CODE

```
import mysql.connector

from flask_cors import CORS

from flask import *

app = Flask(__name__)

cors = CORS(app)

from web3 import Web3

from solcx import compile_standard, install_solc

from Crypto.Cipher import Blowfish

app.config['CORS_HEADERS'] = 'Content-Type'

from Crypto.Random import get_random_bytes

import requests


# IPFS server API endpoint

ipfs_api_url = "http://127.0.0.1:5001/api/v0"  # Replace with the actual API
URL of your IPFS server


def encrypt_file(input_file, output_file, key):

    cipher = Blowfish.new(key, Blowfish.MODE_ECB)

    chunk_size = 64  # Blowfish block size

    with open("static/upload/"+input_file, 'rb') as infile,
open("static/encrypt/"+output_file, 'wb') as outfile:
        while True:
            chunk = infile.read(chunk_size)
            if len(chunk) == 0:
                break
            elif len(chunk) % 8 != 0:
                # Padding the last block if its size is not a multiple of 8 bytes
                chunk += b' ' * (8 - (len(chunk) % 8))
            encrypted_chunk = cipher.encrypt(chunk)
            outfile.write(encrypted_chunk)
    return upload_file_to_ipfs(output_file)
def decrypt_file(input_file, output_file, key):
    cipher = Blowfish.new(key, Blowfish.MODE_ECB)
    chunk_size = 64  # Blowfish block size
    with open("static/download/"+input_file, 'rb') as infile,
open("static/decrypt"+output_file, 'wb') as outfile:
        while True:
            chunk = infile.read(chunk_size)
            if len(chunk) == 0:
                break
            decrypted_chunk = cipher.decrypt(chunk)
            outfile.write(decrypted_chunk)
```

```
def upload_file_to_ipfs(file_path):
    try:
        # Send a POST request to add the file to IPFS
        response = requests.post(f"{ipfs_api_url}/add", files={"file":
open("static/encrypt/"+file_path, "rb")})
        if response.status_code == 200:
            json_response = response.json()
            print(json_response)
            # The file has been successfully uploaded to IPFS
            ipfs_hash = json_response["Hash"]
            print(ipfs_hash)
            return ipfs_hash
        else:
            print(f"Failed to upload file to IPFS. Status code:
{response.status_code}")
            return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None
def download_file(f,fileid,key):
     # The URL of the file you want to download
    url = "http://127.0.0.1:8080/ipfs/%s?filename=%s"%(fileid,fileid)  #
Replace with the actual API URL of your IPFS server
    # The local file path where you want to save the downloaded file


    try:
        response = requests.get(url)
        if response.status_code == 200:
            with open("static/download"+f, "wb") as file:
                file.write(response.content)
            print(f"File downloaded and saved to {f}")
            decrypt_file(f,f,key)


        else:
            print(f"Failed to download the file. Status code:
{response.status_code}")
    except Exception as e:
        print(f"An error occurred: {e}")
def soliditycontract(e):
  import json
  install_solc("0.6.0")
  with open("./SimpleStorage.sol", "r") as file:
    simple_storage_file = file.read()
    compiled_sol = compile_standard(
        {
            "language": "Solidity",
            "sources": {"SimpleStorage.sol": {"content": simple_storage_file}},
            "settings": {
                "outputSelection": {
                    "*": {
                        "*": ["abi", "metadata", "evm.bytecode",
"evm.bytecode.sourceMap"]
                    }
                }
            },
        },
        solc_version="0.6.0",
    )

    with open("compiled_code.json", "w") as file:
        json.dump(compiled_sol, file)

    bytecode =
compiled_sol["contracts"]["SimpleStorage.sol"]["SimpleStorage"]["evm"][
        "bytecode"
    ]["object"]
    # get abi
    abi = json.loads(
        compiled_sol["contracts"]["SimpleStorage.sol"]["SimpleStorage"]["metad
ata"]
    )["output"]["abi"]




    w3 = Web3(Web3.HTTPProvider('HTTP://127.0.0.1:7545'))
