Assignment 5 thi: Secure Data Encryption System

**streamlit:**

Hum Streamlit naam ka ek tool use kar rahe hain. Ye tool humein Python se app banane deta hai (jese buttons, text box, etc.)

**hashlib:**

hashlib ek built-in (Python ka apna) tool hai Jo password ko aise badal deta hai ke koi doosra usay samajh na sake (ye hota hai "hashing").

**json:**

Data ko .json file mein save karne ke liye. json ek tool hai jo data ko save aur read karne mein madad karta hai jese name/password ko file mein likhna.

**os:**

File check karne ke liye (file exist karti hai ya nahi). os ka matlab hota hai Operating System. Is se hum check karte hain: koi file hai ya nahi? ya file banana hai?

**time:**

Login fail hone par time delay dene ke liye. time se hum samay ka pata laga sakte hain — jese 60 second tak wait karwana, ,

**Fernet:**

ek encryption tool hai — ye tumhara data ko **lock aur unlock** karta hai. Ye totally **secret key** se kaam karta hai, bina key ke data nahi khulta.

**urlsafe\_b64encode:** Key ko safe banata hai. Ye function chabi (key) ko **safe format** mein convert karta hai, taake encryption mein use ho sake. Ye password ko super-strong format mein change karta hai, na copy ho sakta hai na guess.  
"Ye password ko ek iron box mein seal karta hai — koi nahi dekh sakta." 🧰🛠️

------------------------------------------------------------------------------------------------------

|  |
| --- |
| Data\_file = "Secure\_Data.json" |

Ye ek **file ka naam** hai jisme hum sari users ki info save karenge. (is ka name hm kuch bhi rkh skty hen )

|  |
| --- |
| SALT = b"secure\_salt\_value" |

SALT ek extra chhoti si cheez hoti hai jo password ke sath add hoti hai, taake secure ban jaye.

|  |
| --- |
| LOCKOUT\_DURATION = 60 |

Yeh batata hai ke agar user galat password 3 baar daale…  
Toh usko **60 second tak band** kar diya jaye — yani wait karna padega 😴

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |
| --- |
| if "authenticated\_user" not in st.session\_state:  st.session\_state.authenticated\_user = None |

Yeh check karta hai: kya koi banda login hua hua hai?  
Agar nahi hua toh default None yani khaali set kar do.

👩‍🏫 Samajh:  
"Yani agar koi mehmaan ghar nahi aaya, toh mehmaan ki kursi khaali rakh do." 🪑

📌 session\_state ek magic box hota hai jisme tum temporary data rakh sakti ho app ke andar.

|  |
| --- |
| if "failed\_attempts" not in st.session\_state:  st.session\_state.failed\_attempts = 0 |

Yeh count karta hai ke user ne kitni baar galat password likha.  
Agar pehli baar likha, toh count 0 se shuru hoga.

👩‍🏫 Samajh:  
"Jaise teacher galtiyon ki ginti karti hain: 0 galti, 1 galti, 2 galti..

|  |
| --- |
| if "lockout\_time" not in st.session\_state:  st.session\_state.lockout\_time = 0 |

Yeh batata hai: user ko kab tak wait karna hai agar usne 3 galtiyan kar di.

⏰ Samajh:  
"Yeh alarm clock hai jo kehti hai: 1 minute wait karo, fir try karo." ⏰

|  |
| --- |
| def load\_data():  if os.path.exists(Data\_file):  with open(Data\_file , "r") as f:  return json.load(f)  return {} |

Yeh ek function hai — ek box jise tum bula sakti ho jab data chahiye ho file se.

📂 Samajh:  
"Jaise tum apni diary (JSON file) kholo aur dekho ke andar kya likha hai." 📖

💡 json.load(f) ka matlab hai:  
“File ke andar se pura data Python mein le aao.”

**Part 3: "Secret Password Ko Chhupana (Hashing)"**

🧒 Story:

Socho Mona ne ek magic password likha hai: "choco123"  
Lekin Mona chaahti hai koi doosra usse na padh le. To Mona us password ko aik machine mein daalti hai — Hash Machine!

Yeh machine us password ko tod modi kar ke uska aik secret code (hash) bana deti hai.  
Aur Mona ne bola: “Ab main original password nahi save karungi, sirf hash save karungi!”

✅ Iska fayda:

Agar koi computer chura le bhi le, to usay sirf hash milega — wo kabhi asli password nahi dekh sakega!

import hashlib

SALT = b"secure\_salt\_value" #Secret **namak (salt)** — ye password ko aur zyada strong banata hai. # Mona ka special secret namak!

def hash\_password(password): #Ek function bna rahe hain jo password ko hash karega.

return hashlib.pbkdf2\_hmac('sha256', password.encode(), SALT, 100000).hex()

|  |
| --- |
| hex() |

|  |
| --- |
| Hash ko readable text mein badal raha hai. |
| hash\_password("choco123")  'fa9e42c1b672c91e44f18b23ae8cf4577ff676d63bdb3a2eeabf33...' |

## 🧩 Part 4: “Secret Lock (Key) Banana” 🔐

Function: generate\_key(passkey)

### 🧒 Story Style:

Socho Mona ne aik **secret passkey** banai: "icecream786"  
Ab Mona ke paas ek **treasure box** hai (encrypted data) — us box ko sirf ek **key (chaabi)** se khola ja sakta hai.

