Practica No.15: Tkinter y SQLite: Insert

Juan Antonio Montoya Ramirez

121040911

S181

Programación Orientada a Objetos

SQLite

Sistema para el control de base de datos relacional, la cual se considera de dominio público, debido a que es de código abierto open source.

Se caracteriza por ser transaccional.

DB Browser

La idea es facilitarnos en la medida de lo posible la creación y administración de bases de datos basadas en SQLite.
Viene integrado dentro de Python

- Crear bases de datos y compactarlas.
- Definir y eliminar tablas, entradas e índices personalizados.
- Dispone de un buscador de entradas con filtros.
- Importar y exportar entradas en modo texto.
- Convertir tablas a ficheros CSV.
- Posibilidad de importar y exportar bases de datos SQL.

Funciones de Controlador

```python
def conexionBD(self):

    try:
        conexion = sqlite3.connect("/Users/juanmontoya/Desktop/UPQ/5 cuatri/POO/Parcial 1/Git/POOS181/3Parcial/tkinterSQLite/DBUs
        print("Conectado a la BD")
        return conexion

    except sqlite3.OperationalError:
        print("No se pudo conectar a la BD")


#metodo para guardar usuarios
def guardarUsuario(self,nom,cor,con):

    #1.-se usa la conexion pasada
    conx=self.conexionBD()

    #2.-se validan los parametors, para que no haya vacios

    if(nom=="" or cor=="" or con==""):
        messagebox.showinfo("Aguas", "Formulario incompleto")

    else:

        #3.-preparamos los datos
        #cursor es el que ejecuta los movimientos a la base de datos
        cursor=conx.cursor()
        conH=self.encriptarCon(con)
        datos=(nom,cor,conH)
        qrInsert="insert into TBRegistrados(Nombre, Correo, Contra) values(?,?,?)"

        #4.-ejecuta Insert y cerramos Conexion
        cursor.execute(qrInsert,datos)
        conx.commit()
        conx.close
        messagebox.showinfo("Bien","Se ha guardado el usuario")


def encriptarCon(self,con):
    conPlana= con
    conPlana= conPlana.encode() #convertimos con a bytes
    sal= bcrypt.gensalt() #regresa caracteres aleatorios en bytes
    conHa= bcrypt.hashpw(conPlana,sal)#a hash, con parametro contraseña en bytes y la sal
    print(conHa)
    return conHa
```

Importación y realización de la ventana

```python
from tkinter import *
from tkinter import ttk
import tkinter as tk

#1.- se realiza la importacion de clase para que se conozcan
from controladorBD import *

#2.-creamos un objeto de la clase Controlador BD

#(nombre del objeto) = (clase importada)
controlador = controladorBD()

#3.-Funcion para el boton
def ejecutaInsert():
    #se ingresan los datos obtenidos desde aqui
    controlador.guardarUsuario(varNom.get(),varCor.get(),varCon.get())


Ventana=Tk()
Ventana.title("CRUD de Usuarios")
Ventana.geometry("500x300")

panel= ttk.Notebook(Ventana)
panel.pack(fill="both", expand="yes")

pestana1= ttk.Frame(panel)
pestana2= ttk.Frame(panel)
pestana3= ttk.Frame(panel)
pestana4= ttk.Frame(panel)

titulo=Label(pestana1, text="Registro de Usuarios",fg="black", font=("Modern",18)).pack()

varNom=tk.StringVar()
etqNom=Label(pestana1,text="Nombre: ").pack()
txtNom=Entry(pestana1,textvariable=varNom).pack()

varCor=tk.StringVar()
etqCor=Label(pestana1,text="Correo: ").pack()
txtCor=Entry(pestana1,textvariable=varCor).pack()

varCon=tk.StringVar()
etqCon=Label(pestana1,text="Contraseña: ").pack()
txtCon=Entry(pestana1,textvariable=varCon).pack()
```


Conclusión:

Se llevó de manera adecuada todo el proceso, la conexión fue algo nuevo para mi, al igual que la encriptación, pero me gustó mucho y ya comprendí de mejor manera la importación.