    chain_id = 1337
    print(w3.is_connected())
    my_address = e[0]
    private_key =e[1]
    # initialize contract
    SimpleStorage = w3.eth.contract(abi=abi, bytecode=bytecode)
    nonce = w3.eth.get_transaction_count(my_address)
    # set up transaction from constructor which executes when firstly
    transaction = SimpleStorage.constructor().build_transaction(
        {"chainId": chain_id, "from": my_address, "nonce": nonce}
    )
    signed_tx = w3.eth.account.sign_transaction(transaction,
private_key=private_key)
    tx_hash = w3.eth.send_raw_transaction(signed_tx.rawTransaction)
    tx_receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
    tx_receipt = "".join(["{:02X}".format(b) for b in
tx_receipt["transactionHash"]])
    return tx_receipt



@app.route('/forenics/updatedata', methods=["POST"], strict_slashes=False)
def updatedata():
    r=request.json
    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
    d="update data set filename ='%s',codeid ='%s',keyvalue ='%s',caseid ='%s'
where did='%s'"%(r['filename'],r['codeid'],r['keyvalue'],r['caseid'],r['did'])
    mycursor = mydb.cursor()
    mycursor.execute(d)
    mydb.commit()
    mydb.close()
    return 's'


@app.route('/forenics/viewdata', methods=["POST"], strict_slashes=False)
```

```
def viewdata():
        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
        mycursor = mydb.cursor()
        tx="select *   from data"
        mycursor.execute(tx)
        e=mycursor.fetchall()
        mydb.close()
        return json.dumps(e)
@app.route('/forenics/deletedata', methods=["POST"], strict_slashes=False)
def deletedata():
        r=request.json
        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
        mycursor = mydb.cursor()
        tx="delete from data where did={0}".format(r['id'])
        mycursor.execute(tx)
        mydb.commit()
        mydb.close()
@app.route('/forenics/inserttransactiondata', methods=["POST"],
strict_slashes=False)
def inserttransactiondata():
    r=request.json
    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
    mycursor = mydb.cursor()
    tx = 'select td from transactiondata order by td desc limit 1'
    mycursor.execute(tx)
    e = mycursor.fetchall()
    if len(e) == 0:
            eid = 1
    else:
            eid = e[0][0]+1
    d="insert into
transactiondata(td,trandata,uid,did,transcation,alltrans,trandate)values
('%s','%s','%s','%s','%s','%s','%s')"%(eid,r['trandata'],r['uid'],r['did'],r['transcation
'],r['alltrans'],r['trandate'])
    mycursor = mydb.cursor()
    mycursor.execute(d)
    mydb.commit()
    mydb.close()
    return 'e'


@app.route('/forenics/updatetransactiondata', methods=["POST"],
strict_slashes=False)
```

```
def updatetransactiondata():
    r=request.json
    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
    d="update transactiondata set trandata ='%s',uid ='%s',did ='%s',transcation
='%s',alltrans ='%s',trandate ='%s' where
td='%s'"%(r['trandata'],r['uid'],r['did'],r['transcation'],r['alltrans'],r['trandate'],r['td
'])
    mycursor = mydb.cursor()
    mycursor.execute(d)
    mydb.commit()
    mydb.close()
    return 's'


@app.route('/forenics/viewtransactiondata', methods=["POST"],
strict_slashes=False)
def viewtransactiondata():
        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
        mycursor = mydb.cursor()
        tx="select *   from transactiondata"
        mycursor.execute(tx)
        e=mycursor.fetchall()
        mydb.close()

        return json.dumps(e)

@app.route('/forenics/deletetransactiondata', methods=["POST"],
strict_slashes=False)

def deletetransactiondata():

        r=request.json

        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")

        mycursor = mydb.cursor()

        tx="delete from transactiondata where td={0}".format(r['id'])

        mycursor.execute(tx)

        mydb.commit()

        mydb.close()

        return 's'

@app.route('/forenics/insertusers', methods=["POST"], strict_slashes=False)

def insertusers():

    r=request.json

