Aplicación de escritorio

Francisco José Barrera Román
1º DAW

Índice

Bibliotecas

Lo primero que tendremos que hacer será instalar las bibliotecas necesarias para poder hacer la aplicación de escritorio, estas nos permitirán hacer el diseño de la aplicación y luego a su vez poder conectarla con nuestras funciones en python.

```
# Bibliotecas
import sys
from PyQt5.QtWidgets import *
from PyQt5 import uic
from almacenamiento_datos import *
```

Primera función y clase

Lo primero que tendremos que hacer será crear una clase para meter ahí todas las funciones de todos los botones que necesitaremos conectar.

```
# Creamos una clase
class MiVentana(QMainWindow):
```

La primera función hará que la aplicación inicie y dentro de ella conectaremos los botones de la aplicación.

```python
#Creamos la primera función
def __init__(self):  fran *
    super().__init__()
    uic.loadUi( uifile: 'ropa_betis.ui', self)

# Conectar los botones del menú
    self.ver_base_datos.clicked.connect(self.cargar_datos_tabla)
    self.registrar.clicked.connect(lambda: self.stackedWidget.setCurrentWidget(self.registrar_page))
    self.actualizar.clicked.connect(lambda: self.stackedWidget.setCurrentWidget(self.actualizar_page))
    self.eliminar.clicked.connect(lambda: self.stackedWidget.setCurrentWidget(self.eliminar_page))

# Conectar botón de aceptar
    self.bot_aceptar.clicked.connect(self.crear_nueva_prenda)
    self.bot_eliminar.clicked.connect(self.eliminar_prenda)
    self.bot_buscar.clicked.connect(self.buscar_id_prenda)
    self.bot_actu.clicked.connect(self.actualizar_prenda)
```

Segunda función

Esta función nos permitirá ver la base de datos que teníamos creada dentro de la aplicación.

```python
# Cargar los datos de la base de datos a la aplicación
def cargar_datos_tabla(self):  1 usage  fran
    self.stackedWidget.setCurrentWidget(self.base_de_datos)
    ropa = consultar_datos()
    self.tabla_base_datos.setRowCount(len(ropa))

    for fila, ropa in enumerate(ropa, start=0):
        for columna, campo_ropa in enumerate(ropa.values(), start=0):
            self.tabla_base_datos.setItem(fila, columna, QTableWidgetItem(str(campo_ropa)))
    self.tabla_base_datos.resizeColumnsToContents()
```

Tercera función

Esta función lo que nos permitirá hacer es crear nuevos datos en la base de datos a través de la aplicación que hemos creado, ya que podremos rellenar los datos en la aplicación y directamente se guarda en la base de datos.

```
# Crear nuevos registros en la base de datos
def crear_nueva_prenda(self):  1 usage  fran
    nueva_prenda = dict()
    nueva_prenda['Nombre'] = self.input_nombre.text()
    self.input_nombre.setText("")
    nueva_prenda['Foto'] = self.input_foto.text()
    self.input_foto.setText("")
    nueva_prenda['Tipo_prenda'] = self.input_tipodeprenda.text()
    self.input_tipodeprenda.setText("")
    nueva_prenda['Temporada'] = self.input_temporada.text()
    self.input_temporada.setText("")
    nueva_prenda['Precio'] = float(self.input_precio.text())
    self.input_precio.setText("")
    insertar_datos(nueva_prenda)
```

Cuarta función

Esta función nos permitirá eliminar cualquier registro de la base de datos a través del id.

```
# Eliminar datos de la base de datos
def eliminar_prenda(self):  1 usage  fran
    eliminar(self.input_eliminar.text())
    self.input_eliminar.setText("")
    QMessageBox.information(self,  title: "Éxito",  text: "Prenda eliminada con éxito")
```

Quinta función

Esta función nos permitirá buscar por id cualquier registro de la base de datos y mostrarlo por pantalla en la aplicación que hemos creado para posteriormente editarla y subirla a la base de datos para ello tenemos dos funciones:
La primera que estará situada en el archivo anterior donde tenemos todo el almacenamiento de datos y lo que hará será conectarse con la base de datos para buscar el registro a través del id.

```
# Buscar por id
def buscar_id(id):  1 usage  new *
    conexion = conectar_bbdd()
    cursor = conexion.cursor(dictionary=True)
    select_query = "select * from ropa where id =" + str(id)
    cursor.execute(select_query)
    lista_ropa = cursor.fetchall()
    conexion.close()
    return lista_ropa
```

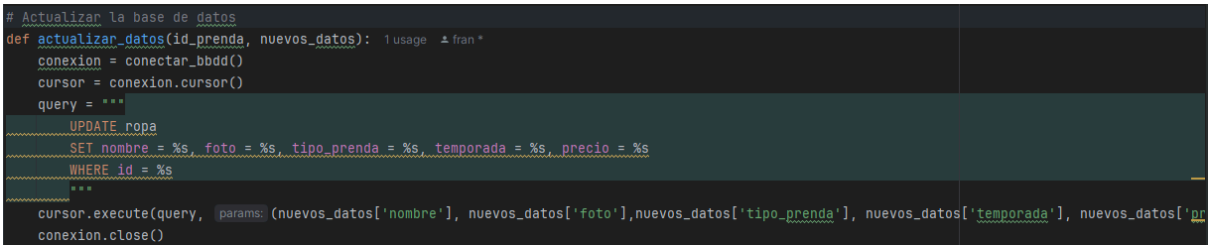
La segunda lo que hará será poner en la aplicación en su lugar correspondiente cada valor que hayamos metido dentro de la tabla en la base de datos.

```
# Buscar por id en la base de datos
def buscar_id_prenda(self):  1 usage  new *
    id = self.line_id.text()
    lista_ropa = buscar_id(id)

    self.linea_nombre.setText(lista_ropa[0]['nombre'])
    self.linea_foto.setText(lista_ropa[0]['foto'])
    self.linea_prenda.setText(lista_ropa[0]['tipo_prenda'])
    self.linea_temporada.setText(lista_ropa[0]['temporada'])
    self.linea_precio.setText(str(lista_ropa[0]['precio']))
```

Sexta función

Esta función lo que hará será guardar en la base de datos los cambios que hayamos realizado. Consta también de dos funciones:
La primera que estará situada en el archivo de almacenamiento de datos y está conectada a la base de datos y lo que hará será actualizar la base de datos con los cambios que hayamos realizado.



La segunda que lo que hará es recoger los nuevos datos que se hayan editado para posteriormente llamar a la función que hemos explicado arriba para poder actualizar la base de datos.