from cryptography.fernet import Fernet

import json

import os

import tkinter as tk

from tkinter import messagebox

import hashlib

class PasswordManager:

def \_\_init\_\_(self):

self.key = None

self.file = 'passwords.json'

self.passwords = {}

self.master\_password = None

def authenticate(self, entered\_password):

hashed\_password = hashlib.sha256(entered\_password.encode()).hexdigest()

return hashed\_password == self.master\_password

def load\_key(self):

key\_file = 'key.key'

if os.path.exists(key\_file):

with open(key\_file, 'rb') as file:

self.key = file.read()

else:

self.key = Fernet.generate\_key()

with open(key\_file, 'wb') as file:

file.write(self.key)

def load\_passwords(self):

try:

with open(self.file, 'rb') as file:

data = file.read()

if data:

decrypted\_data = Fernet(self.key).decrypt(data)

self.passwords = json.loads(decrypted\_data)

except FileNotFoundError:

pass

def save\_passwords(self):

with open(self.file, 'wb') as file:

encrypted\_data = Fernet(self.key).encrypt(json.dumps(self.passwords).encode())

file.write(encrypted\_data)

def add\_password(self, service, username, password):

hashed\_password = hashlib.sha256(password.encode()).hexdigest()

self.passwords[service] = {'username': username, 'password': hashed\_password}

self.save\_passwords()

messagebox.showinfo("Success", f"Password for '{service}' added successfully.")

def get\_password(self, service):

if service in self.passwords:

return self.passwords[service]

else:

messagebox.showerror("Error", f"Password for '{service}' not found.")

def list\_services(self):

service\_list = "\nStored Services:\n"

for service in self.passwords:

service\_list += service + "\n"

messagebox.showinfo("Stored Services", service\_list)

def display\_passwords(self):

password\_list = "\nStored Passwords:\n"

for service, info in self.passwords.items():

password\_list += f"Service: {service}\nUsername: {info['username']}\nPassword: {info['password']}\n\n"

messagebox.showinfo("Stored Passwords", password\_list)

def set\_master\_password():

global manager

master\_password = master\_password\_entry.get()

if master\_password:

hashed\_password = hashlib.sha256(master\_password.encode()).hexdigest()

manager.master\_password = hashed\_password

manager.load\_key()

manager.load\_passwords()

messagebox.showinfo("Success", "Master Password Set Successfully.")

root.destroy()

else:

messagebox.showerror("Error", "Please enter a valid master password.")

def add\_password():

service = service\_entry.get()

username = username\_entry.get()

password = password\_entry.get()

if service and username and password:

manager.add\_password(service, username, password)

service\_entry.delete(0, tk.END)

username\_entry.delete(0, tk.END)

password\_entry.delete(0, tk.END)

else:

messagebox.showerror("Error", "Please fill in all fields.")

def retrieve\_password():

service = service\_entry.get()

password\_info = manager.get\_password(service)

if password\_info:

messagebox.showinfo("Password Info", f"Username: {password\_info['username']}\nPassword: {password\_info['password']}")

else:

messagebox.showerror("Error", f"Password for '{service}' not found.")

def list\_services():

manager.list\_services()

def display\_passwords():

manager.display\_passwords()

def exit\_app():

root.destroy()

# GUI Setup

root = tk.Tk()

root.title("Set Master Password")

master\_password\_label = tk.Label(root, text="Enter Master Password: ")

master\_password\_label.pack()

master\_password\_entry = tk.Entry(root, show='\*')

master\_password\_entry.pack()

set\_password\_button = tk.Button(root, text="Set Password", command=set\_master\_password)

set\_password\_button.pack()

manager = PasswordManager() # Create manager instance before main loop

root.mainloop()

# Validation for master password set

if not manager.master\_password:

messagebox.showerror("Authentication Failed", "Master password not set. Exiting.")

exit()

# GUI for password manager

root = tk.Tk()

root.title("Password Manager")

# Labels and Entries

tk.Label(root, text="Service: ").grid(row=0, column=0)

tk.Label(root, text="Username: ").grid(row=1, column=0)

tk.Label(root, text="Password: ").grid(row=2, column=0)

service\_entry = tk.Entry(root)

username\_entry = tk.Entry(root)

password\_entry = tk.Entry(root, show='\*')

service\_entry.grid(row=0, column=1)

username\_entry.grid(row=1, column=1)

password\_entry.grid(row=2, column=1)

# Buttons

add\_button = tk.Button(root, text="Add Password", command=add\_password)

add\_button.grid(row=3, column=0, columnspan=2, pady=10)

retrieve\_button = tk.Button(root, text="Retrieve Password", command=retrieve\_password)

retrieve\_button.grid(row=4, column=0, columnspan=2, pady=10)

list\_button = tk.Button(root, text="List Services", command=list\_services)

list\_button.grid(row=5, column=0, columnspan=2, pady=10)

display\_button = tk.Button(root, text="All Accounts", command=display\_passwords)

display\_button.grid(row=6, column=0, columnspan=2, pady=10)

exit\_button = tk.Button(root, text="Exit", command=exit\_app)

exit\_button.grid(row=7, column=0, columnspan=2, pady=10)

root.mainloop()