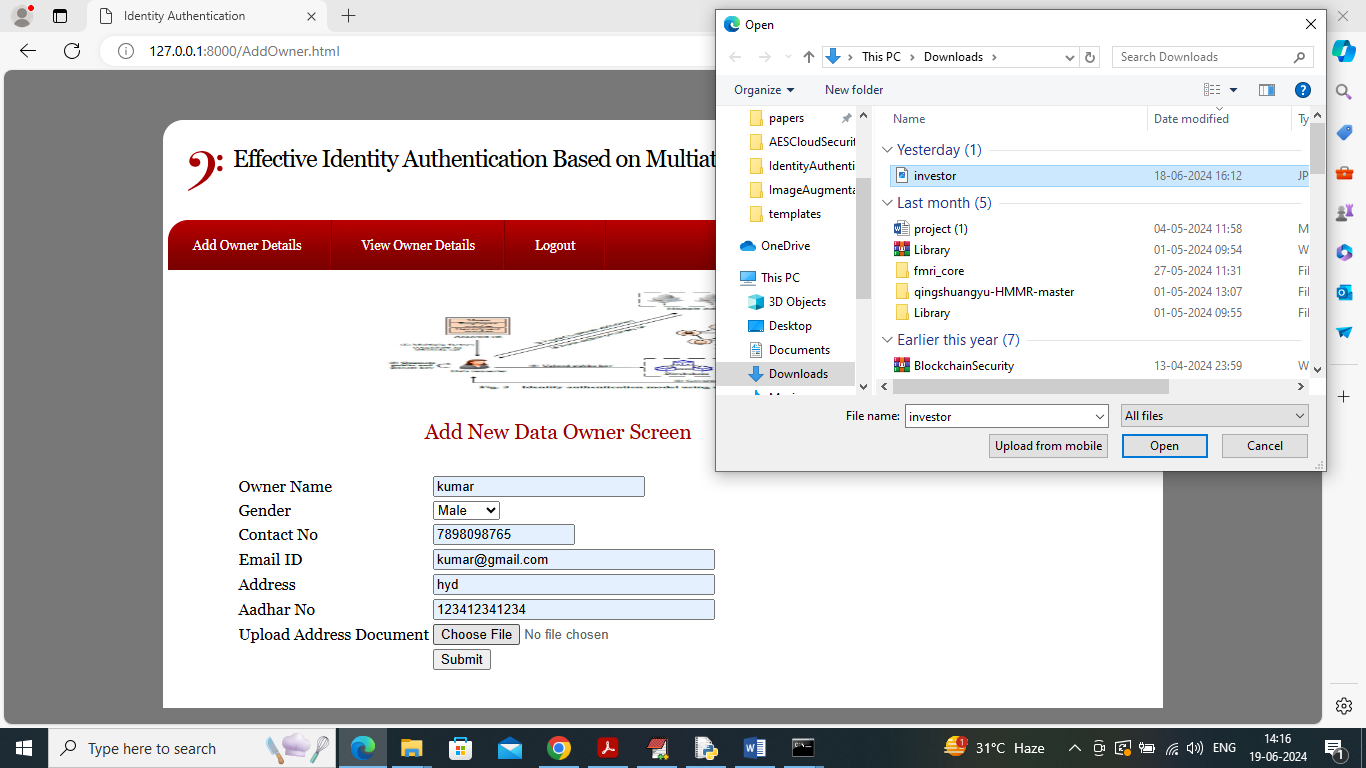
In above screen click on ‘Admin Login’ link to get below login screen



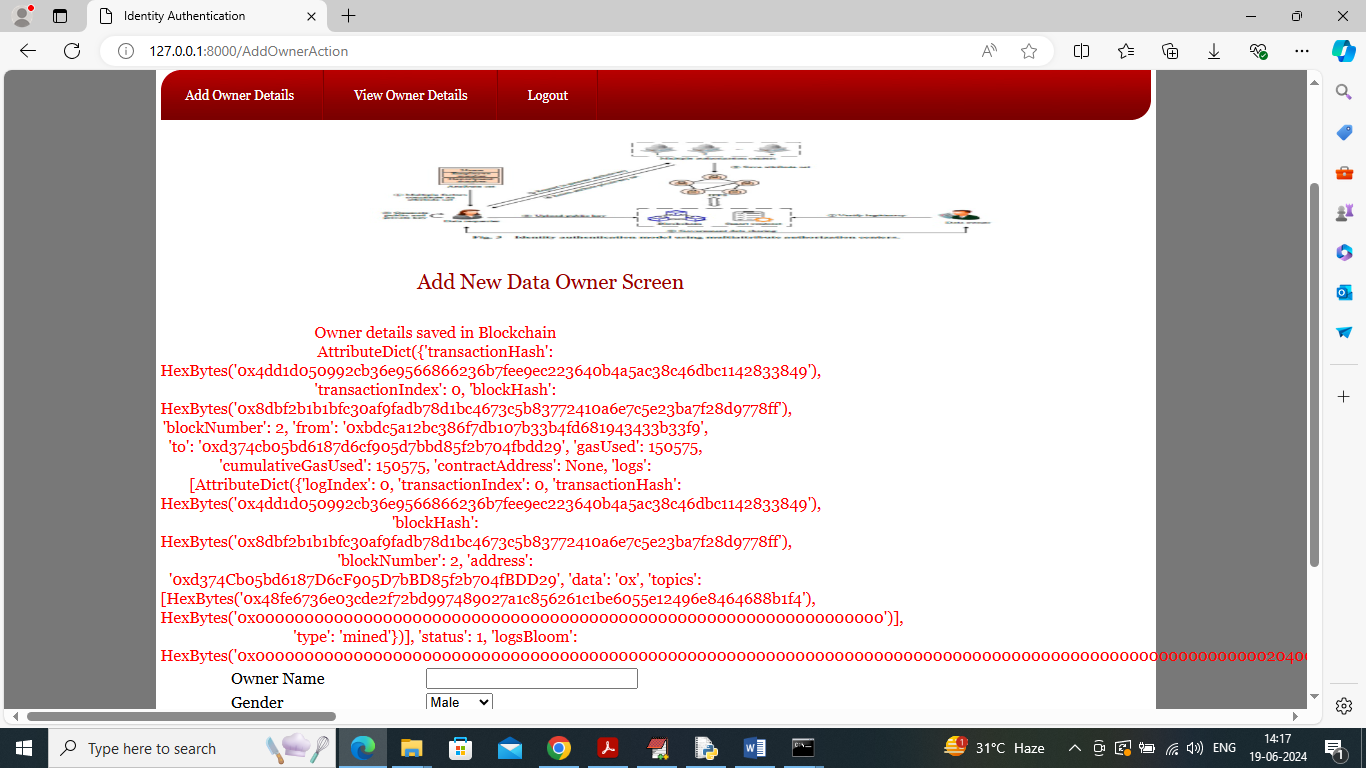
In above screen admin is login and after login will get below page