    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")

    mycursor = mydb.cursor()

    tx = 'select uid from users order by uid desc limit 1'

    mycursor.execute(tx)
    e = mycursor.fetchall()
    if len(e) == 0:
            eid = 1
    else:
            eid = e[0][0]+1
    d="insert into users(uid,name,email,password,addresss,keydata)values
('%s','%s','%s','%s','%s','%s')"%(eid,r['name'],r['email'],r['password'],r['addresss']
,r['keydata'])
    mycursor = mydb.cursor()
    mycursor.execute(d)
    mydb.commit()
    mydb.close()
    return 'e'


@app.route('/forenics/updateusers', methods=["POST"], strict_slashes=False)
def updateusers():
    r=request.json
    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
    d="update users set name ='%s',email ='%s',password ='%s',addresss
='%s',keydata ='%s' where
uid='%s'"%(r['name'],r['email'],r['password'],r['addresss'],r['keydata'],r['uid'])
    mycursor = mydb.cursor()
    mycursor.execute(d)

    mydb.commit()

    mydb.close()

    return 's'


@app.route('/forenics/viewusers', methods=["POST"], strict_slashes=False)

def viewusers():

        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")

        mycursor = mydb.cursor()

        tx="select *   from users"

        mycursor.execute(tx)

        e=mycursor.fetchall()

        mydb.close()

        return json.dumps(e)

@app.route('/forenics/deleteusers', methods=["POST"], strict_slashes=False)

def deleteusers():

        r=request.json

        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")

        mycursor = mydb.cursor()

        tx="delete from users where uid={0}".format(r['id'])
        mycursor.execute(tx)

        mydb.commit()

        mydb.close()

        return 's'


@app.route('/forenics/upload', methods = ['POST'])

def success():

    if request.method == 'POST':

        f = request.files['file']

        caseid=request.form["caseid"]

        key = get_random_bytes(8)

        address=request.form["address"]

        private=request.form["private"]


        uid=request.form["uid"]

        ha=soliditycontract([address,private])


        f.save("static/upload/"+f.filename)

        received=encrypt_file(f.filename,"en"+f.filename, key)

        print(f,caseid,key,"en"+f.filename,received)

        mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
```

```
    mycursor = mydb.cursor()

    tx = 'select did from data order by did desc limit 1'

    mycursor.execute(tx)

    e = mycursor.fetchall()

    if len(e) == 0:

        eid = 1

    else:

        eid = e[0][0]+1

    d="""insert into data(did,filename,codeid,keyvalue,caseid)values
("%s","%s","%s","%s","%s")"""%(eid,f.filename,received,key,caseid)

    mycursor = mydb.cursor()

    mycursor.execute(d)

    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")

    mycursor = mydb.cursor()

    tx = 'select td from transactiondata order by td desc limit 1'

    mycursor.execute(tx)

    e = mycursor.fetchall()

    if len(e) == 0:

        did = 1

    else:

        did = e[0][0]+1
    d="insert into transactiondata(td,trandata,uid,did,alltrans)values
('%s','%s','%s','%s','%s')"%(did,ha,uid,eid,'insert')
    mycursor = mydb.cursor()
    mycursor.execute(d)
    mydb.commit()
    mydb.close()
    return 'e'



@app.route('/forenics/login', methods=["POST"], strict_slashes=False)
def login():
    r=request.json
    mydb = mysql.connector.connect(host="localhost",
user="root",  password="",  database="forenics")
    mycursor = mydb.cursor()
    tx="select *   from users where uid='%s' and
password='%s'"%(r["id"],r["password"])
    mycursor.execute(tx)
    e=mycursor.fetchone()
    mydb.close()
    return json.dumps(e)
```

```
if __name__ == '__main__':
    app.run(debug=True)
import axios from "axios";
import { useState } from "react";
const Adddata = () => {
  const [caseid, setcaseid] = useState("");
  const data = JSON.parse(window.localStorage.getItem("data"));

  const [file, setFile] = useState("");
  function handleChange(event) {
    setFile(event.target.files[0]);
  }
