

# Desarrollo de Interfaces



I.E.S. Carrillo Salcedo

2º Desarrollo de Aplicaciones  
Multiplataforma

Curso 2024 / 2025

Tema 03 – Boletín 01

Trabajo Realizado por Equipo 01:

Molero Marín, Juan

Rodríguez López, Julio

Martínez Lorda, Julián

## Contenido

0.- Pasos Previos – Despliegue de la Aplicación: .....	3
Entorno Virtual:.....	3
Docker y BBDD PostgreSQL: .....	5
1.- Creación de Componentes:.....	6
1.1.- Custom Button: .....	6
1.2.- Custom Search Bar: .....	7
1.3.- Custom Table View: .....	10
2.- Eventos y Señales de los Componentes:.....	14
3.- Empaquetado de los Componentes:.....	16
3.1.- Archivo setup.py: .....	18
4.- Pruebas / Test Unitarios:.....	19
4.1.- Pruebas Custom Button – test_custom_button.py: .....	19
4.2.- Pruebas Custom Search Bar – test_custom_search_bar.py: .....	20
4.3.- Pruebas Custom Table View – test_custom_tableview.py: .....	21
5.- Bibliografía: .....	22
6.- Enlace a Repositorio Github: .....	23

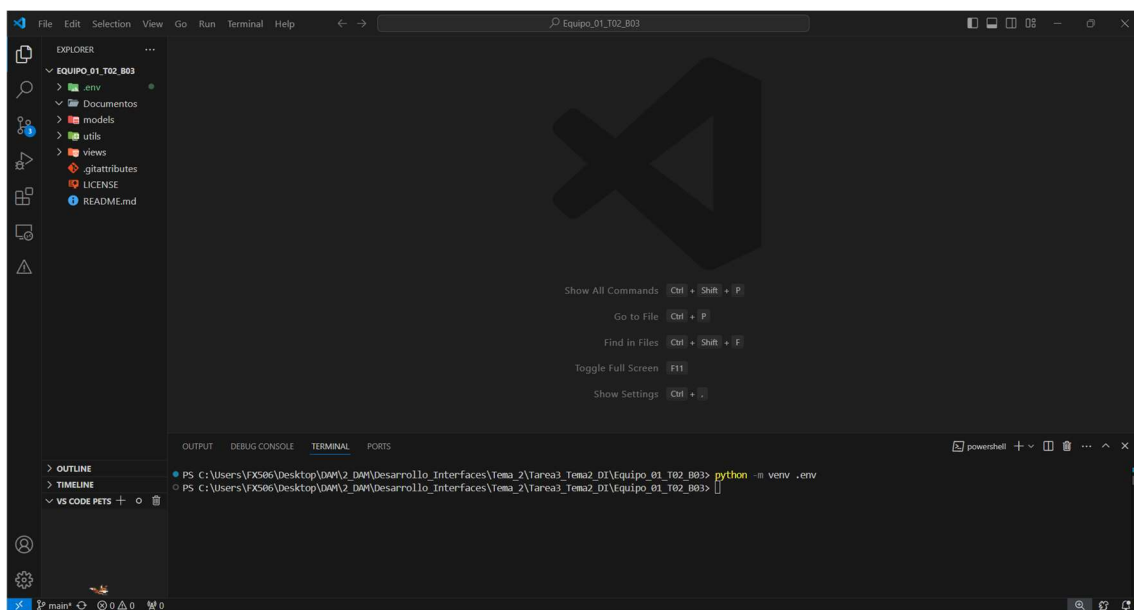
## 0.- Pasos Previos – Despliegue de la Aplicación:

### Entorno Virtual:

De forma previa a comenzar la realización de este proyecto, es necesario instalar el entorno virtual para poder trabajar, ya que entre otros aspectos fundamentales, trae integrado todas las dependencias que necesitaremos para poder desarrollar nuestro código, permitiendo instalar en cada proyecto lo necesario sin tener que interferir dentro de otros.

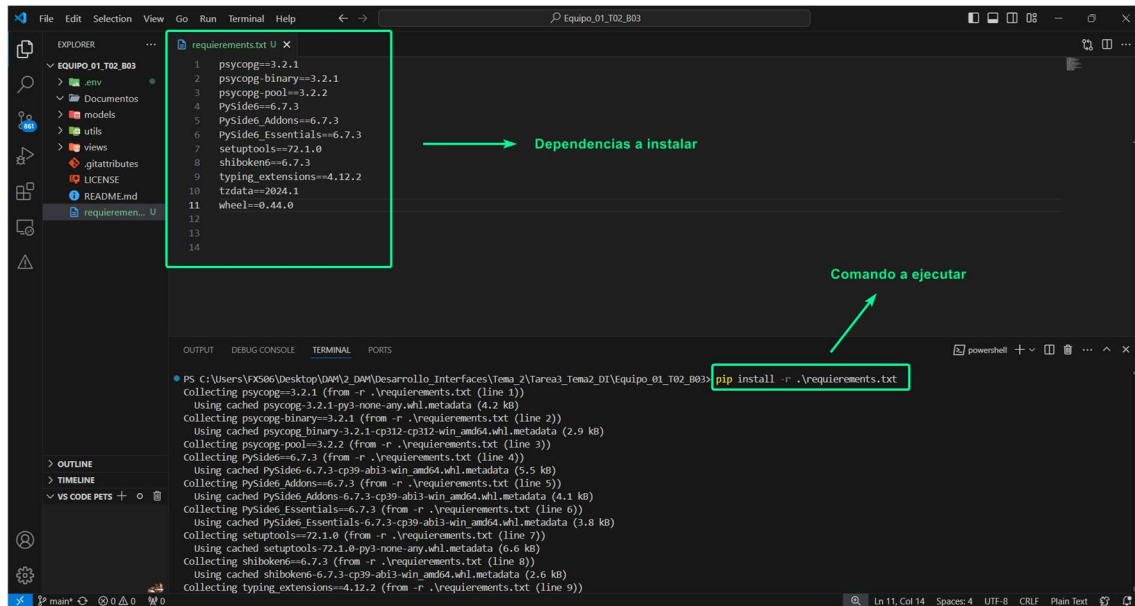
Para ello, lo primero que tenemos que hacer es crearnos un directorio, y desde VSC navegar hacia él. Una vez estemos dentro de la raíz del mismo, ejecutaremos el siguiente comando:

**python -m venv 'nombre del entorno':**



Una vez se haya creado el entorno virtual, lo siguiente será instalar las dependencias respectivamente, las cuáles se incluyen dentro de un archivo denominado “requirements.txt”, el cual deberá ser modificado en función de las especificaciones de cada proyecto, respectivamente. Es por ello que nos traemos dicho archivo a la raíz de nuestro proyecto, y posteriormente, ejecutamos el comando:

