**Dynamic AES Encryption and Blockchain Key Management: A Novel Solution for Cloud Data Security**

In cloud all user’s data will be maintained by single centralized server and each user file is encrypted by using different keys and all keys will be stored at single centralized server and if this server hack then hacker can get all keys to encrypt all files. Cloud employees can also alter keys stored at centralized server and there is no possible ways to detect such alteration.

To overcome from above issue author of this paper suggesting to employ Blockchain to manage and store AES encryption keys as Blockchain has inbuilt support of decentralized key storage (keys will be saved at multiple nodes and if keys at one node hack or changes then verification will get failed from all other nodes. So hacker needs to hack all Blockchain nodes which is impossible. Blockchain also support tamper proof data storage which means Blockchain store each record as Block/transaction and associate each block with unique hash code and this hash code will get verify and if any record change then result into different hash code which leads to failed verification and due to this reason Blockchain consider as Tamper proof.

In propose paper author using Dynamic AES keys which can be generated by using combination of File hash code and Block hash code. Both file and block hash code will be XORED to generate AES dynamic key and then file will be encrypted using AES dynamic key and ECC (elliptic curve cryptography) algorithm. Dynamically keys will be generated for each file and will be stored at Decentralized Blockchain so hackers or malicious employees cannot able to steal or tamper keys.

In propose paper consists of two algorithms

In algorithm 1 hash code will generated for both file and last block and then this hash will XORED to generate keys. This keys will be generated for each file block and first block will be started at empty Blockchain location and remaining blocks or files counter incremented by 1.

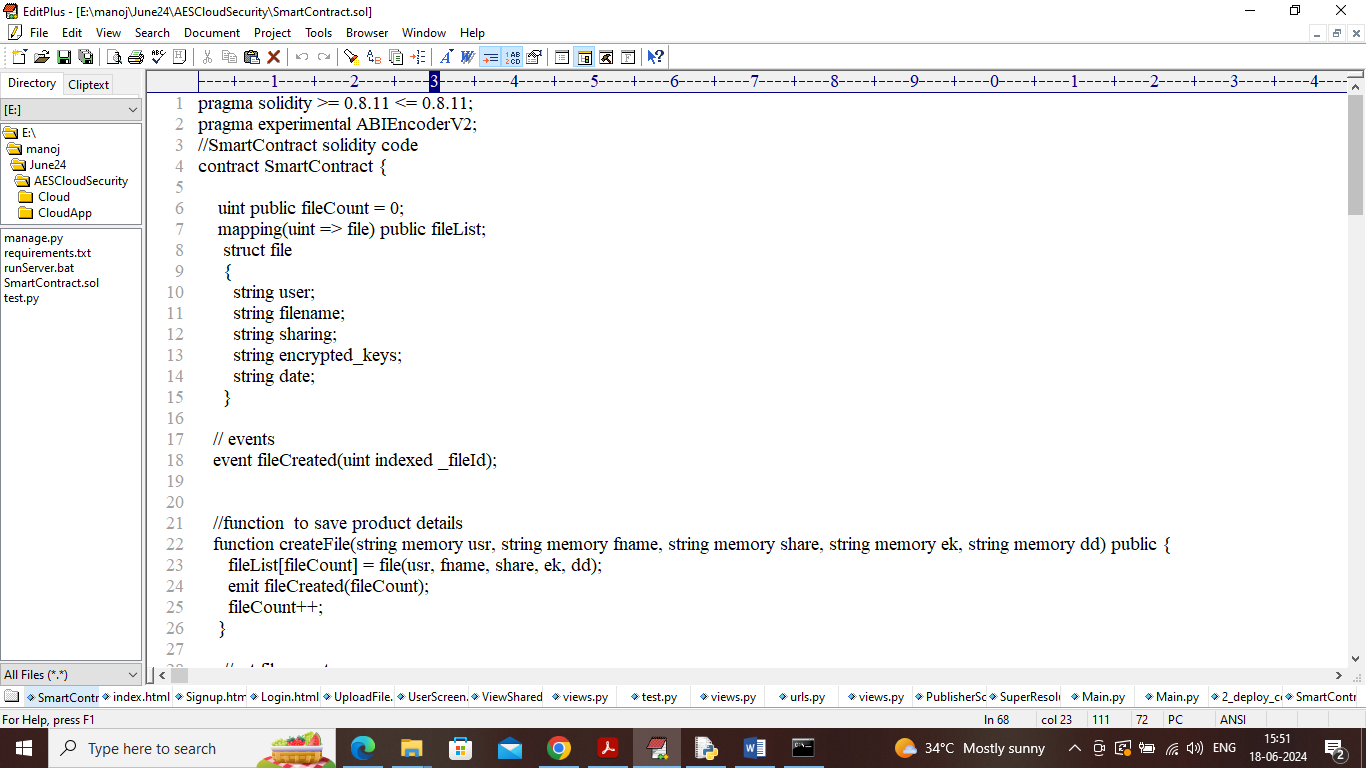
In algorithm2 Generated AES keys and ECC algorithm will be utilized to encrypt file and then this encrypted data will stored in cloud.

In propose paper to secure cloud sharing data author has used images and then encrypted those images using AES key and ECC algorithm and during encryption author has calculated BIT density which represents how much image is uniform shape. In propose work author found original image is not in uniform shape and but after encryption with dynamic keys image has come to uniform share and then calculate BIT density. Image with uniform BIT density shape will have high sensitivity compare to un-uniform shape.