In above screen admin can click on ‘Add Owner Details’ link to get below page



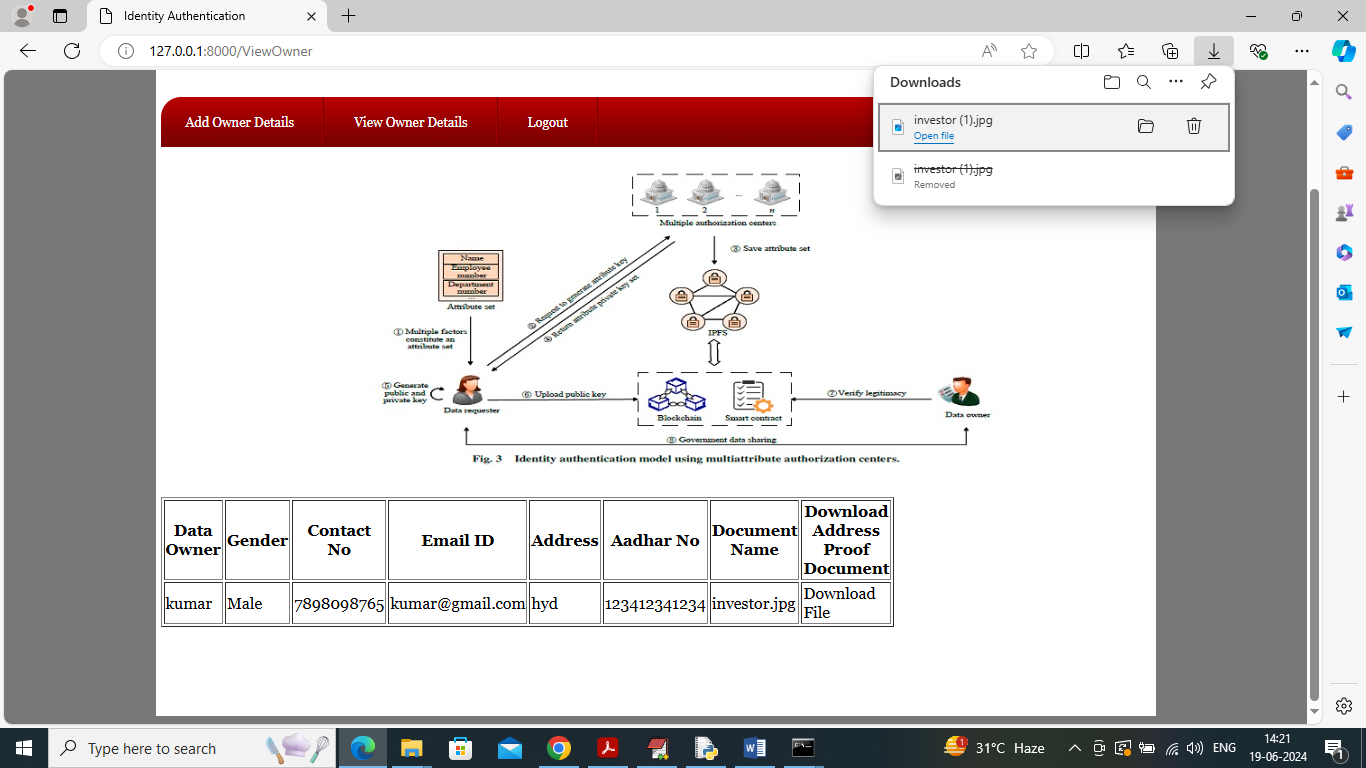
In above screen admin adding ‘Data Owner’ details and then uploading supporting address proof document and then press button to saved data owner details in Blockchain and get below page