## pip install -r requirements.txt



The screenshot shows the Visual Studio Code interface. In the Explorer panel on the left, the file `requirements.txt` is selected. The file's content is displayed in the editor, listing various Python packages and their versions. A green box highlights the file name in the Explorer, with an arrow pointing to it labeled "Dependencias a instalar". In the Terminal panel at the bottom, the command `pip install -r .\requirements.txt` has been entered. A green arrow points to this command with the label "Comando a ejecutar". The terminal output shows the progress of the installation, including messages like "Collecting psycogp==3.2.1" and "Using cached psycogp-3.2.1-py3-none-any.whl.metadata (4.2 kB)".

```
requirements.txt
1 psycogp==3.2.1
2 psycogp-binary==3.2.1
3 psycogp-pool==3.2.2
4 PySide6==6.7.3
5 PySide6-Addons==6.7.3
6 PySide6-Essentials==6.7.3
7 setuptools==72.1.0
8 shiboken6==6.7.3
9 typing_extensions==4.12.2
10 tzdata==2024.1
11 wheel==0.44.0
12
13
14
```

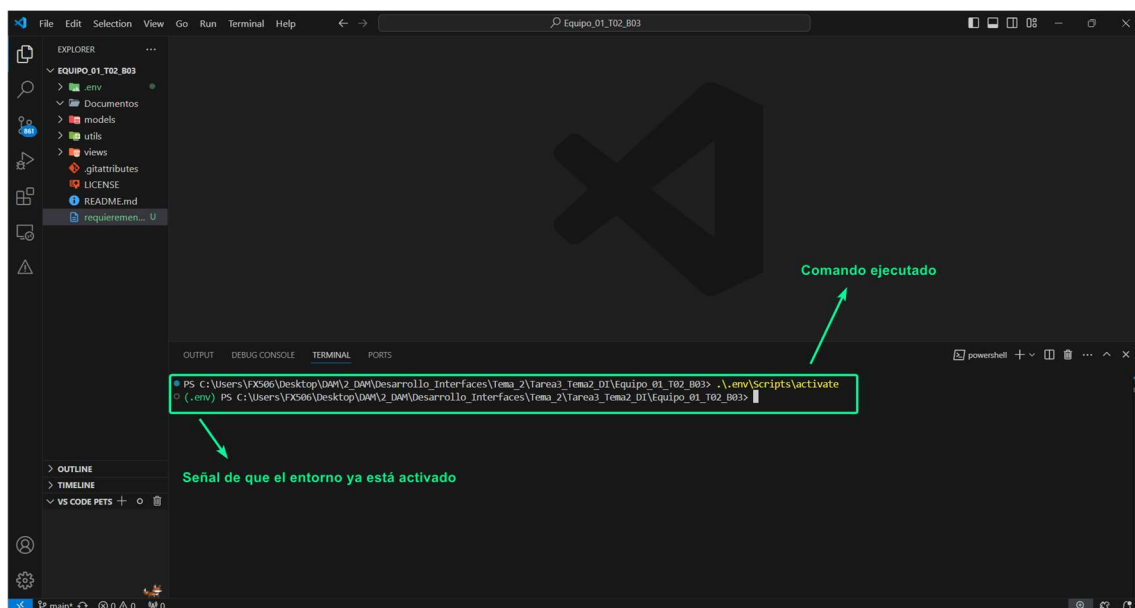
```
PS C:\Users\FX566\Desktop\DAW\2_DAW\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_DI\Equipo_01_T02_B03> pip install -r .\requirements.txt
Collecting psycogp==3.2.1 (from -r .\requirements.txt (line 1))
  Using cached psycogp-3.2.1-py3-none-any.whl.metadata (4.2 kB)
Collecting psycogp-binary==3.2.1 (from -r .\requirements.txt (line 2))
  Using cached psycogp-binary-3.2.1-cp312-cp312-win_amd64.whl.metadata (2.9 kB)
Collecting psycogp-pool==3.2.2 (from -r .\requirements.txt (line 3))
  Using cached psycogp-pool-3.2.2-cp39-cp312-win_amd64.whl.metadata (5.5 kB)
Collecting PySide6==6.7.3 (from -r .\requirements.txt (line 4))
  Using cached PySide6-6.7.3-cp39-cp312-win_amd64.whl.metadata (4.1 kB)
Collecting PySide6-Addons==6.7.3 (from -r .\requirements.txt (line 5))
  Using cached PySide6-Addons-6.7.3-cp39-cp312-win_amd64.whl.metadata (4.1 kB)
Collecting PySide6-Essentials==6.7.3 (from -r .\requirements.txt (line 6))
  Using cached PySide6-Essentials-6.7.3-cp39-cp312-win_amd64.whl.metadata (3.8 kB)
Collecting setuptools==72.1.0 (from -r .\requirements.txt (line 7))
  Using cached setuptools-72.1.0-py3-none-any.whl.metadata (6.6 kB)
Collecting shiboken6==6.7.3 (from -r .\requirements.txt (line 8))
  Using cached shiboken6-6.7.3-cp39-cp312-win_amd64.whl.metadata (2.6 kB)
Collecting typing_extensions==4.12.2 (from -r .\requirements.txt (line 9))
```

Una vez tengamos todas las dependencias que necesitamos instaladas, nuestro siguiente paso será el de activar el entorno previamente. Esto lo conseguimos mediante un comando, así como para desactivarlo y/o eliminarlo tendremos otros respectivamente:

**.\env\Scripts\activate** → Comando con el que activaremos el entorno virtual.

**deactivate** → Nos desactiva el entorno virtual.

**Remove-Item -Recurse -Force .\nombre\_entorno** → Nos elimina el entorno virtual, para lo cual primero debe de ser desactivado.



The screenshot shows the Visual Studio Code interface. In the Explorer panel on the left, the file `requirements.txt` is selected. The file's content is displayed in the editor. A large, semi-transparent watermark of the PyTorch logo is visible in the background. In the Terminal panel at the bottom, the command `.\env\Scripts\activate` has been entered. A green arrow points to this command with the label "Comando ejecutado". The terminal output shows the command being executed: `PS C:\Users\FX566\Desktop\DAW\2_DAW\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_DI\Equipo_01_T02_B03> .\env\Scripts\activate`. A green arrow points to the prompt `(.env)` in the output, with the label "Señal de que el entorno ya está activado".

```
PS C:\Users\FX566\Desktop\DAW\2_DAW\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_DI\Equipo_01_T02_B03> .\env\Scripts\activate
(.env) PS C:\Users\FX566\Desktop\DAW\2_DAW\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_DI\Equipo_01_T02_B03>
```

## Docker y BBDD PostgreSQL:

Para esta parte, se ha empleado, una herramienta de Docker denominada Docker Compose, la cual nos permite definir, ejecutar y gestionar respectivamente un conjunto de contenedores o un solo contenedor, con el objetivo de poder iniciarlos, detenerlos y/o configurarlos para que puedan gestionar el servicios postgres.