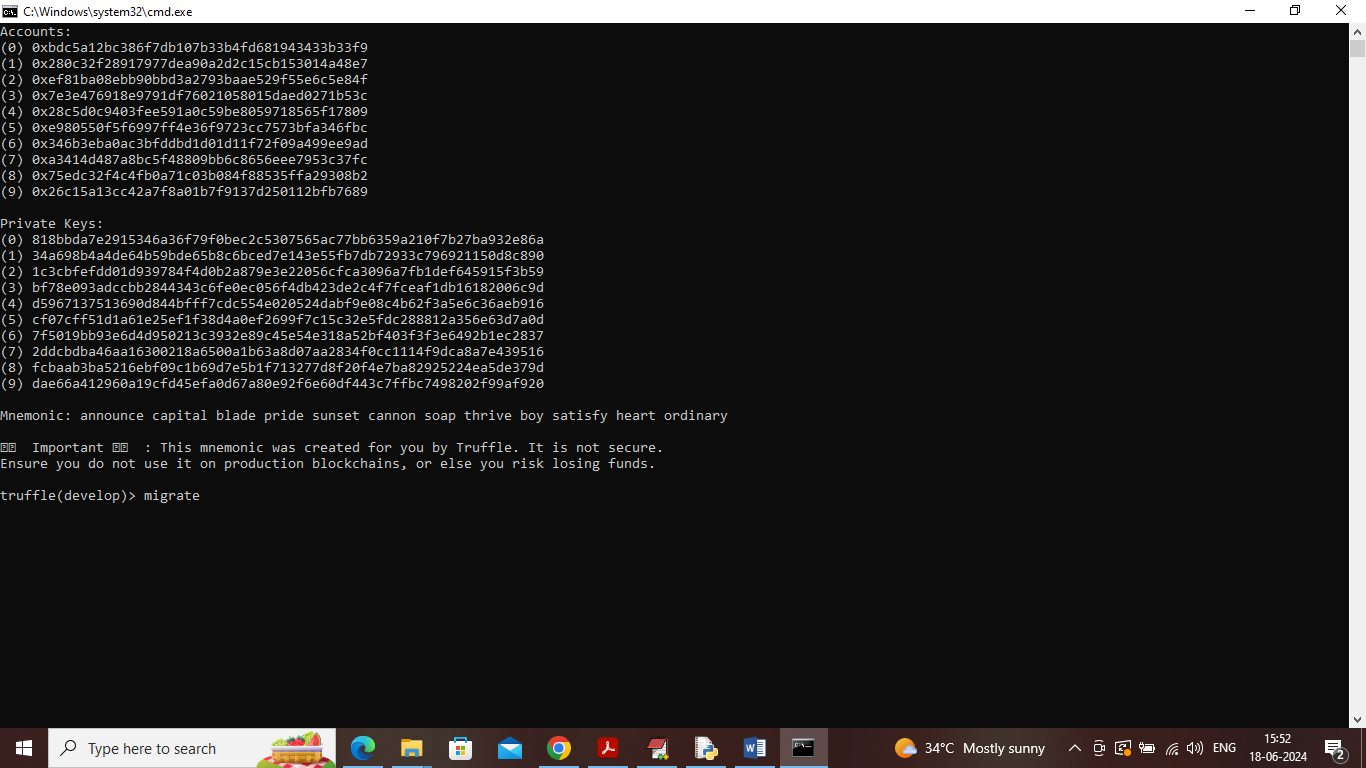
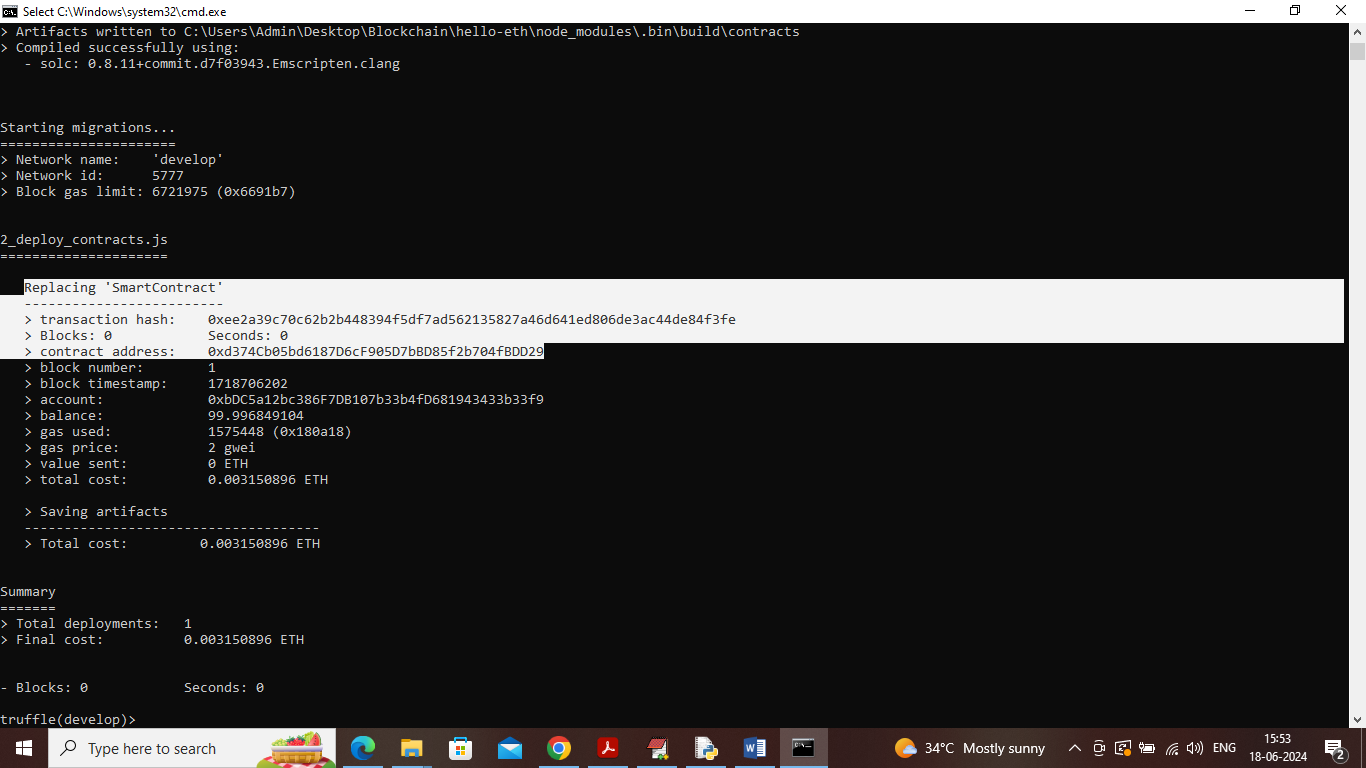
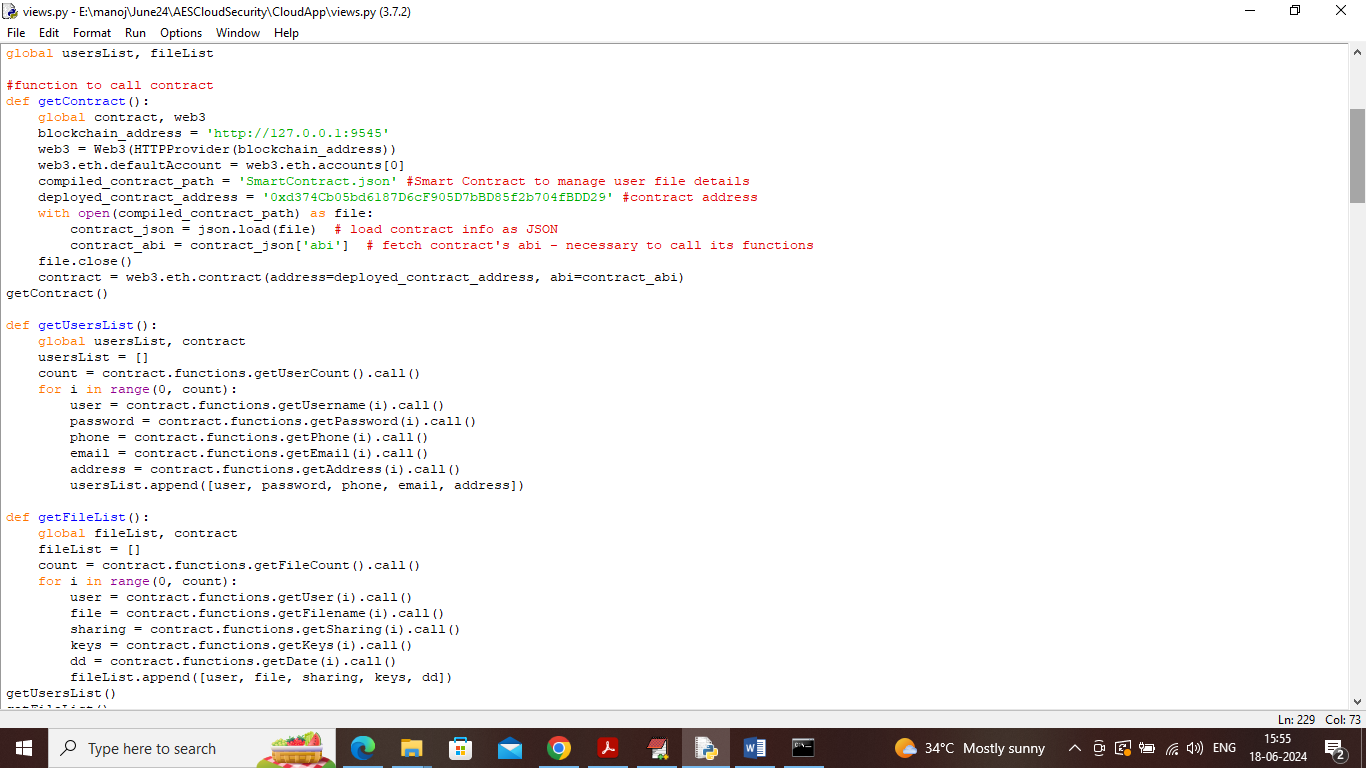
Extension Concept 1: In propose paper author has used only images but as extension 1 we have make this algorithm to work with any file

Extension Concept 2: In propose paper author used AES or ECC algorithm which is computationally very heavy and to reduce this computation cost we have implemented another advance algorithm called CHACHA20 which is more secure and lightweight in computation. We have shown experiment between AES and CHACHA20 algorithm using computation cost.