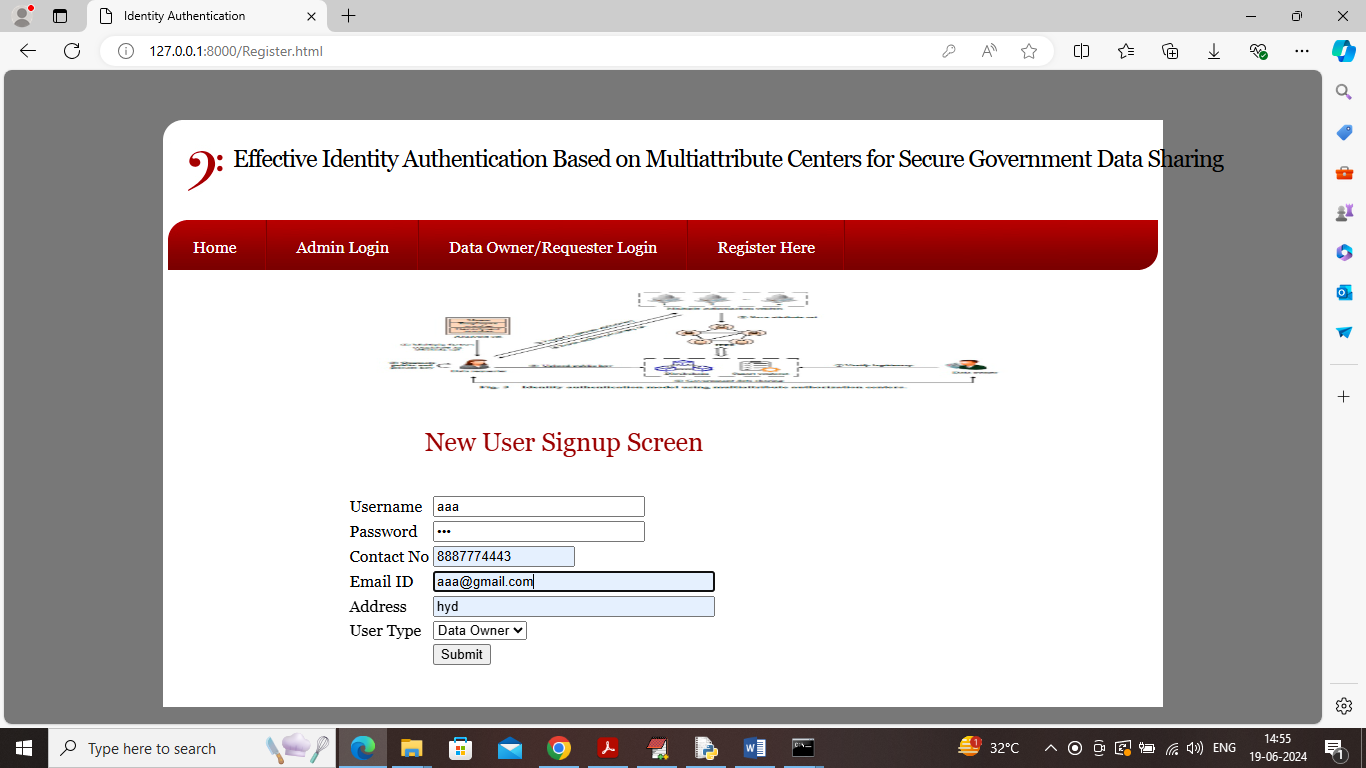
In above screen can see data owner details saved in Blockchain and then I am displaying entire logs obtained from Blockchain and this log contains information like Block No, Transaction No, Transaction hash code and many other details. Similarly by using above screen admin can add any number of ‘Data Owner’ details. Now click on ‘View Owner Details’ link to view all registered owner and get below page



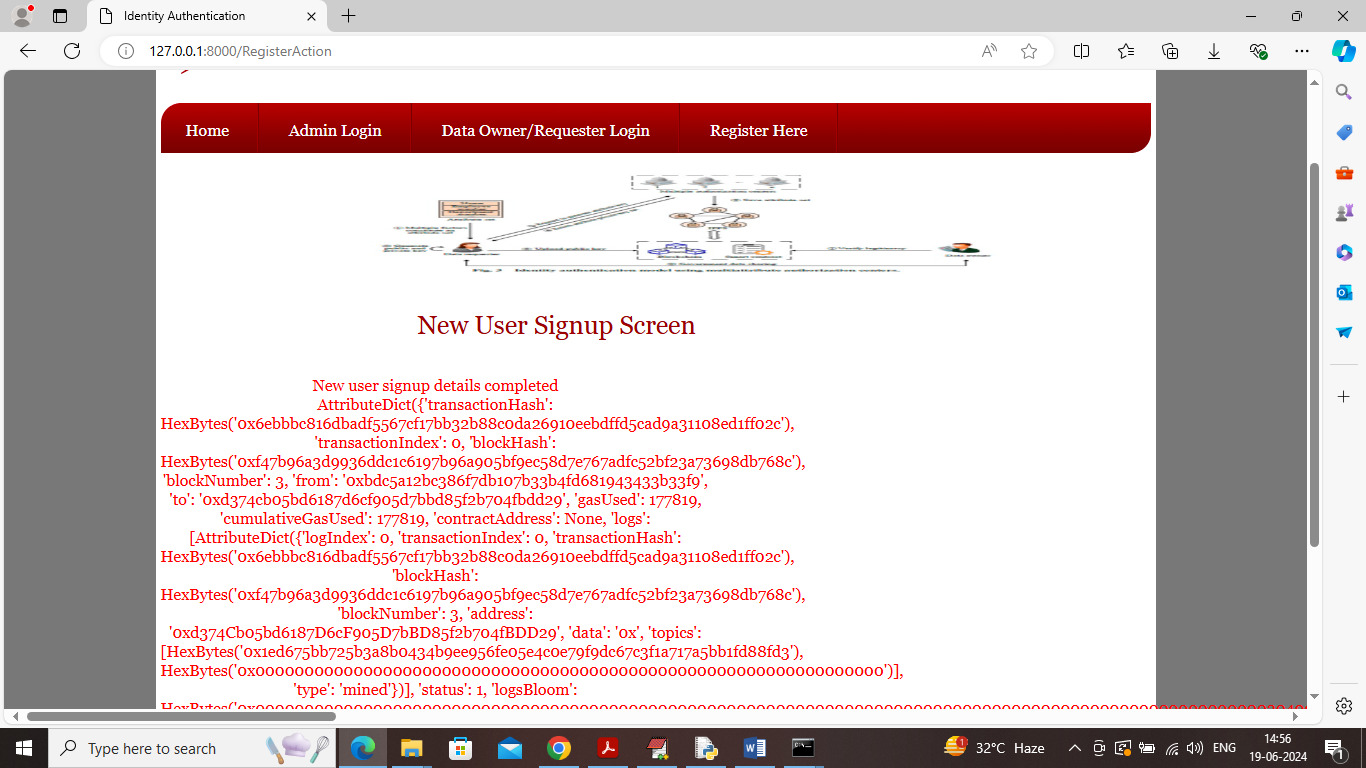
In above screen admin can see list of added Blockchain owner details and can click on ‘Download File’ link to download file and get below page