  function handleSubmit(event) {
    event.preventDefault();
    const url = "http://localhost:5000/forenics/upload";
    const formData = new FormData();
    formData.append("file", file);
    formData.append("fileName", file.name);
    formData.append("caseid", caseid);
    formData.append("address", data[4]);
    formData.append("private", data[5]);
    formData.append("uid", data[0]);
    const config = {
      headers: {
        "content-type": "multipart/form-data",
      },
    };
    axios.post(url, formData, config).then((response) => {
      console.log(response.data);
      alert("uploaded");
      setcaseid("");
      setFile("");
    });
  }

  return (
    <div>
      <h1>Upload data</h1>
      <div className="form-floating mb-3 mt-3">
        <input
          type="text"
          className="form-control"
          onChange={(e) => setcaseid(e.target.value)}
          value={caseid}
```

```
          placeholder="Enter caseid"
        />
        <label htmlFor="caseid">caseid</label>
      </div>
      <div className="form-floating mb-3 mt-3">
        <form onSubmit={handleSubmit}>
          <input type="file" onChange={handleChange} className="form-
control" />

          <button type="submit" className="btn btn-primary">
            Upload
          </button>
        </form>
      </div>

      {/* <div className="form-floating mb-3 mt-3">
          <input
            type="text"
            className="form-control"
            onChange={(e) => setcodeid(e.target.value)}
            value={codeid}
            placeholder="Enter codeid"
          />
          <label htmlFor="codeid">codeid</label>
        </div>

        <div className="form-floating mb-3 mt-3">
          <input
            type="text"
            className="form-control"
            onChange={(e) => setkeyvalue(e.target.value)}
            value={keyvalue}
            placeholder="Enter keyvalue"
          />
          <label htmlFor="keyvalue">keyvalue</label>
        </div> */}
    </div>
  );
};
export default Adddata;

import axios from "axios";
import { useState } from "react";
const Addtransactiondata = () => {
```

```
const [trandata, settrandata] = useState("");
const [uid, setuid] = useState("");
const [did, setdid] = useState("");
const [transcation, settranscation] = useState("");
const [alltrans, setalltrans] = useState("");
const [trandate, settrandate] = useState("");
const submitdata = () => {
  const value = {
    trandata: trandata,
    uid: uid,
    did: did,
    transcation: transcation,
    alltrans: alltrans,
    trandate: trandate,
  };
  axios.post("http://localhost:5000/forenics/inserttransactiondata", value);
  alert("success");
  settrandata("");
  setuid("");
  setdid("");
  settranscation("");
  setalltrans("");
  settrandate("");
};
return (
  <div>
    <h1>Add transactiondata</h1>
    <div className="form-floating mb-3 mt-3">
      <input
        type="text"
        className="form-control"
        onChange={(e) => settrandata(e.target.value)}
        value={trandata}
        placeholder="Enter trandata"
      />
      <label htmlFor="trandata">trandata</label>
    </div>

    <div className="form-floating mb-3 mt-3">
      <input
        type="text"
        className="form-control"
        onChange={(e) => setuid(e.target.value)}
        value={uid}
    placeholder="Enter uid"
  />
  <label htmlFor="uid">uid</label>
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="text"
    className="form-control"
    onChange={(e) => setdid(e.target.value)}
    value={did}
    placeholder="Enter did"
  />
  <label htmlFor="did">did</label>
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="text"
    className="form-control"
    onChange={(e) => settranscation(e.target.value)}
    value={transcation}
    placeholder="Enter transcation"
  />
  <label htmlFor="transcation">transcation</label>
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="text"
    className="form-control"
    onChange={(e) => setalltrans(e.target.value)}
    value={alltrans}
    placeholder="Enter alltrans"