We can save and get data from Blockchain using Smart Contracts which will contains function to be called for data management. To save keys we have designed following contract



In above contract we have defined functions to manage file and keys details and this contract need to be deployed in Blockchain ETHEREUM tool using below steps

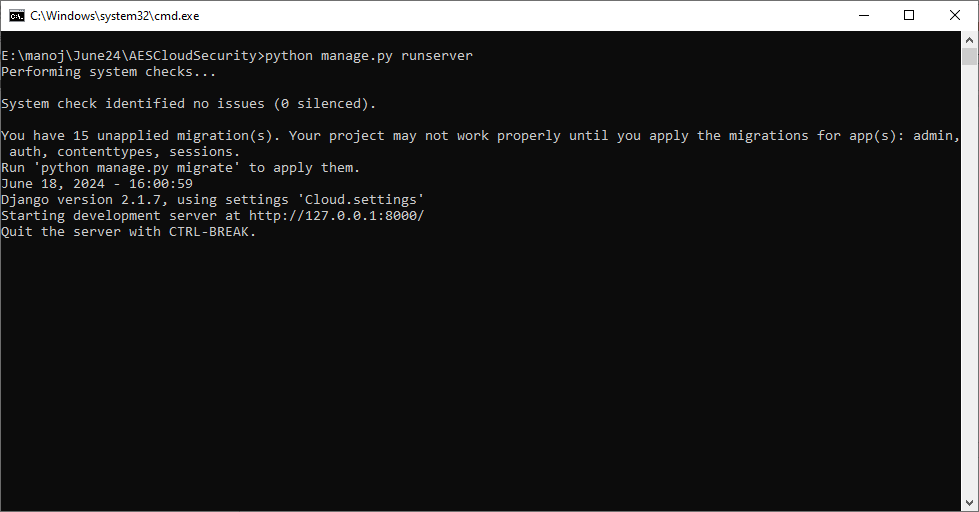
1. First go inside ‘hello-eth/node-modules/bin’ folder and then look and find for ‘runBlockchain.bat’ file to get below page
2. 
3. In above screen Blockchain started with default private keys and accounts and now type command as ‘migrate’ and press enter key to deployed contract and get below page
4. 
5. In above screen ‘Smart Contract’ deployed and got contract address also and this address need to specify in python code to call Smart contract functions to manage cloud sharing keys data. In below screen showing python code calling above contract
6. 
7. In above screen read red colour comments to know about contract calling using python code. In above screens we have seen contract deployed and running and let it run.

To implement this project we have designed following modules

1. User Sign up: new user can sign up to application and all users get saved in Blockchain
2. User Login: user can login to system
3. Upload File: user can upload any type of file by selecting desired sharing users and then AES dynamic keys will be generated using file and block hash code and then get encrypted with keys and ECC algorithm.
4. Access File from Cloud: all shared users can see list of files available in cloud and then can download desired file by clicking on link. System will collect AES dynamic keys form Blockchain and then employ ECC algorithm to decrypt file
5. Comparison Graph: using this module we are encrypting sample image using AES and ECC algorithm and then encrypting with CHACHA20 extension algorithm and then displaying computation time with all algorithms. We are calculating sensitive values of existing algorithm and propose algorithm as comparison graph

SCREEN SHOTS

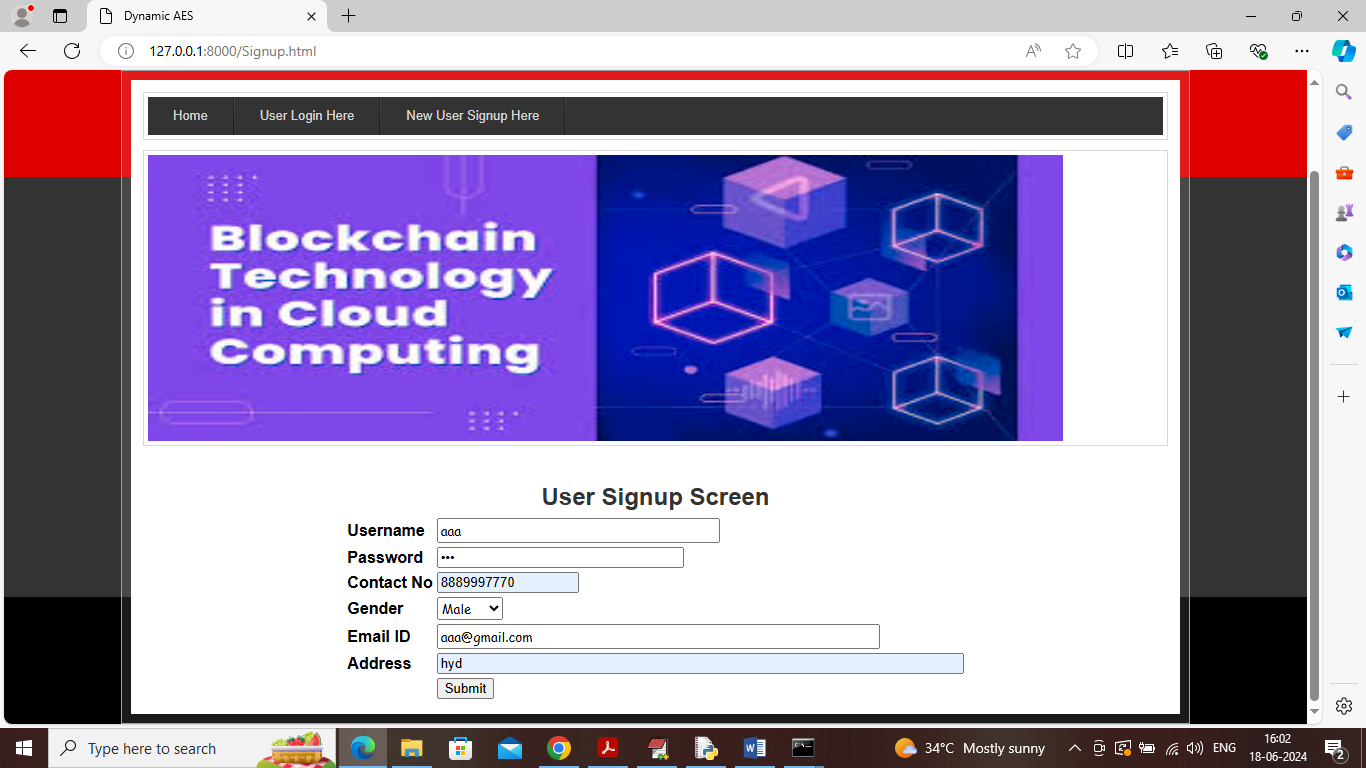
To run project double click on ‘runServer.bat file to start python server and get below page