In above screen can see file is downloaded in top right browser side and now logout and sign up new user.



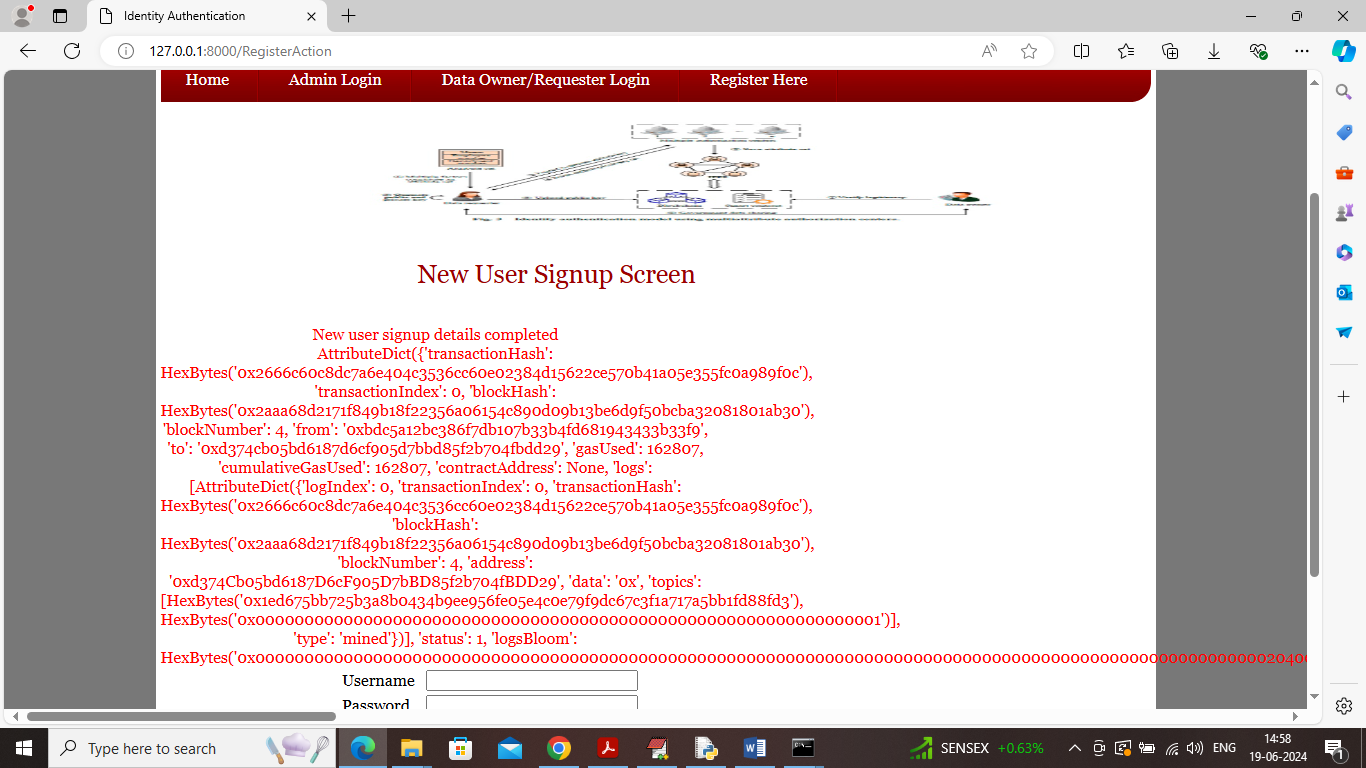
In above screen adding data owner details and then press button to save details in IPFS and Blockchain and then will get below output