Es por ello que hemos tenido que realizar y configurar el archivo **docker-compose.yml** en la raíz del proyecto, mediante el cual con el comando **docker-compose up** podremos levantar y ejecutar nuestro contenedor de Docker, y mediante **docker-compose down** lo detendremos y eliminaremos posteriormente:

```
1 version: '3.8' # Especifica la versión de Docker Compose a usar
2
3 services:
4   postgres: # Define un servicio llamado 'postgres'
5     image: postgres:latest # Utiliza la última imagen de PostgreSQL disponible
6     container_name: tr_02_PR2 # Asigna un nombre al contenedor, en este caso 'tr_02_PR2'
7     environment: # Define las variables de entorno necesarias para la configuración de PostgreSQL
8       POSTGRES_USER: admin # Nombre de usuario para la base de datos
9       POSTGRES_PASSWORD: '0000' # Contraseña para el usuario de la base de datos
10      POSTGRES_DB: eq_01_gamevault_db # Nombre de la base de datos que se creará al iniciar el contenedor
11      PGDATA: /var/lib/postgresql/data # Ruta dentro del contenedor donde se almacenarán los datos de PostgreSQL
12    ports:
13      - "5432:5432" # Expone el puerto 5432 del contenedor en el puerto 5432 de la máquina host para acceder a la base de datos desde el exterior
14    volumes:
15      - postgres-data:/var/lib/postgresql/data # Asocia un volumen llamado 'postgres-data' a la carpeta de datos de PostgreSQL en el contenedor
16    networks:
17      - bridge # Asigna el contenedor a una red llamada 'bridge'
18
19 volumes:
20   postgres-data:
21     driver: local # Define el volumen 'postgres-data' con el controlador local, que almacena los datos en el sistema de archivos de la máquina host
22
23 networks:
24   bridge:
25     driver: bridge # Define una red de tipo bridge para la comunicación de contenedores en esta red
26
```

Con el archivo ya configurado y listo, es hora de ejecutar el comando, crear e iniciar el contenedor, y por lo tanto, iniciar el servicio para poder tener conexión con la base de datos:

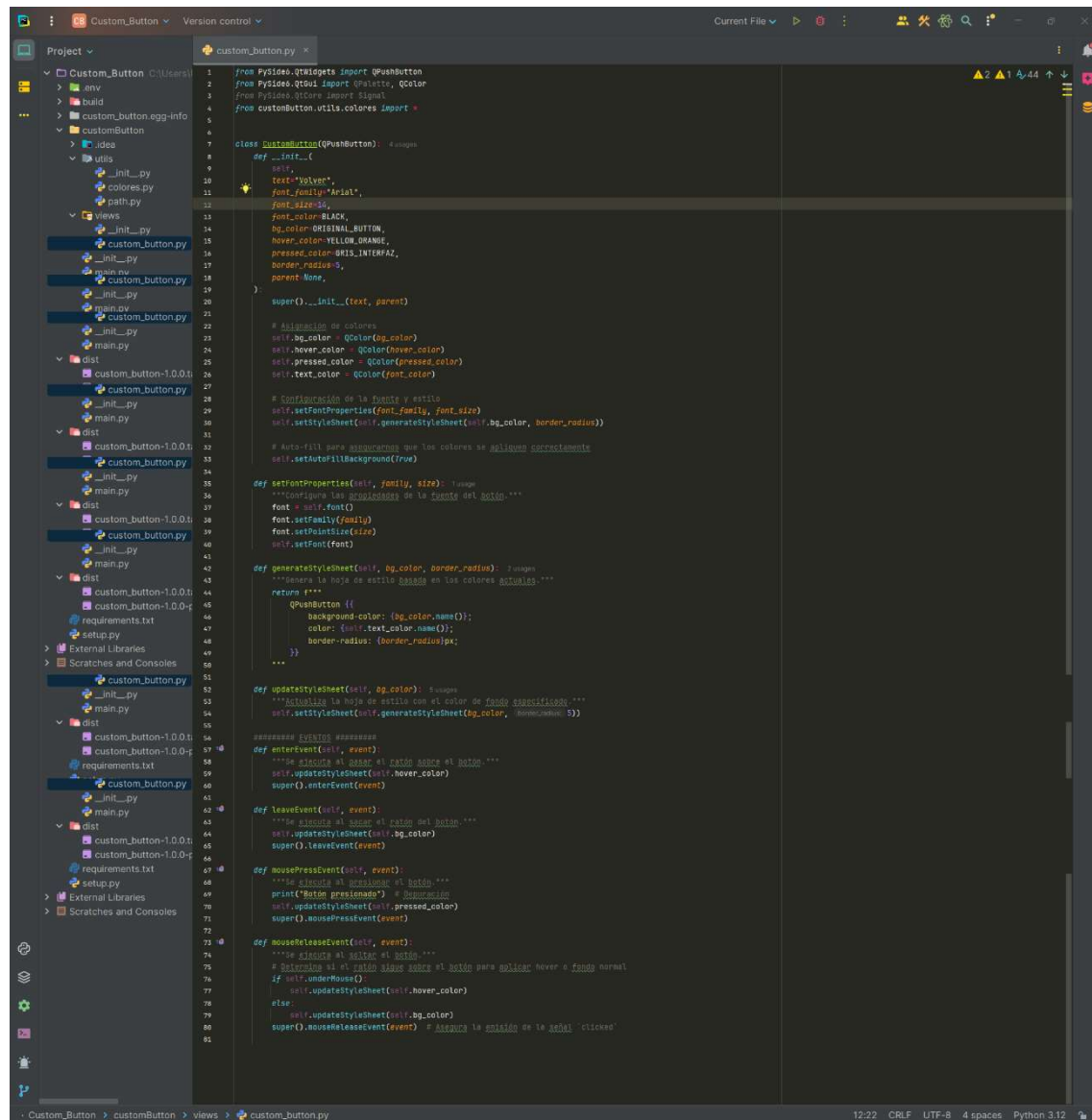
```
1 version: '3.8' # Especifica la versión de Docker Compose a usar
2
3 services:
4   postgres: # Define un servicio llamado 'postgres'
5     image: postgres:latest # Utiliza la última imagen de PostgreSQL disponible
6     container_name: tarea3_tema2_Equipol # Asigna un nombre al contenedor, en este caso 'tr_02_PR2'
7     environment: # Define las variables de entorno necesarias para la configuración de PostgreSQL
8       POSTGRES_USER: admin # Nombre de usuario para la base de datos
9       POSTGRES_PASSWORD: '0000' # Contraseña para el usuario de la base de datos
10      POSTGRES_DB: eq_01_gamevault_db # Nombre de la base de datos que se creará al iniciar el contenedor
11      PGDATA: /var/lib/postgresql/data # Ruta dentro del contenedor donde se almacenarán los datos de PostgreSQL
12    ports:
13      - "5432:5432" # Expone el puerto 5432 del contenedor en el puerto 5432 de la máquina host para acceder a la base de datos desde el exterior
14    volumes:
15      - postgres-data:/var/lib/postgresql/data # Asocia un volumen llamado 'postgres-data' a la carpeta de datos de PostgreSQL en el contenedor
16    networks:
17      - bridge # Asigna el contenedor a una red llamada 'bridge'
18
19 volumes:
20   postgres-data:
21     driver: local # Define el volumen 'postgres-data' con el controlador local, que almacena los datos en el sistema de archivos de la máquina host
22
23 networks:
24   bridge:
25     driver: bridge # Define una red de tipo bridge para la comunicación de contenedores en esta red
26
```

```
PS C:\Users\VF506\Desktop\DAW2_DAM\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_Equipol\Equipo_01_T02_B03> docker-compose up
time="2024-11-04T18:38:22+01:00" level=warning msg="C:\Users\VF506\Desktop\DAW2_DAM\Desarrollo_Interfaces\Tema_2\Tarea3_Tema2_Equipol\Equipo_01_T02_B03\docker-compose.yml: the attribute 'version' is obsolete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 2/2
✔ Network equipo_01_t02_b03_bridge Created 0.0s
✔ Container tarea3_tema2_Equipol Created 0.1s
Attaching to tarea3_tema2_Equipol
tarea3_tema2_Equipol | PostgreSQL Database directory appears to contain a database; Skipping initialization
tarea3_tema2_Equipol | 2024-11-04 17:38:22.668 UTC [1] LOG: starting PostgreSQL 17.0 (Debian 17.0-1.pgd120+1) on x86_64-pc-linux-gnu, compiled by gcc (Debian 12.2.0-14) 12.2.0, 64-bit
tarea3_tema2_Equipol | 2024-11-04 17:38:22.668 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
tarea3_tema2_Equipol | 2024-11-04 17:38:22.668 UTC [1] LOG: listening on IPv6 address "::", port 5432
tarea3_tema2_Equipol | 2024-11-04 17:38:22.674 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSOCKET.5432"
```

