**DESCRIPCION DEL FUNCIONAMIENTO DE “MAIN.PY”**

El código en main.py es un script Python que maneja la comunicación con una base de datos (base\_datos.db) usando el módulo sqlite3.

El código define dos clases, Comunicación y Ventana, para gestionar datos en la base de datos y crear una ventana con varios widgets, respectivamente. La clase Comunicación se encarga de la comunicación con la base de datos, incluyendo la inserción, actualización y recuperación de datos. La clase Ventana crea una ventana con campos de entrada, botones y una tabla para mostrar y manipular los datos.

La clase Comunicación utiliza el módulo sqlite3 para conectarse a una base de datos SQLite y realizar operaciones SQL. También incluye una clase anidada UserInput que representa la entrada del usuario y proporciona métodos de validación utilizando la librería pydantic. La clase Ventana utiliza la biblioteca Tkinter para crear una ventana GUI y configurar varios widgets como etiquetas, botones y una tabla.

Desglose del código:

El script comienza importando varios módulos, incluyendo Tkinter para crear una GUI, sqlite3 para interactuar con la base de datos, pandas y matplotlib para el análisis y visualización de datos, y re para expresiones regulares.

A continuación, define una clase llamada Comunicación, que representa la comunicación con la base de datos. La clase tiene varios atributos, incluyendo conexión (la conexión a la base de datos), cursor (el cursor para ejecutar consultas SQL), nombre, edad, correo, y teléfono (representando el nombre del usuario, edad, correo electrónico, y número de teléfono, respectivamente).

La clase Comunicación también tiene varios métodos, incluyendo \_\_init\_\_, que inicializa los atributos y establece una conexión con la base de datos. También hay métodos para insertar datos en la base de datos (inserta\_datos), recuperar datos de la base de datos (mostrar\_datos), actualizar datos en la base de datos (actualiza\_datos) y eliminar datos de la base de datos (elimina\_datos).

Además, hay una clase anidada llamada UserInput dentro de la clase Comunicación. Esta clase representa la entrada del usuario y proporciona métodos de validación para cada campo (nombre, edad, correo y teléfono).

Por último, existe otra clase llamada Ventana, que representa una ventana con varios widgets para gestionar los datos de la base de datos. Esta clase hereda de la librería Tkinter e incluye métodos para crear y configurar los widgets.

Análisis de código

Entradas:

La clase 'Comunicación' no toma entradas en su constructor.

La clase 'UserInput' dentro de la clase 'Comunicación' toma cuatro entradas: 'nombre' (str), 'edad' (int), 'correo' (str), y 'teléfono' (int).

El método 'inserta\_datos' de la clase 'Comunicación' toma cuatro entradas: 'nombre' (str), 'edad' (int), 'correo' (str), y 'teléfono' (int).

El método 'actualiza\_datos' de la clase 'Comunicación' toma cinco entradas: 'ID' (int), 'nombre' (str), 'edad' (int), 'correo' (str), y 'teléfono' (int).

La clase 'Ventana' toma una entrada: 'master' (objeto Tk).

Flujo:

La clase 'Comunicación' establece una conexión a una base de datos SQLite en su constructor.

La clase 'UserInput' dentro de la clase 'Comunicación' proporciona un método 'validate\_input' para validar la entrada del usuario para cada campo (nombre, edad, correo, teléfono).

El método 'inserta\_datos' de la clase 'Comunicación' valida la entrada mediante el método 'validate\_input' e inserta los datos en la base de datos si son válidos.

El método 'mostrar\_datos' de la clase 'Comunicación' recupera todos los datos de la base de datos.

El método 'elimina\_datos' de la clase 'Comunicación' borra los datos de la base de datos en función del campo nombre.

El método 'actualiza\_datos' de la clase 'Comunicación' valida la entrada mediante el método 'validate\_input' y actualiza los datos de la base de datos en función del campo ID.

La clase 'Ventana' crea una ventana GUI con varios widgets para gestionar los datos.

El método 'agregar\_datos' de la clase 'Ventana' añade los datos de los campos de entrada a la tabla y a la base de datos.

El método 'actualizar\_tabla' de la clase 'Ventana' actualiza la tabla con los últimos datos de la base de datos.

El método 'actualizar\_datos' de la clase 'Ventana' actualiza los datos seleccionados en la tabla y en la base de datos.

El método 'eliminar\_datos' de la clase 'Ventana' borra los datos seleccionados de la tabla y de la base de datos.

El método 'limpiar\_campos' de la clase 'Ventana' limpia los campos de entrada.

El método 'guardar\_datos' de la clase 'Ventana' exporta los datos de la tabla a un fichero Excel.

El método 'change\_frame\_color' de la clase 'Ventana' cambia el color de fondo del marco.

Salidas:

El método 'inserta\_datos' de la clase 'Comunicación' inserta datos en la base de datos.

El método 'mostrar\_datos' de la clase 'Comunicación' devuelve una lista de tuplas que representan los datos.

El método 'elimina\_datos' de la clase 'Comunicación' borra datos de la base de datos.

El método 'actualiza\_datos' de la clase 'Comunicación' actualiza los datos de la base de datos.

El método 'agregar\_datos' de la clase 'Ventana' añade datos a la tabla y a la base de datos.

El método 'actualizar\_tabla' de la clase 'Ventana' actualiza la tabla con los últimos datos de la base de datos.

El método 'actualizar\_datos' de la clase 'Ventana' actualiza los datos de la tabla y de la base de datos.

El método 'eliminar\_datos' de la clase 'Ventana' borra datos de la tabla y de la base de datos.

El método 'limpiar\_campos' de la clase 'Ventana' limpia los campos de entrada.

El método 'guardar\_datos' de la clase 'Ventana' exporta los datos de la tabla a un fichero Excel.

El método 'change\_frame\_color' de la clase 'Ventana' cambia el color de fondo del marco.

En general, este script combina la comunicación con la base de datos, la validación de la entrada del usuario y la funcionalidad GUI para interactuar con una base de datos.

Pydantic

Pydantic es una herramienta de validación de datos y gestión de configuración usando notación de tipos en Python. Se puede usar en Python a partir de la versión 3.6.

La documentación se encuentra en este link: https://docs.pydantic.dev/latest/

Instalación de Pydantic:

pip3 install pydantic

pip install ‘pydantic[email]’

pip install pydantic==2.0

pip install -U pydantic

pip install email-validator