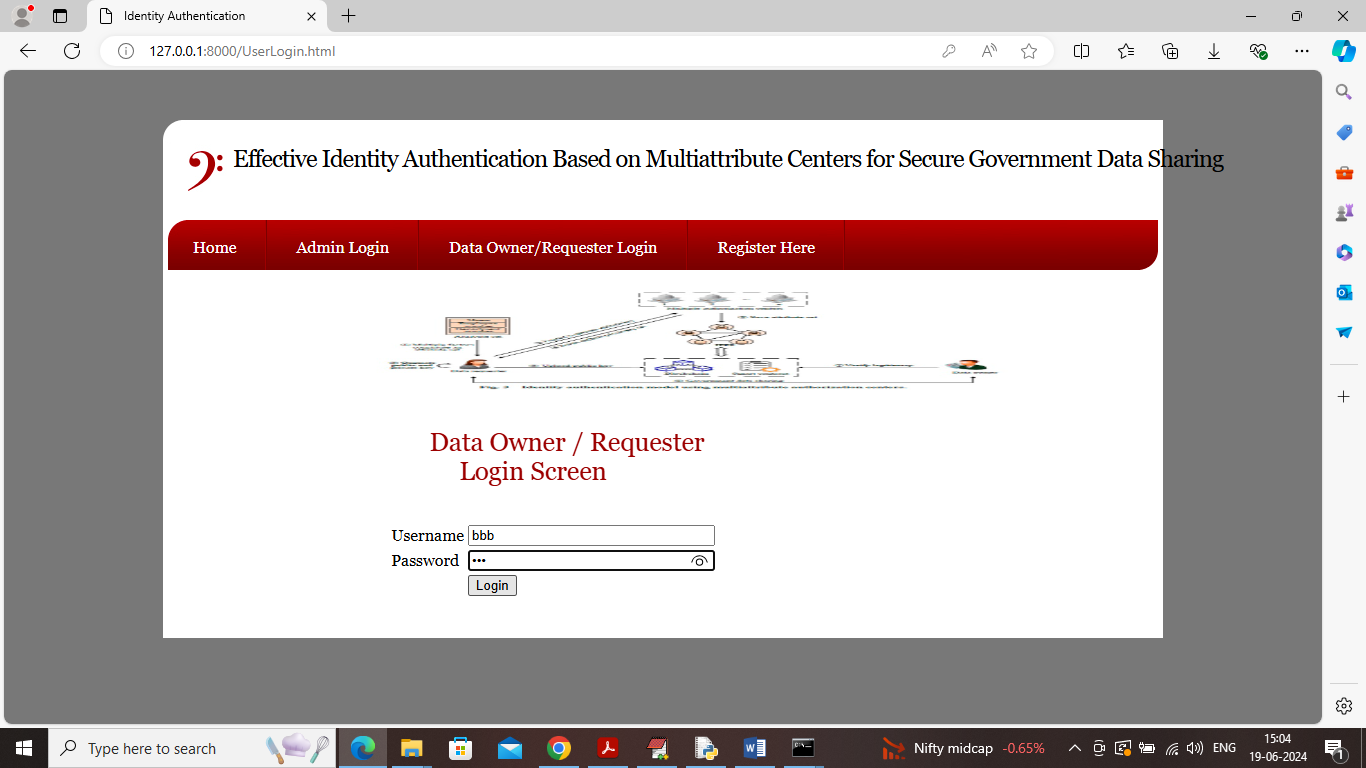
In above screen data owner details added in Blockchain and now similarly adding ‘Requester’ details also



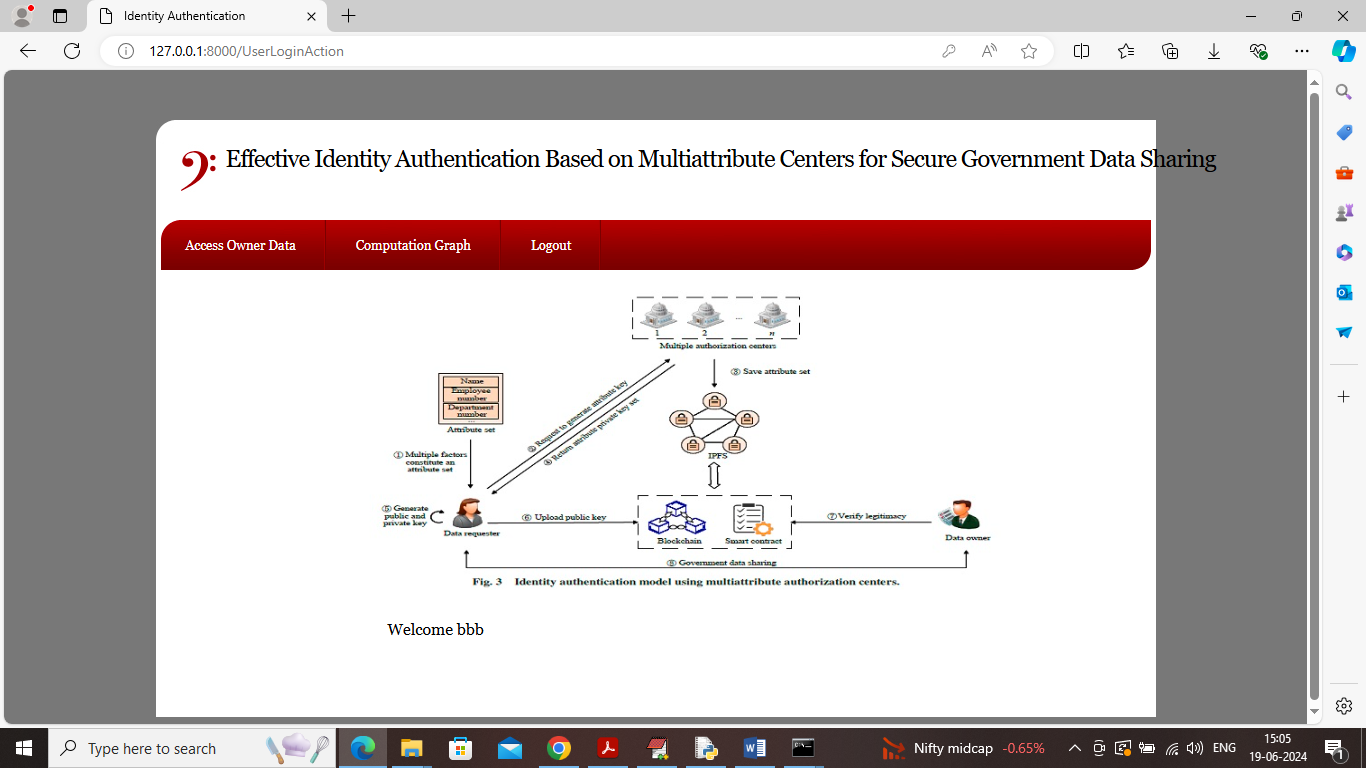
In above screen adding requester details and then press button to get below page