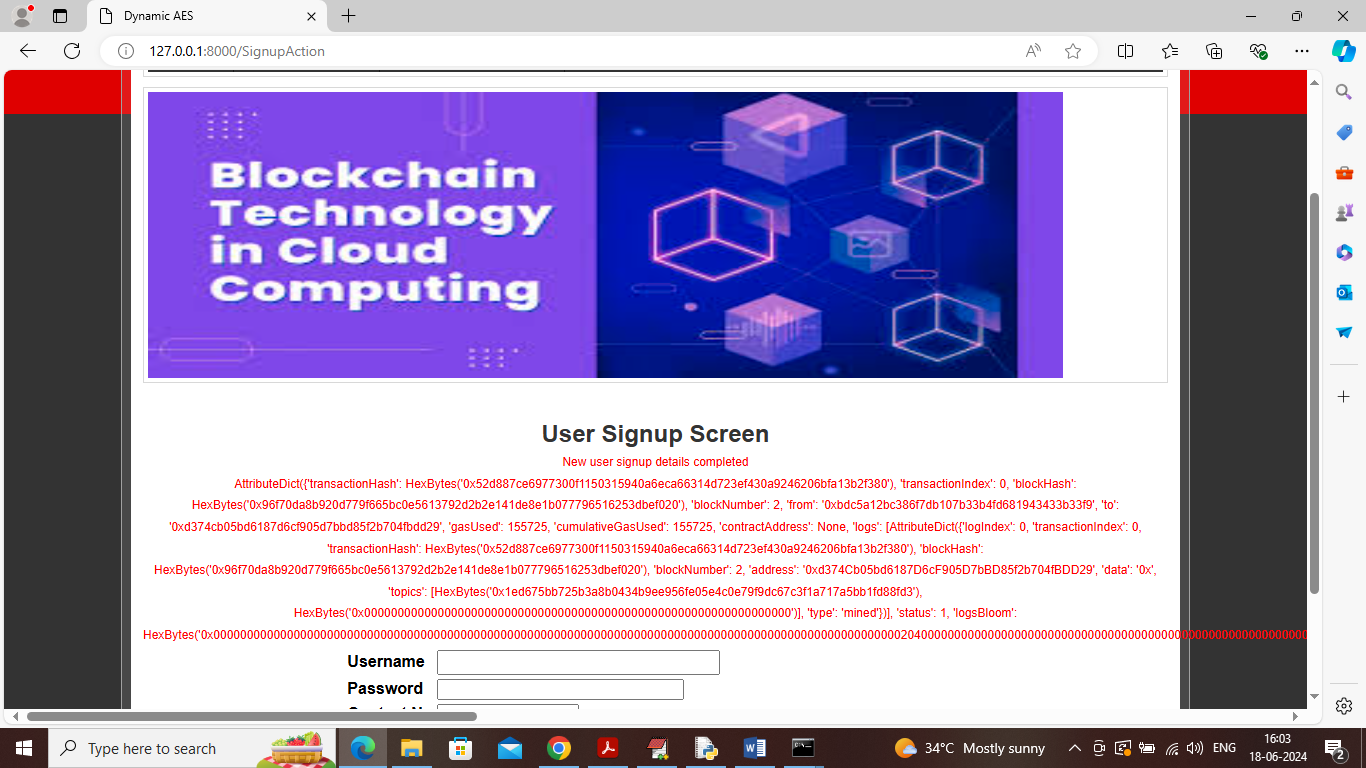
In above screen python server started and now open browser and enter URL as <http://127.0.0.1:8000/index.html> and press enter key to get below page



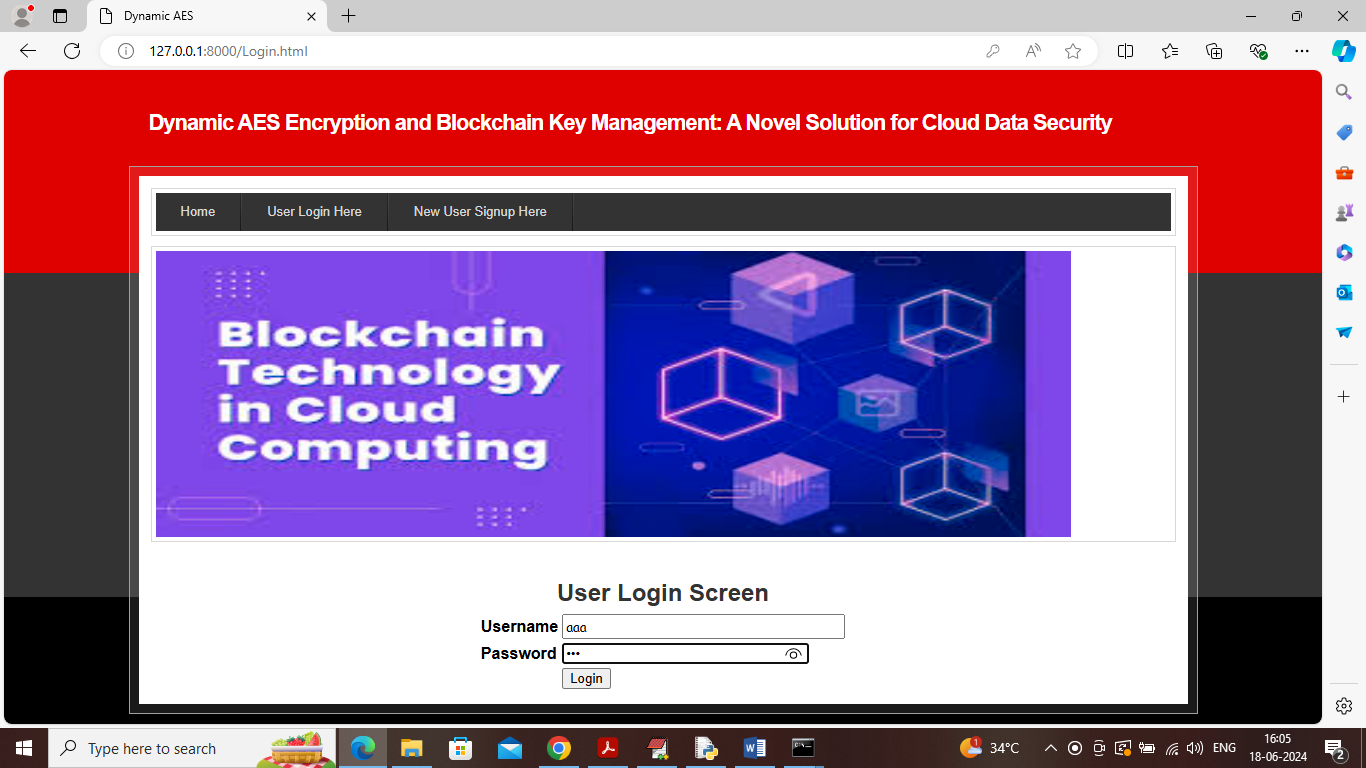
In above screen click on ‘New User Sign up’ link to get below page