  />
  <label htmlFor="alltrans">alltrans</label>
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="text"
    className="form-control"
    onChange={(e) => settrandate(e.target.value)}
    value={trandate}
```

```
          placeholder="Enter trandate"
        />
        <label htmlFor="trandate">trandate</label>
      </div>

      <input
        type="submit"
        className="btn btn-primary"
        onClick={submitdata}
        style={{ width: "100%" }}
      />
    </div>
  );
};
export default Addtransactiondata;

import axios from "axios";
import { useState } from "react";
const Addusers = () => {
  const [name, setname] = useState("");
  const [email, setemail] = useState("");
  const [password, setpassword] = useState("");
```

```
};

const [addresss, setaddresss] = useState("");

const [keydata, setkeydata] = useState("");

const submitdata = () => {

  const value = {

    name: name,

    email: email,

    password: password,

    addresss: addresss,

    keydata: keydata,

  };

  axios

    .post("http://localhost:5000/forenics/insertusers", value)

    .then((res) => {

      alert("success");

      setname("");

      setemail("");

      setpassword("");

      setaddresss("");

      setkeydata("");

    });

};

return (
<div>
 <h1>Register users</h1>
 <div className="form-floating mb-3 mt-3">
  <input
   type="text"
   className="form-control"
   onChange={(e) => setname(e.target.value)}
   value={name}
   placeholder="Enter name"
  />
  <label htmlFor="name">name</label>
 </div>

 <div className="form-floating mb-3 mt-3">
  <input
   type="text"
   className="form-control"
   onChange={(e) => setemail(e.target.value)}
   value={email}
   placeholder="Enter email"
  />
  <label htmlFor="email">email</label>
```

```
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="password"
    className="form-control"
    onChange={(e) => setpassword(e.target.value)}
    value={password}
    placeholder="Enter password"
  />
  <label htmlFor="password">password</label>
</div>

<div className="form-floating mb-3 mt-3">
  <input
    type="text"
    className="form-control"
    onChange={(e) => setaddresss(e.target.value)}
    value={addresss}
    placeholder="Enter addresss"
  />
  <label htmlFor="addresss">addresss</label>
```

```
      </div>


      <div className="form-floating mb-3 mt-3">
        <input
          type="text"
          className="form-control"
          onChange={(e) => setkeydata(e.target.value)}
          value={keydata}
          placeholder="Enter keydata"
        />
        <label htmlFor="keydata">keydata</label>
      </div>


      <input
        type="submit"
        className="btn btn-primary"
        onClick={submitdata}
        style={{ width: "100%" }}
      />
    </div>
  );
};
```

```
export default Addusers;


import Updatedata from "./updatedata";
import Viewdata from "./viewdata";
import Adddata from "./adddata";
import Updatetransactiondata from "./updatetransactiondata";
import Viewtransactiondata from "./viewtransactiondata";
import Addtransactiondata from "./addtransactiondata";
import Updateusers from "./updateusers";
import Viewusers from "./viewusers";
import Addusers from "./addusers";
import Login from "./Login";
import { Route, Routes } from "react-router-dom";


const App = () => {
    return ( <>
        <Routes>
        <Route path="/updatedata" element={<Updatedata/>} />
        <Route path="/viewdata" element={<Viewdata/>} />
        <Route path="/adddata" element={<Adddata/>} />
        <Route path="/updatetransactiondata"
element={<Updatetransactiondata/>} />
```

```
        <Route path="/viewtransactiondata" element={<Viewtransactiondata/>} />

        <Route path="/addtransactiondata" element={<Addtransactiondata/>} />

        <Route path="/updateusers" element={<Updateusers/>} />

        <Route path="/viewusers" element={<Viewusers/>} />

        <Route path="/addusers" element={<Addusers/>} />

        <Route path="/" element={<Login/>} />

        </Routes>

    </> );

}


export default App;
```