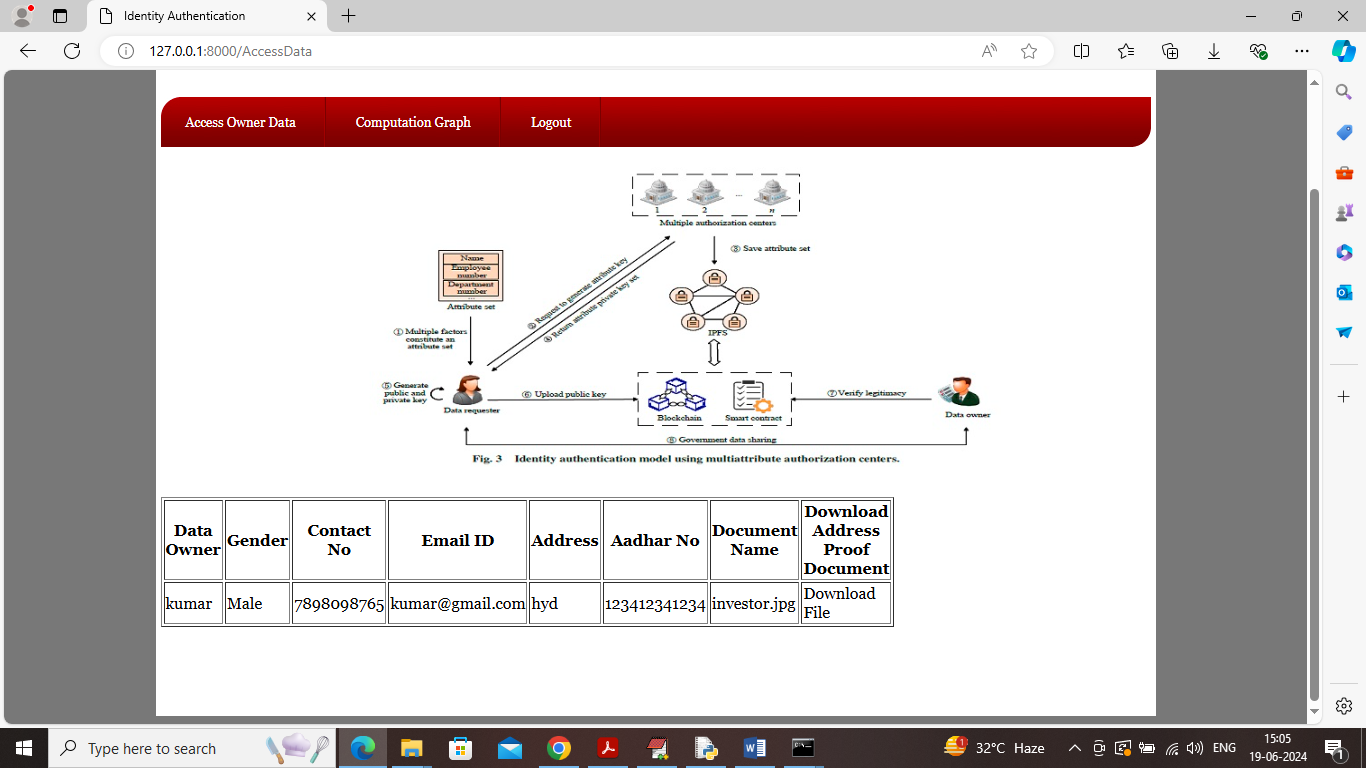
In above screen requester details added to IPFS and Blockchain and now click on ‘Data Owner/Requester Login’ link to get below screen