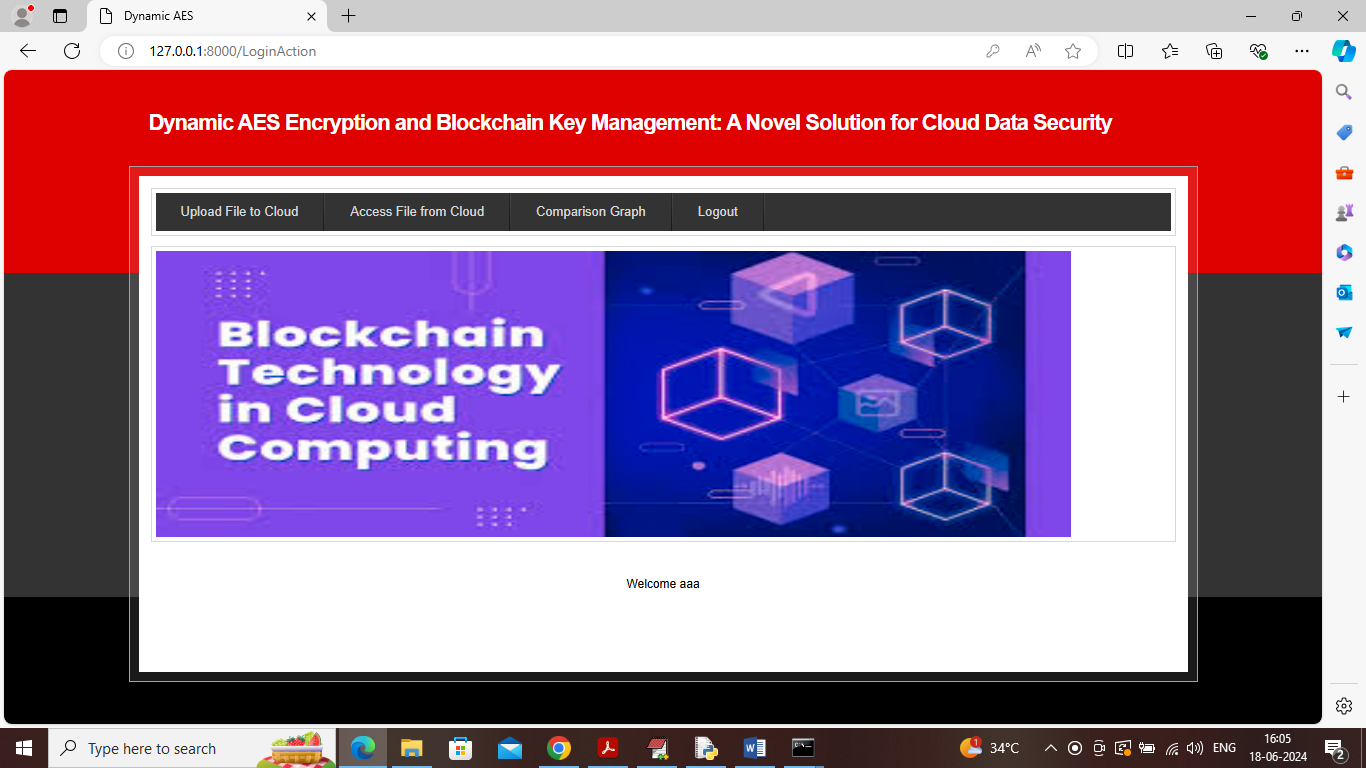
In above screen user entering sign up details and then press button to saved details in Blockchain and then will get below page



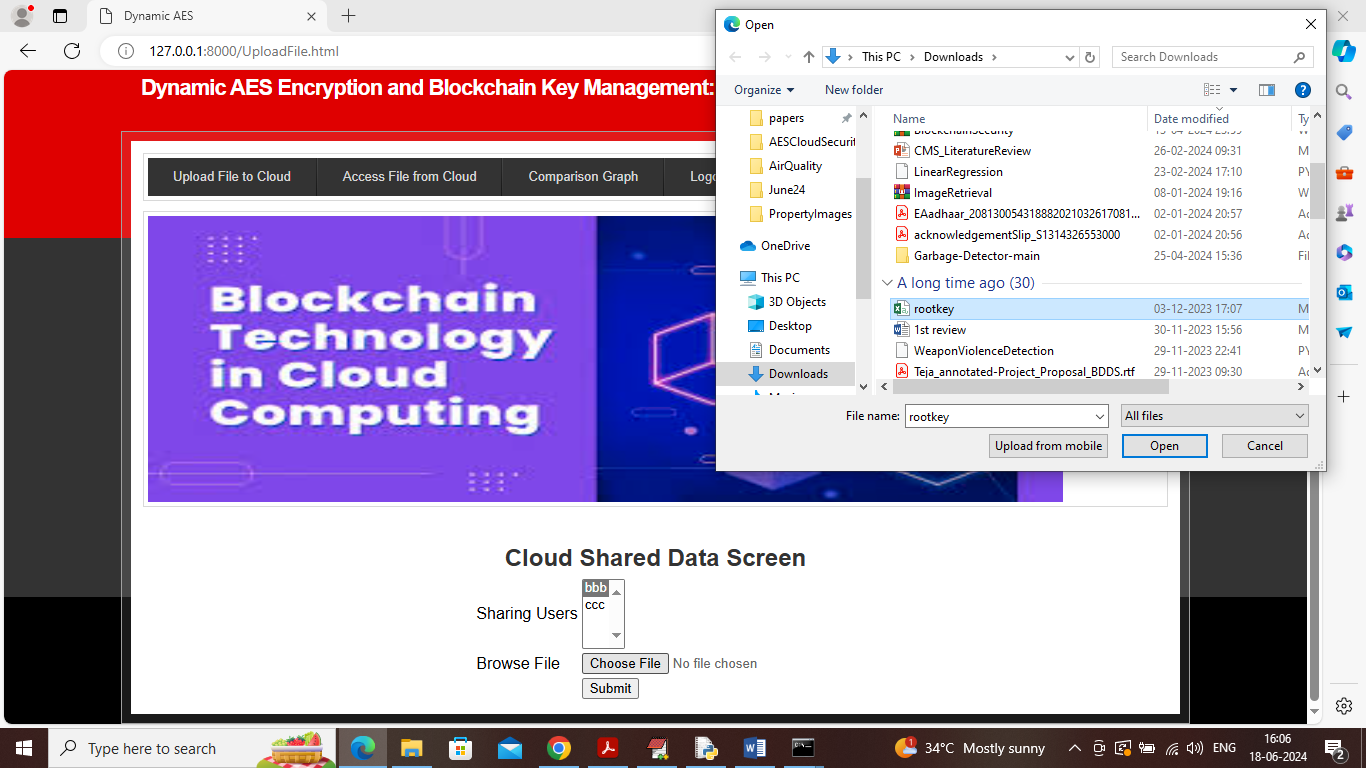
In above screen can see user details saved in Blockchain and then displaying all log data obtained from Blockchain after storage and can see details like Block No, Transaction No, Gas value, hash code and many other detail. Similarly you can saved any number of users and now click on ‘User Login’ link to get below page



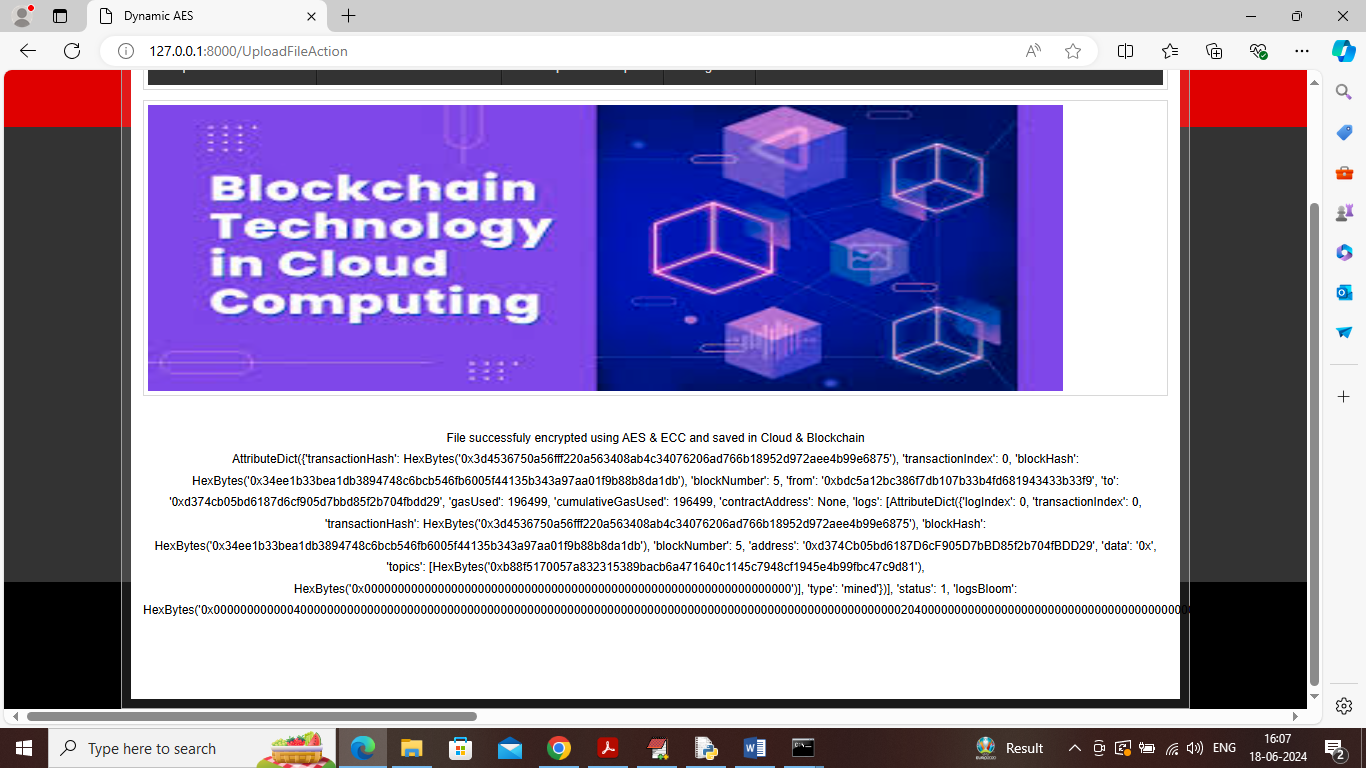
In above screen user is login and after login will get below page