## 1.- Creación de Componentes:

### 1.1.- Custom Button:

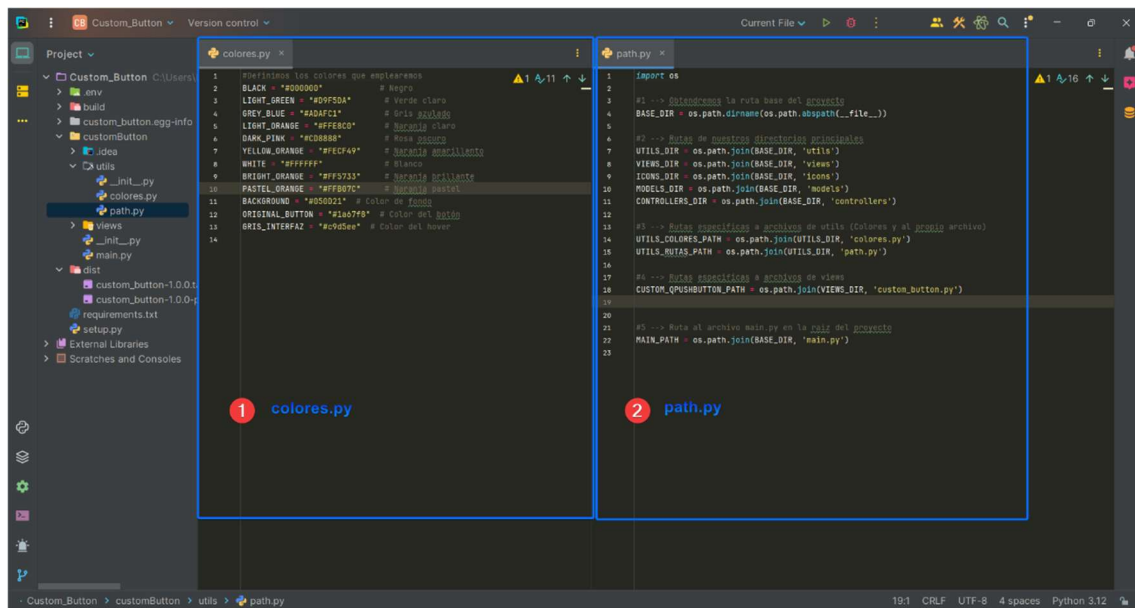
Nuestro primer componente para la nueva ventana de “**Búsqueda**” será un botón, el cuál será el encargado de redirigirnos nuevamente a la ventana de “**Login**” una vez el usuario haya finalizado las consultas necesarias. Para ello en un proyecto independiente, hemos desarrollado la siguiente clase, ya que posteriormente deberá ser empaquetada dentro de nuestro proyecto principal:



```
1 from PySide6.QtWidgets import QPushButton
2 from PySide6.QtGui import QPainter, QColor
3 from PySide6.QtCore import Signal
4 from customButton.utils.colores import *
5
6
7 class CustomButton(QPushButton):
8     def __init__(self, text="Botón", font_family="Arial", font_size=14, bg_color=ORIGINAL_BUTTON, hover_color=YELLOW_ORANGE, pressed_color=GRIS_INTERFAZ, border_radius=5, parent=None):
9         super().__init__(text, parent)
10
11         # Configuración de colores
12         self.bg_color = QColor(bg_color)
13         self.hover_color = QColor(hover_color)
14         self.pressed_color = QColor(pressed_color)
15         self.text_color = QColor(font_color)
16
17         # Configuración de la fuente y estilo
18         self.setFont(font_family, font_size)
19         self.setStyleSheet(self.generateStyleSheet(self.bg_color, border_radius))
20
21         # Auto-fill para asegurarnos que los colores se aplican correctamente
22         self.setAutoFillBackground(True)
23
24     def setFontProperties(self, font_family, font_size):
25         """Configura las propiedades de la fuente del botón"""
26         font = QFont()
27         font.setFamily(font_family)
28         font.setPointSize(font_size)
29         self.setFont(font)
30
31     def generateStyleSheet(self, bg_color, border_radius):
32         """Genera la hoja de estilo basada en los colores actuales"""
33         return f"""
34         QPushButton {{
35             background-color: {bg_color.name()};
36             color: {self.text_color.name()};
37             border-radius: {border_radius}px;
38         }}
39         """
40
41     def updateStyleSheet(self, bg_color):
42         """Actualiza la hoja de estilo con el color de fondo especificado"""
43         self.setStyleSheet(self.generateStyleSheet(bg_color, self.borderRadius()))
44
45     @abstractmethod
46     def enterEvent(self, event):
47         """Se ejecuta al pasar el ratón sobre el botón"""
48         self.updateStyleSheet(self.hover_color)
49         super().enterEvent(event)
50
51     def leaveEvent(self, event):
52         """Se ejecuta al quitar el ratón del botón"""
53         self.updateStyleSheet(self.bg_color)
54         super().leaveEvent(event)
55
56     def mousePressEvent(self, event):
57         """Se ejecuta al presionar el botón"""
58         print("Botón presionado") # Suprimir
59         self.updateStyleSheet(self.pressed_color)
60         super().mousePressEvent(event)
61
62     def mouseReleaseEvent(self, event):
63         """Se ejecuta al soltar el botón"""
64         # Restablece el color al estado normal
65         if self.isUnderMouse():
66             self.updateStyleSheet(self.hover_color)
67         else:
68             self.updateStyleSheet(self.bg_color)
69         super().mouseReleaseEvent(event) # Asegura la emisión de la señal 'clicked'
```



Además de ello, también hemos incluido un módulo para los colores, y otro para las rutas, tal y como podemos apreciar aquí:



