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	Function	Result
Encrypt	`encrypt_data()`	Secret message ban gaya
Decrypt	`decrypt_data()`	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:

```
"mona2025": {
  "password": "ab23c4e....",
  "data": []
}
```

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

Line	Kya Hota Hai
`login_user`	User ka naam input lo
`password`	Password input lo
`if ... == hash_password(password)`	Check karo: password sahi hai ya nahi
✅ Sahi ho to	Login successful
❌ Galat ho to	Ginti ghatao (3 tries allowed)
🔒 3 galti ke baad	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	**Matlab Kya Hai**
`raw_data = st.text_input(...)`	User koi message likhe
`encrypt(raw_data)`	Us message ko **gala** (encrypt) karo
`append(encrypted)`	Us encrypted message ko list mein save karo
`save_Data(...)`	Usay file mein likh do permanently
Line	**Matlab**
`encrypted_data_list = ...["data"]`	User ke saved encrypted messages
`for i, enc in ...`	Har message ke liye loop
`decrypt(enc)`	Message ko waapis normal form mein lao
`st.write(...)`	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! 🎉