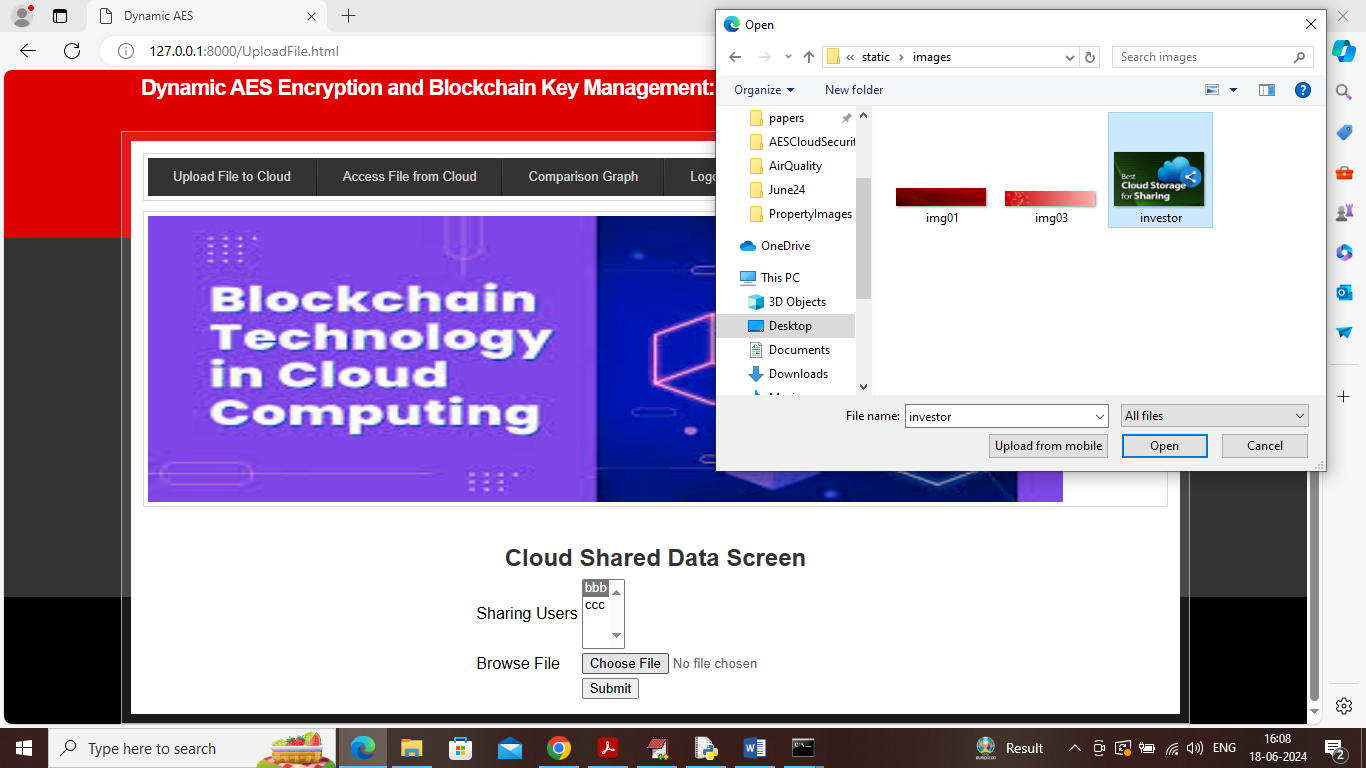
In above screen user can click on ‘Upload File to Cloud’ link to upload file to cloud using ECC, dynamic AES keys etc.



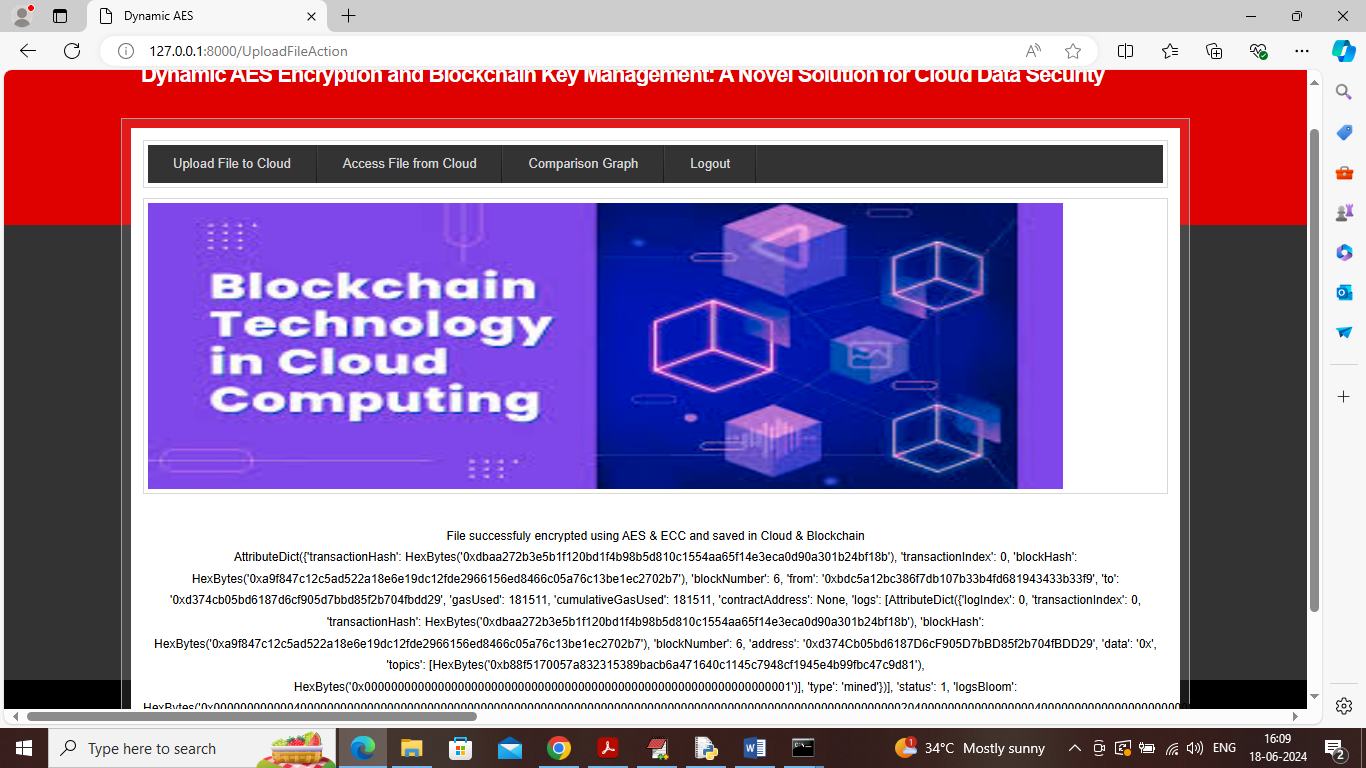
In above screen in first drop down box I am showing list of all users and then by holding CTRL keys you can select any number of users for sharing and then upload any type of file and in above screen I am uploading WORD file and then press button to save file and get below page