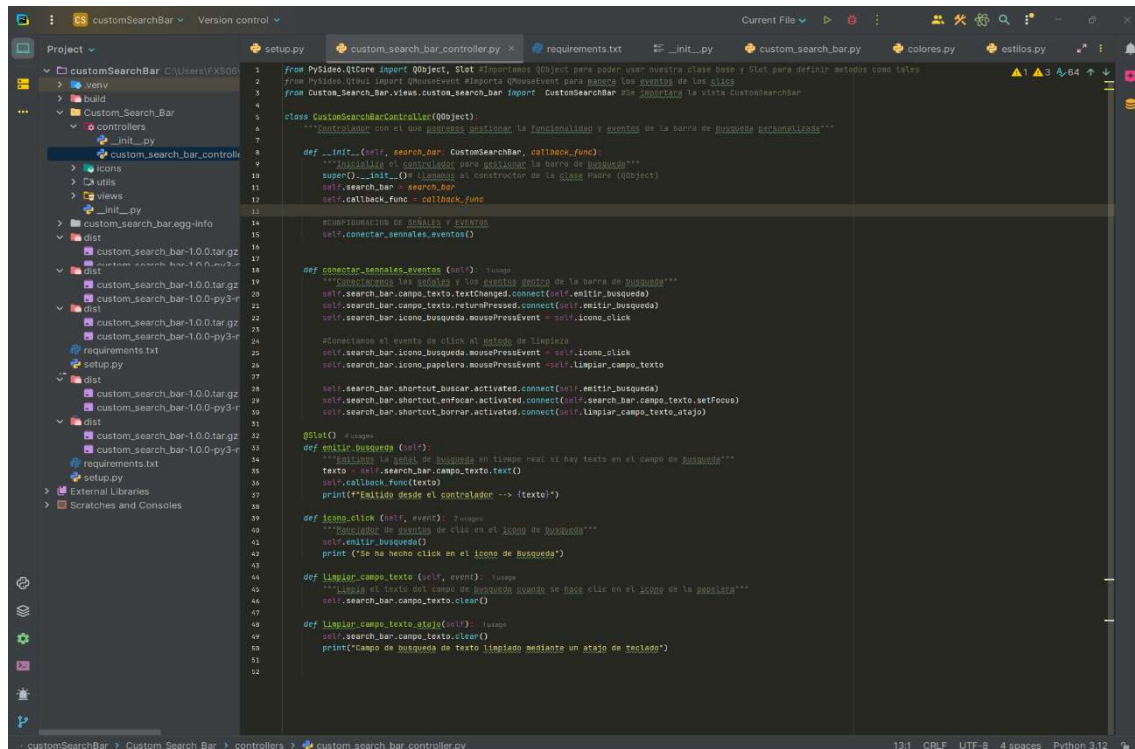
```

1 #Definimos los colores que emplearemos
2 BLACK = "#000000" # Negro
3 LIGHT_GREEN = "#90EE90" # Verde claro
4 GREY_BLUE = "#ADD8E6" # Azul grisáceo
5 LIGHT_ORANGE = "#FFDAB9" # Naranja claro
6 DARK_PINK = "#DC143C" # Rosa oscuro
7 YELLOW_ORANGE = "#FFD700" # Naranja brillante
8 WHITE = "#FFFFFF" # Blanco
9 BRIGHT_ORANGE = "#FF8C00" # Naranja brillante
10 PASTEL_ORANGE = "#FFD700" # Naranja pastel
11 BACKGROUND = "#D3D3D3" # Color de fondo
12 ORIGINAL_BUTTON = "#D3D3D3" # Color del botón
13 GRIS_INTERFAZ = "#C0C0C0" # Color del hover
14
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

```

## 1.2.- Custom Search Bar:

### 1.2.1.- Custom Seach Bar Controller:



```

1 from PySide6.QtCore import QObject, Slot
2 from PySide6.QtGui import QMouseEvent
3 from Custom_Search_Bar.Views.custom_search_bar import CustomSearchBar
4
5 class CustomSearchBarController(QObject):
6     """Controlador de la barra de búsqueda"""
7
8     def __init__(self, search_bar: CustomSearchBar, callback_func):
9         """Inicializa el controlador para gestionar la barra de búsqueda"""
10         super().__init__()
11         self.search_bar = search_bar
12         self.callback_func = callback_func
13
14     @Slot()
15     def conectar_senales_y_eventos(self):
16         """Conecta las señales y los eventos de la barra de búsqueda"""
17         self.search_bar.campo_texto.textChanged.connect(self.emitir_búsqueda)
18         self.search_bar.campo_texto.returnPressed.connect(self.emitir_búsqueda)
19         self.search_bar.icona_búsqueda.mousePressEvent = self.icona_click
20
21         #Conectamos el evento de click al botón de limpieza
22         self.search_bar.icona_búsqueda.mousePressEvent = self.icona_click
23
24         #Conectamos el evento de click al botón de búsqueda
25         self.search_bar.shortcut_buscador.activated.connect(self.emitir_búsqueda)
26         self.search_bar.shortcut_enfocar.activated.connect(self.search_bar.campo_texto.setFocus)
27         self.search_bar.shortcut_borrar.activated.connect(self.limpiar_campo_texto)
28
29     @Slot()
30     def emitir_búsqueda(self):
31         """Envía la señal de búsqueda en tiempo real si hay texto en el campo de búsqueda"""
32         texto = self.search_bar.campo_texto.text()
33         self.callback_func(texto)
34         print(f"Enviado desde el controlador -> {texto}")
35
36     def icona_click(self, event):
37         """Envía la señal de click al botón de búsqueda"""
38         self.emitir_búsqueda()
39         print(f"Se ha hecho click en el icono de Búsqueda")
40
41     def limpiar_campo_texto(self, event):
42         """Envía la señal de click al botón de limpieza"""
43         self.search_bar.campo_texto.clear()
44
45     def limpiar_campo_texto_atras(self):
46         """Envía la señal de click al botón de limpieza"""
47         self.search_bar.campo_texto.clear()
48         print("Campo de búsqueda de texto limpiado mediante un atajo de teclado")
49
50
51
52

```

En el módulo, hemos implementado todas las funciones que poseerá nuestra barra de búsqueda, donde mediante slots hemos implementado funcionalidades y eventos.