Lekin Mona ko khud ye key banana nahi aata — is liye wo aik **Key Machine** chalati hai, jo passkey se proper chaabi banata hai.

Ye machine ka naam hai:

from base64 import urlsafe\_b64encode

from hashlib import pbkdf2\_hmac

SALT = b"secure\_salt\_value"

def generate\_key(passkey): # Ye pura process ek function mein rakha gaya hai. Passkey do, key lo. 🗝️

key = pbkdf2\_hmac('sha256', passkey.encode(), SALT, 100000) # Ye **passkey ko hash karke aik raw key** banata hai — bilkul strong aur unique!

return urlsafe\_b64encode(key) # Ye raw key ko readable format mein convert karta hai — taki encryption system samajh sake.

"I like chocolate ice cream"

Lekin Mona ko dar hai ke koi aur na padh le — is liye wo message ko lock kar deti hai ek secret passkey ke zariye, jaise "icecream786"

Phir baad mein Mona apni hi passkey se us message ko unlock (decrypt) kar leti hai.

from cryptography.fernet import Fernet (Ye Python ka tool hai jo encryption/decryption karta hai.)

def encrypt\_data(text, KEY):

cipher = Fernet(generate\_key(KEY)) (Ye original message ko encrypt karta hai (chhupata hai).)

return cipher.encrypt(text.encode()).decode()

|  |
| --- |
| msg = "I like chocolate ice cream"  key = "icecream786"  encrypted = encrypt\_data(msg, key)  print("Encrypted:", encrypted)  Encrypted: gAAAAABlYvm... (long hidden string) |

|  |
| --- |
| def decrypt\_data(encrypted\_text, KEY):  try:  cipher = Fernet(generate\_key(KEY))  return cipher.decrypt(encrypted\_text.encode()).decode()  except:  return None |

|  |
| --- |
| original = decrypt\_data(encrypted, key)  print("Decrypted:", original)  Decrypted: I like chocolate ice cream |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Kaam** | | --- | | Encrypt | | Decrypt | | | **Function** | | --- | | encrypt\_data() | | decrypt\_data() | | | **Result** | | --- | | Secret message ban gaya | | Wapas original message mil gaya | |

## 🧩 Part 6: Register & Login System 👤🔑

Is part mein tum banati ho:

1. Naya user 📋
2. Password save hota hai 🔒
3. User login karta hai ✔️
4. Galat password par lock lagta hai 🚫⏱️

elif choice == "Register":

st.subheader("Register New User")

user\_name = st.text\_input("Choose Username")

password = st.text\_input("Choose Password", type="password")

if st.button("Register"):

if user\_name and password:

if user\_name in stored\_data: (Check karo ke wo user pehle se to nahi)

st.warning("User already exists.")

else:

stored\_data[user\_name] = { #(Naye user ka data save karo)

"password": hash\_password(password), #(Password ko hide karke save karo (like a secret code))

"data": []

}

save\_Data(stored\_data)

st.success("User Register successful")

else:

st.error("Both fields are required.")

 pbkdf2\_hmac = Special built-in function jo password ko **gala kar deta hai** (matlab readable nahi rehta)

 .hex() = Us gala data ko readable characters mein badal deta hai (but still hidden)

## ✅ Example Flow:

1. 👧 Mona likhti hai:  
   **Username:** mona2025  
   **Password:** icecream786
2. ✔️ Mona Register karti hai
3. 🔒 System store karta hai:

|  |
| --- |
| 1. "mona2025": { 2. "password": "ab23c4e....", 3. "data": [] 4. } |

Mona ka **original password save nahi hota**! 😎  
Sirf gala version (hash) save hota hai — secure! ✅

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Line** | | --- | | login\_user | | password | | if ... == hash\_password(password) | | ✅ Sahi ho to | | ❌ Galat ho to | | 🔒 3 galti ke baad | | | **Kya Hota Hai** | | --- | | User ka naam input lo | | Password input lo | | Check karo: password sahi hai ya nahi | | Login successful | | Ginti ghatao (3 tries allowed) | | System 60 second ke liye lock ho jata hai | |

## 🧩 Part 7: Data Save & Retrieve System 💾🔐

Is part mein tum yeh seekhogi:

1. **User kuch likhega** (like: "My secret message") 📝
2. Uska **encrypted version save hoga** (secure form) 🔒
3. Baad mein wo **decrypt karke** usi message ko waapis dekh sakega 👀

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | **Line** | | --- | | raw\_data = st.text\_input(...) | | encrypt(raw\_data) | | append(encrypted) | | save\_Data(...) | | | **Matlab Kya Hai** | | --- | | User koi message likhe | | Us message ko **gala** (encrypt) karo | | Us encrypted message ko list mein save karo | | Usay file mein likh do permanently | |
| | **Line** | | --- | | encrypted\_data\_list = ...["data"] | | for i, enc in ... | | decrypt(enc) | | st.write(...) | | | **Matlab** | | --- | | User ke saved encrypted messages | | Har message ke liye loop | | Message ko waapis normal form mein lao | | Usay screen pe dikhado | |

Mona, tumhara **secure locker system** tayyar hai! 🔐  
User register karta hai, login karta hai, apna message likhta hai, wo secure hota hai, aur baad mein dekh sakta hai — bina kisi aur ke dekhe! 🎉