**PYTHON PROJECT**

**(ASSIGNMENT 2)**

**NAME- MRIDUL SINGH RAWAT**

**SAP ID – 500119881**

**BATCH 32**

**ROLL NO. – R2142231074**

**CONTACT NO. – 7906465904**

**DOMAIN- Security and Privacy management**

**Topic for this project under (Security and Privacy management) – PASSWORD MANAGER**

**(Using objects)**

**CODE SNIPPET**

import random

class Account:

    def \_\_init\_\_(self, name, password):

        self.name = name

        self.password = password

    def is\_strong\_password(self):

        has\_upper = False

        has\_lower = False

        has\_digit = False

        has\_special = False

        for char in self.password:

            if char.isupper():

                has\_upper = True

            elif char.islower():

                has\_lower = True

            elif char.isdigit():

                has\_digit = True

            else:

                has\_special = True

        return has\_upper and has\_lower and has\_digit and has\_special

class PasswordManager:

    def \_\_init\_\_(self):

        self.accounts = {}

    def \_hash\_password(self, password):

        hash\_val = 0

        for char in password:

            hash\_val = (hash\_val \* 31 + ord(char)) % (10\*\*9 + 7)

        return hash\_val

    def add\_account(self, account\_name, password):

        hashed\_password = self.\_hash\_password(password)

        account = Account(account\_name, password)

        self.accounts[account\_name] = account

    def get\_password(self, account\_name):

        account = self.accounts.get(account\_name)

        if account:

            return account.password

        else:

            return "Account not found!"

    def delete\_account(self, account\_name):

        if account\_name in self.accounts:

            del self.accounts[account\_name]

            return "Account deleted successfully!"

        else:

            return "Account not found!"

    def list\_accounts(self):

        return list(self.accounts.keys())

    def suggest\_strong\_password(self, length=12):

        alphabet = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&\*()\_-+=[]{}|;:,.<>?`~"

        password = ''.join(random.choice(alphabet) for \_ in range(length))

        return password

def main():

    password\_manager = PasswordManager()

    while True:

        print("\n1. Add Account")

        print("2. Get Password")

        print("3. Delete Account")

        print("4. List Accounts")

        print("5. Generate Strong Password")

        print("6. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':

            account\_name = input("Enter account name: ")

            password = input("Enter password: ")

            if not Account(account\_name, password).is\_strong\_password():

                print("Warning: Weak password detected! Consider using a stronger password.")

            password\_manager.add\_account(account\_name, password)

            print("Account added successfully!")

        elif choice == '2':

            account\_name = input("Enter account name: ")

            print("Password:", password\_manager.get\_password(account\_name))

        elif choice == '3':

            account\_name = input("Enter account name: ")

            print(password\_manager.delete\_account(account\_name))

        elif choice == '4':

            print("Accounts:", password\_manager.list\_accounts())

        elif choice == '5':

            length = int(input("Enter the length of the password: "))

            print("Suggested Strong Password:", password\_manager.suggest\_strong\_password(length))

        elif choice == '6':

            print("Exiting...")

            break

        else:

            print("Invalid choice! Please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**OUTPUT**