### 1.2.2.-Custom Search Bar:

```
1 # Importación de todos los elementos que necesitaremos desde PySide
2 from PySide.QtWidgets import QWidget, QLineEdit, QLabel, QVBoxLayout, QHBoxLayout
3 from PySide.QtCore import Qt, QSize, Slot, Signal
4 from PySide.QtGui import QPixmap, QCursor, QKeySequence, QShortcut, QIcon
5
6 import Custom_Search_Bar.utils.path as path
7 import Custom_Search_Bar.utils.colores as color
8 import Custom_Search_Bar.utils.etiquetas as etiqueta
9
10
11 # Definición de la clase CustomSearchBar que hereda de QWidget para representar un widget personalizado de búsqueda
12 class CustomSearchBar(QWidget):
13     def __init__(self, estilo = None, parent=None):
14         """Constructor de la clase CustomSearchBar"""
15         # llama al constructor de la clase Base QWidget con el parámetro parent
16         super().__init__(parent)
17
18     def configurar_icono_búsqueda(self):
19         """Configurar el QLabel del icono de búsqueda"""
20         self.icono_búsqueda = QLabel() # Creamos un QLabel para el icono de búsqueda
21         pixmap = QPixmap(path.SCAN_ICON_PATH)
22
23         if not pixmap.isNull():
24             pixmap = pixmap.scaled(20,20,Qt.AspectRatioMode.KeepAspectRatio, Qt.TransformationMode.SmoothTransformation)
25             self.icono_búsqueda.setPixmap(pixmap)
26         else:
27             print("Error, no hemos podido cargar el icono de búsqueda")
28
29         self.icono_búsqueda.setAlignment(Qt.AlignmentFlag.AlignCenter)
30         self.icono_búsqueda.setFixedSize(QSize(30,30))
31         self.icono_búsqueda.setStyleSheet("background-color: transparent;")
32         self.icono_búsqueda.setCursor(QCursor(Qt.CursorShape.PointingHandCursor))
33
34     def configuracion_campo_texto(self):
35         """Configurar el QLineEdit del campo de texto"""
36         self.campo_texto = QLineEdit()
37         self.campo_texto.setPlaceholderText(etiqueta.PLACEHOLDER_CAMPO_BUSQUEDA)
38         self.campo_texto.setStyleSheet(etiqueta.STYLE_Campo_Busqueda)
39         self.campo_texto.setMinimumWidth(200)
40         self.campo_texto.setMaximumWidth(400)
41
42     def configuracion_icono_papelera(self):
43         """Configurar el QLabel del icono de la papelera"""
44         self.icono_papelera = QLabel() # Creamos el QLabel para el icono de la papelera
45         pixmap = QPixmap(path.TRASH_ICON_PATH)
46
47         if not pixmap.isNull():
48             pixmap = pixmap.scaled(20,20, Qt.AspectRatioMode.KeepAspectRatio, Qt.TransformationMode.SmoothTransformation) # Escala el pixmap
49             self.icono_papelera.setPixmap(pixmap)
50         else:
51             print("Error: no se pudo cargar el icono de la papelera")
52
53         self.icono_papelera.setAlignment(Qt.AlignmentFlag.AlignCenter)
54         self.icono_papelera.setFixedSize(QSize(30,30))
55         self.icono_papelera.setStyleSheet("background-color: transparent;")
56         self.icono_papelera.setCursor(QCursor(Qt.CursorShape.PointingHandCursor))
57
58     def configuracion_atajos(self):
59         """Configuración de los atajos de teclado"""
60         self.shortcut_buscar = QShortcut(QKeySequence("Ctrl+B"), self)
61         self.shortcut_enfocar = QShortcut(QKeySequence("Ctrl+E"), self)
62         self.shortcut_borrar = QShortcut(QKeySequence("Ctrl+Q"), self)
63
64     def configuracion_layouts(self):
65         """Configuración de los layouts de la interfaz"""
66         # Configuración del layout horizontal
67         horizontal_layout = QHBoxLayout()
68         horizontal_layout.addWidget(self.icono_búsqueda)
69         horizontal_layout.addWidget(self.campo_texto)
70         horizontal_layout.addWidget(self.icono_papelera)
71         horizontal_layout.setSpacing(5)
72         horizontal_layout.setAlignment(Qt.AlignmentFlag.AlignTop | Qt.AlignmentFlag.AlignCenter)
73
74         # Configuración del layout vertical
75         vertical_layout = QVBoxLayout()
76         vertical_layout.addWidget(horizontal_layout)
77         vertical_layout.setAlignment(Qt.AlignmentFlag.AlignTop | Qt.AlignmentFlag.AlignCenter)
78         vertical_layout.setContentsMargins(20,20,20,20)
79
80         self.setLayout(vertical_layout)
81
82     def configurar_orden_tabulaciones(self):
83         """Configurar el orden de tabulaciones de los widgets en la vista"""
84         self.setTabOrder(self.icono_búsqueda, self.campo_texto)
85         self.setTabOrder(self.campo_texto, self.icono_papelera)
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
24
```



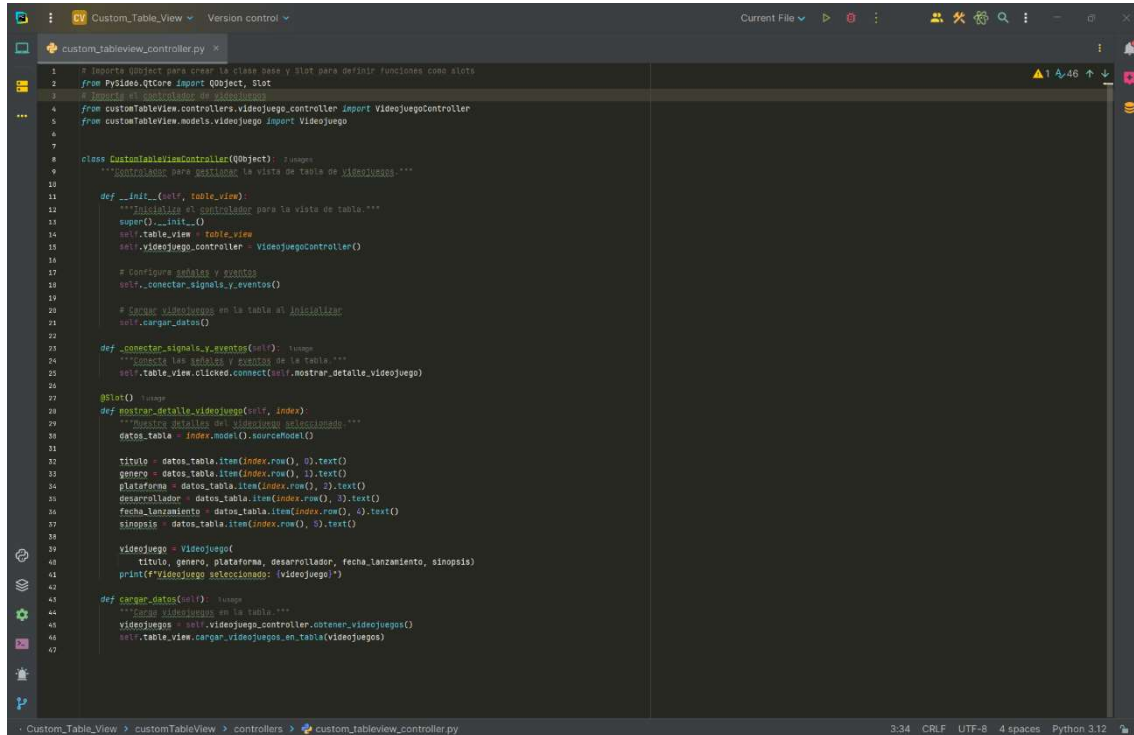
Además del controlador y de la vista respectivamente, también incluye dos iconos .png (una papelera y una lupa que acompañarán respectivamente a la barra, vinculándose en el módulo que hemos visto anteriormente), y un directorio utils, donde recogeremos los módulos **path.py**, **etiquetas.py**, **estilos.py** y **colores.py**:

The screenshot shows the VS Code interface with the 'customSearchBar' project open. The left sidebar displays the project structure, including folders like 'controllers', 'views', 'utils', and 'icons'. The main editor area shows two files: 'colores.py' (labeled 1) and 'estilos.py' (labeled 2). 'colores.py' defines color constants for the application, such as 'DARK\_BACKGROUND', 'LIGHT\_BACKGROUND', and 'DARK\_COLOR'. 'estilos.py' defines CSS styles for the application, including 'ESTILO\_CAMPUS\_TEXT', 'ESTILO\_TABLA', and 'ESTILO\_CAMPUS\_BUTTON'.