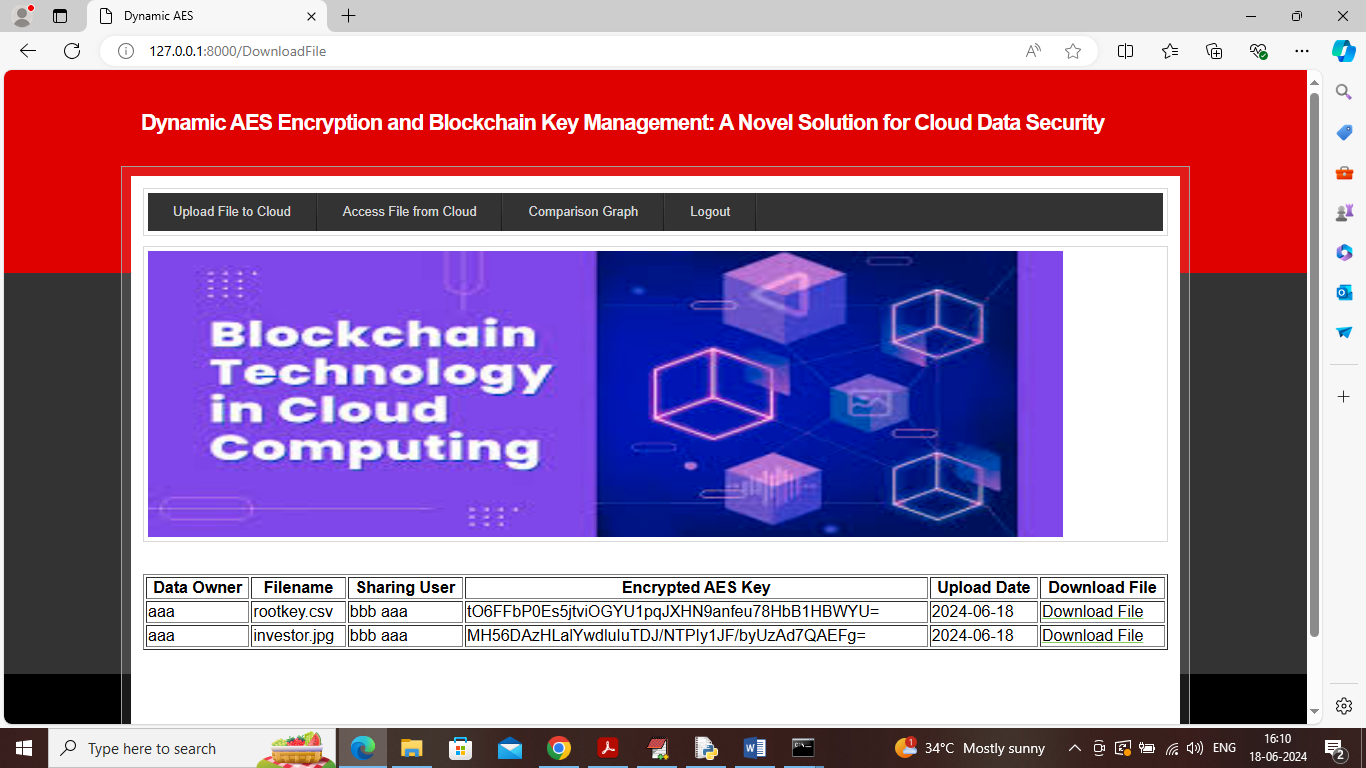
In above screen can see file saved in Cloud and keys saved in Blockchain and then can see other Blockchain output details and now I am uploading image file



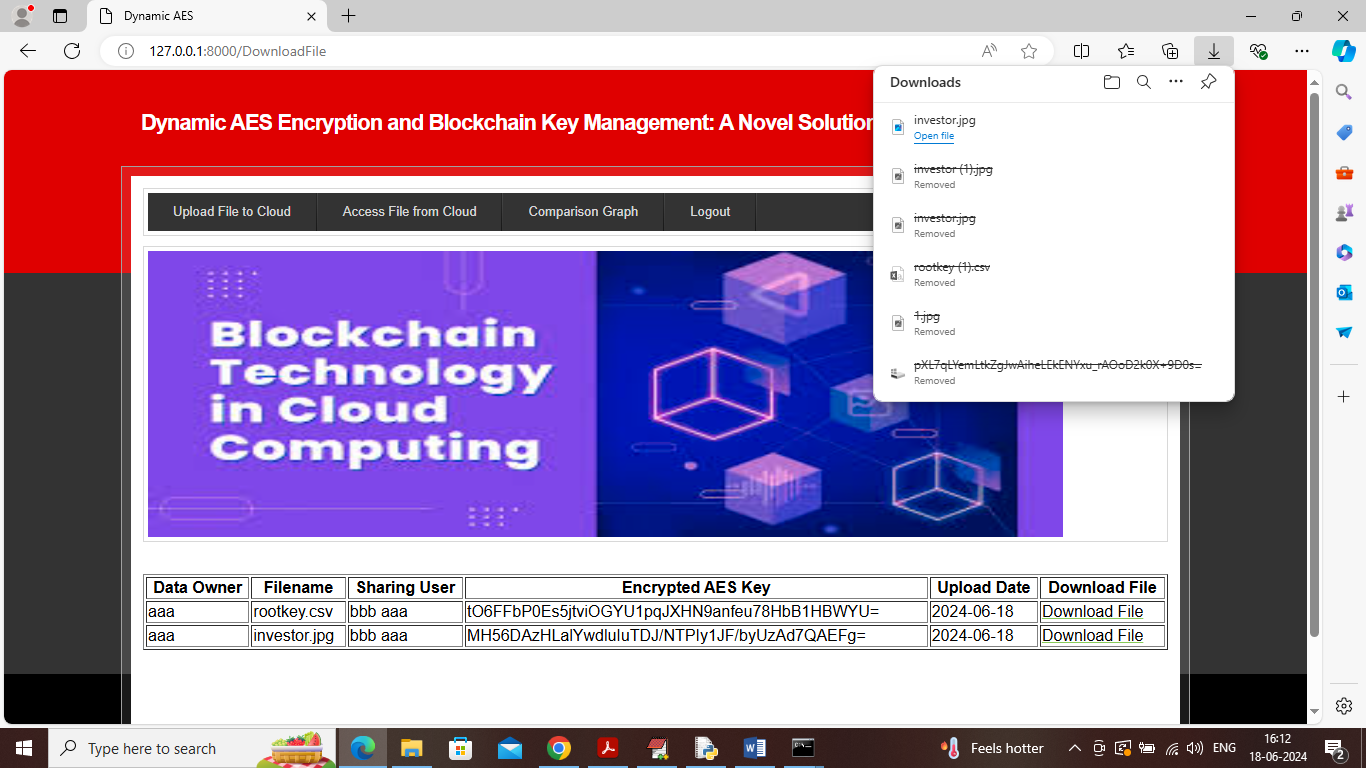
In above screen uploading image file and then click on buttons to get below output