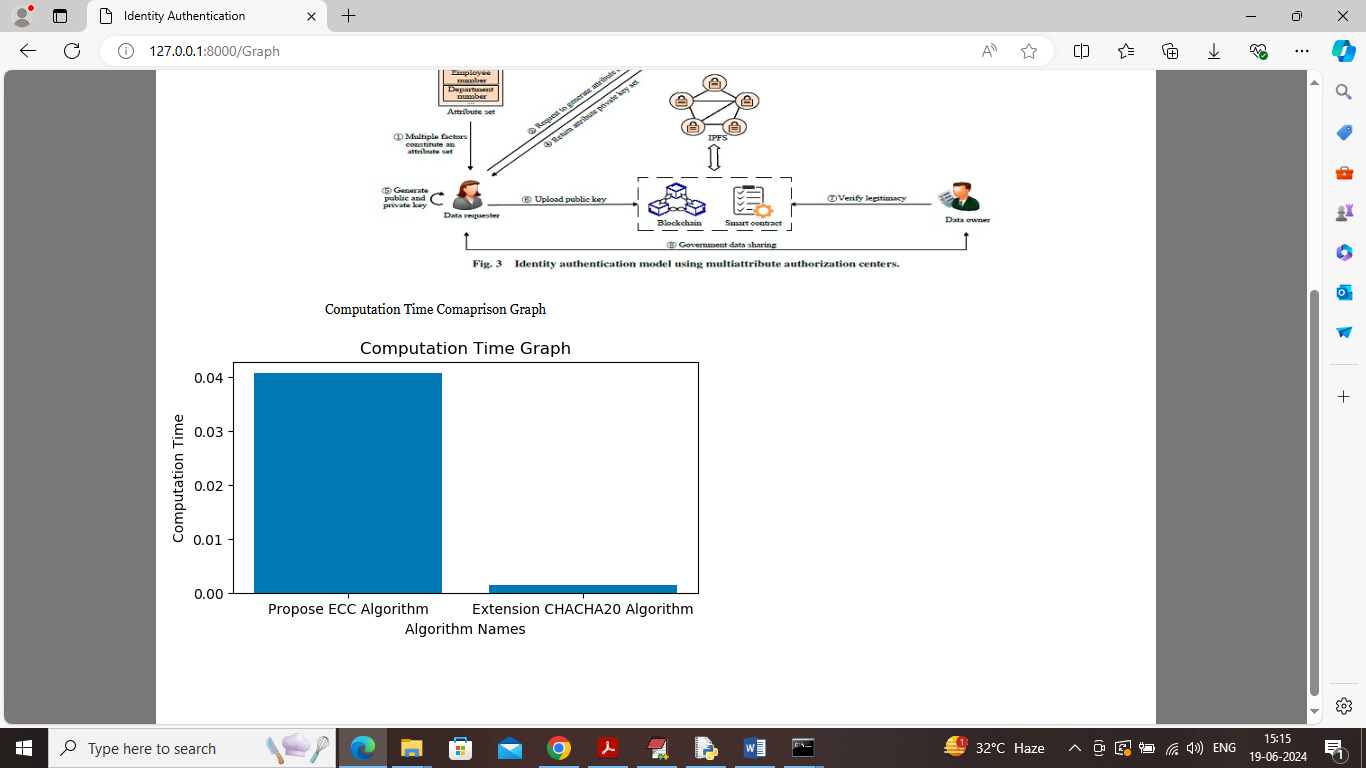
In above screen ‘Requester’ is login and after login will get below page



In above screen Requester can click on ‘Access Owner Data’ link to access all available Data Owner details and get below page



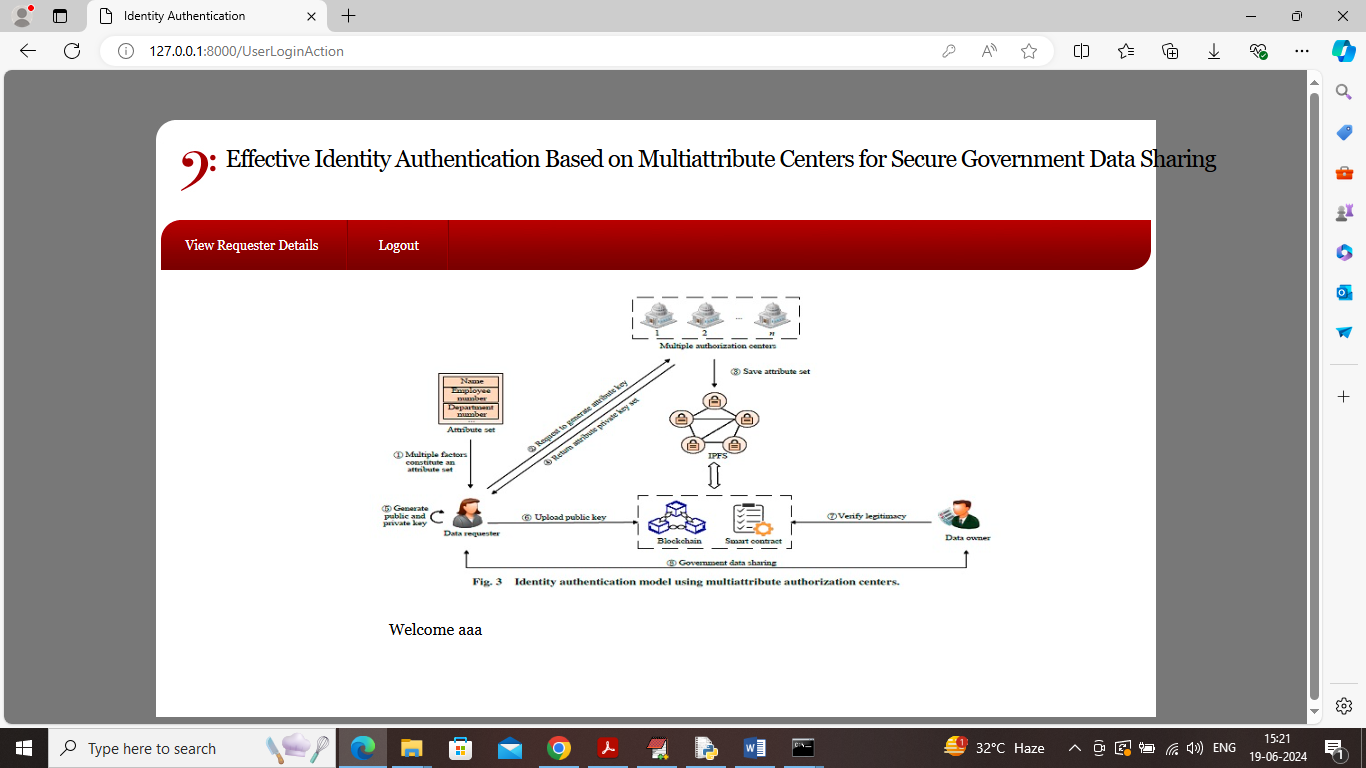
In above screen ‘Requester’ can access all details of ‘Data Owner’ and only those Requester whose keys are available in Blockchain can able access data and now click on ‘Comparison Graph’ link to get below graph