The screenshot shows the VS Code interface with the 'customSearchBar' project open. The left sidebar displays the project structure, including folders like 'controllers', 'views', 'utils', and 'icons'. The main editor area shows two files: 'etiquetas.py' (labeled 3) and 'path.py' (labeled 4). 'etiquetas.py' defines constants for the application, such as 'TITULO\_VENTANA\_PRINCIPAL', 'MENSAJE\_DE\_BUSQUEDA', and 'MENSAJE\_DE\_RESULTADO'. 'path.py' defines the file paths for the application, including 'BASE\_DIR', 'UTILS\_DIR', 'VIEWS\_DIR', 'CONTROLLER\_DIR', 'MODELS\_DIR', 'UTILS\_PATH', 'CUSTOM\_QSEARCHBAR\_PATH', and 'MAIN\_PATH'.

### 1.3.- Custom Table View:

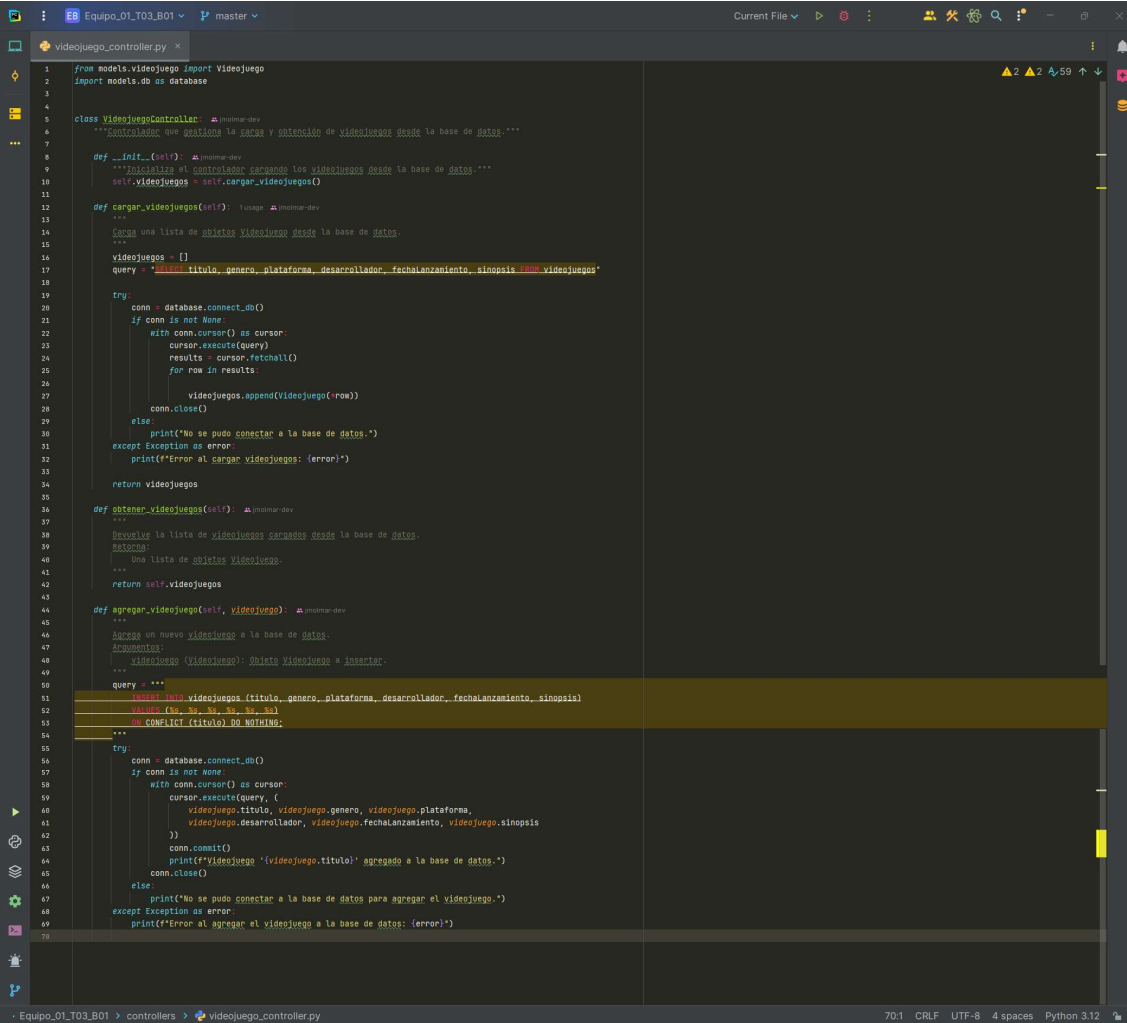
#### 1.3.1-Custom Table View Controller:



```
1 # Importa QObject para crear la clase base y slot para definir funciones como slots
2 from PySide6.QtCore import QObject, Slot
3
4 # Importa la implementación de Videojuegos
5 from customTableView.controllers.videojuego_controller import VideojuegoController
6 from customTableView.models.videojuego import Videojuego
7
8 class CustomTableViewController(QObject):
9     """Controlador para gestionar la vista de tabla de videojuegos"""
10
11     def __init__(self, table_view):
12         """Inicializa el controlador para la vista de tabla"""
13         super().__init__()
14         self.table_view = table_view
15         self.videojuego_controller = VideojuegoController()
16
17         # Configura señales y puentes
18         self._conectar_señales_y_eventos()
19
20         # Carga videojuegos en la tabla al inicializar
21         self.cargar_datos()
22
23     def _conectar_señales_y_eventos(self):
24         """Conecta las señales y puentes de la tabla"""
25         self.table_view.clicked.connect(self.mostrar_detalle_videojuego)
26
27     @Slot()
28     def mostrar_detalle_videojuego(self, index):
29         """Muestra detalles del videojuego seleccionado"""
30         datos_tabla = index.model().sourceModel()
31
32         titulo = datos_tabla.item(index.row(), 0).text()
33         genero = datos_tabla.item(index.row(), 1).text()
34         plataforma = datos_tabla.item(index.row(), 2).text()
35         desarrollador = datos_tabla.item(index.row(), 3).text()
36         fecha_lanzamiento = datos_tabla.item(index.row(), 4).text()
37         sinopsis = datos_tabla.item(index.row(), 5).text()
38
39         videojuego = Videojuego(
40             titulo, genero, plataforma, desarrollador, fecha_lanzamiento, sinopsis)
41         print(f"Videojuego seleccionado: {videojuego}")
42
43     def cargar_datos(self):
44         """Carga videojuegos en la tabla"""
45         videojuegos = self.videojuego_controller.obtener_videojuegos()
46         self.table_view.cargar_videojuegos_en_tabla(videojuegos)
```

Módulo donde vamos a implementar las funcionalidades de nuestra vista o tabla, donde se van a recoger los datos de nuestras instancias (**Videojuegos**), para lo cual vamos a necesitar por un lado una clase Controlador también para este tipo de objeto, así como el modelo donde lo hemos desarrollado también.