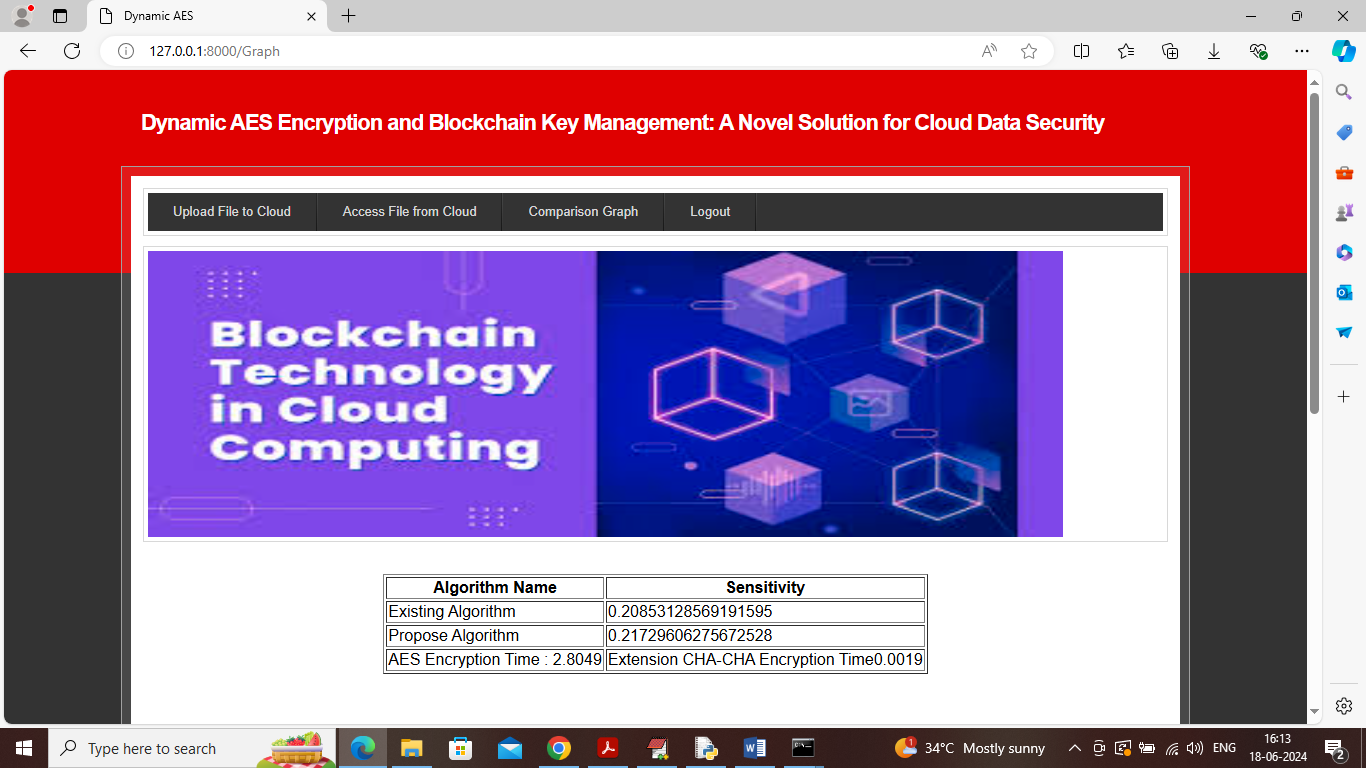
In above screen can see image file also saved in Blockchain and now click on “Access File from Cloud’ to view and download desired file



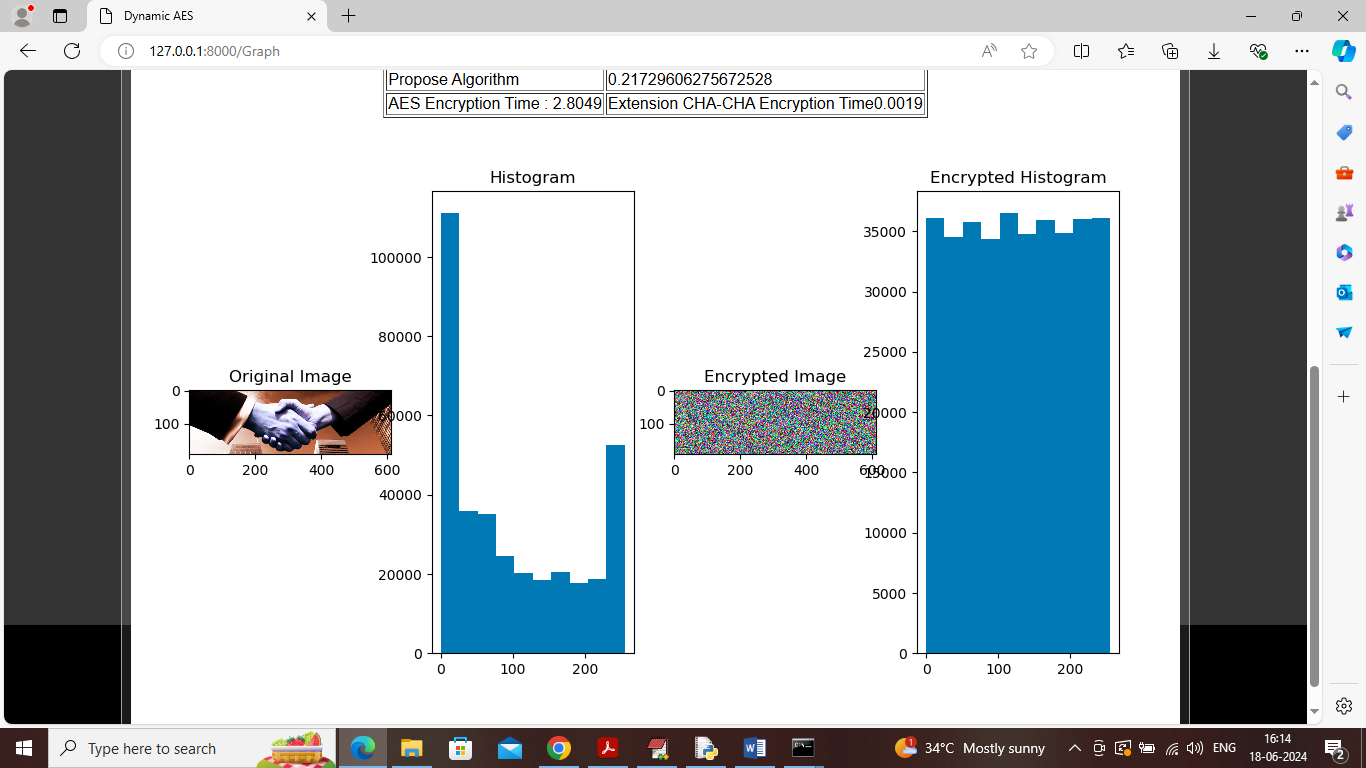
In above screen user can see all file details along with sharing user names and AES dynamic key and can click on ‘Download File’ link to download file in decrypted format



In above screen can see file is downloaded and similarly you can download any file and now click on ‘Comparison Graph’ link to get below page



In above screen can see encryption sensitivity of both existing and propose algorithm and can see propose algorithm has high sensitivity and then in last rows can see AES took 2.80 seconds for key generation and encryption and CHACHA20 extension took only 0.0019 seconds so extension algorithm is much lighter. Scroll down above output screen to get below page



In above screen first image is the original image and then histogram of original image has very much un-uniform values as bars have too much up and down and then can encrypted image and then can see histogram of encrypted image which has uniform values as all bars are equal shape with minor gap.

So by using propose paper concept we can provide decentralized security to cloud data by saving keys at multiple Blockchain nodes