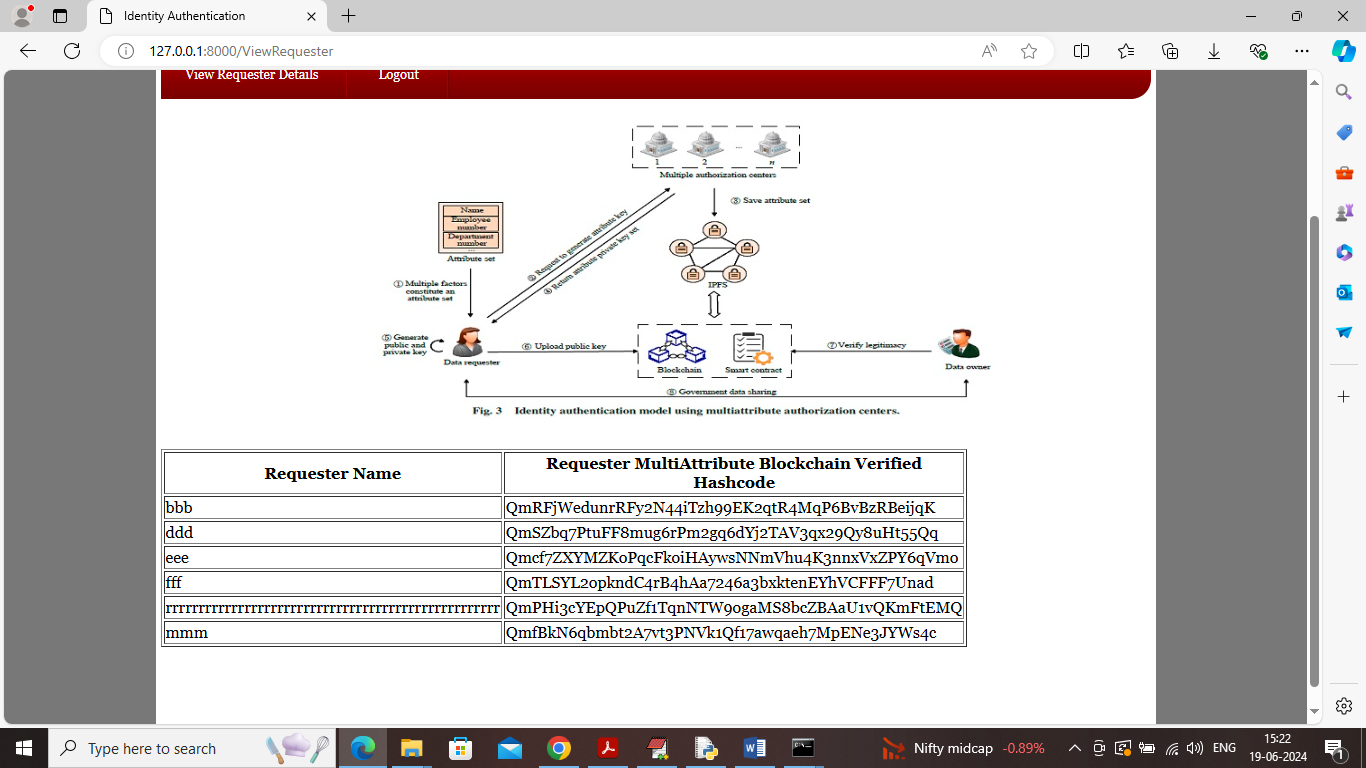
In above screen x-axis represents algorithm names and y-axis represents computation time and in both algorithms extension CHACHA20 took less computation compare to propose algorithm and now logout and login as ‘Data Owner’ to view all verified Requester Hash code.



In above screen ‘Data Owner’ is login and after login will get below page



In above screen ‘Data Owner’ can click on ‘View Requester Details’ link to get below page



In above screen ‘Data Owner’ can view different Requester details along with IPFS hash code to verify with Blockchain.

Similarly by following above screen Government can share secured data with Requester.