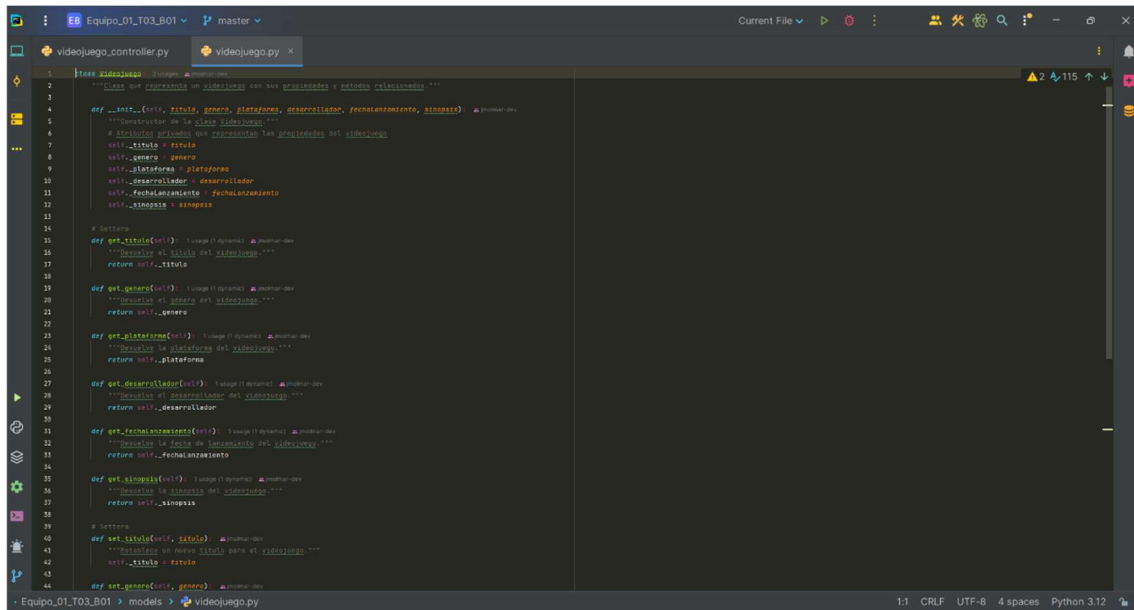
### 1.3.2.- Videojuego Controller:



```
1 from models.videojuego import Videojuego
2 import models.db as database
3
4 class VideojuegoController:
5     """Controlador que gestiona la carga y obtención de videojuegos desde la base de datos."""
6
7     def __init__(self):
8         """Inicializa el controlador cargando los videojuegos desde la base de datos."""
9         self.videojuegos = self.cargar_videojuegos()
10
11     def cargar_videojuegos(self):
12         """
13         Carga una lista de objetos Videojuego desde la base de datos.
14         """
15         videojuegos = []
16         query = "SELECT titulo, genero, plataforma, desarrollador, fechaLanzamiento, sinopsis FROM videojuegos"
17
18         try:
19             conn = database.connect_db()
20             if conn is not None:
21                 with conn.cursor() as cursor:
22                     cursor.execute(query)
23                     results = cursor.fetchall()
24                     for row in results:
25                         videojuegos.append(Videojuego(*row))
26                 conn.close()
27             else:
28                 print("No se pudo conectar a la base de datos.")
29         except Exception as error:
30             print(f"Error al cargar videojuegos: {error}")
31
32         return videojuegos
33
34     def obtener_videojuegos(self):
35         """
36         Devuelve la lista de videojuegos cargados desde la base de datos.
37         """
38         return self.videojuegos
39
40     def agregar_videojuego(self, videojuego):
41         """
42         Agrega un nuevo videojuego a la base de datos.
43         """
44         query = "INSERT INTO videojuegos (titulo, genero, plataforma, desarrollador, fechaLanzamiento, sinopsis) VALUES (%s, %s, %s, %s, %s, %s)"
45         values = (videojuego.titulo, videojuego.genero, videojuego.plataforma, videojuego.desarrollador, videojuego.fechaLanzamiento, videojuego.sinopsis)
46
47         try:
48             conn = database.connect_db()
49             if conn is not None:
50                 with conn.cursor() as cursor:
51                     cursor.execute(query, values)
52                     conn.commit()
53                 conn.close()
54             else:
55                 print("No se pudo conectar a la base de datos para agregar el videojuego.")
56         except Exception as error:
57             print(f"Error al agregar el videojuego a la base de datos: {error}")
```

Módulo donde se inicializará el controlador, cargando los videojuegos desde la base de datos para ser conectado posteriormente con la vista de la tabla, módulo que vamos a mostrar a continuación.

### 1.3.3- Modelo Videojuego:



```
1 class Videojuego:
2     """Clase que representa un videojuego con sus propiedades y métodos relacionados."""
3
4     def __init__(self, titulo, genero, plataforma, desarrollador, fechalanzamiento, sinopsis):
5         """Constructor de la clase Videojuego"""
6         # Se crean los atributos del videojuego
7         self._titulo = titulo
8         self._genero = genero
9         self._plataforma = plataforma
10        self._desarrollador = desarrollador
11        self._fechalanzamiento = fechalanzamiento
12        self._sinopsis = sinopsis
13
14    # Getters
15    def get_titulo(self):
16        """Devuelve el título del videojuego"""
17        return self._titulo
18
19    def get_genero(self):
20        """Devuelve el genero del videojuego"""
21        return self._genero
22
23    def get_plataforma(self):
24        """Devuelve la plataforma del videojuego"""
25        return self._plataforma
26
27    def get_desarrollador(self):
28        """Devuelve el desarrollador del videojuego"""
29        return self._desarrollador
30
31    def get_fechalanzamiento(self):
32        """Devuelve la fecha de lanzamiento del videojuego"""
33        return self._fechalanzamiento
34
35    def get_sinopsis(self):
36        """Devuelve la sinopsis del videojuego"""
37        return self._sinopsis
38
39    # Setters
40    def set_titulo(self, titulo):
41        """Establece el nuevo título para el videojuego"""
42        self._titulo = titulo
43
44    def set_genero(self, genero):
45        """Establece el nuevo genero para el videojuego"""
46        self._genero = genero
```

Módulo donde se ha desarrollado la clase Videojuego, la cuál a partir de ahora será determinante en el desarrollo de la asignatura, tanto para este proyecto como para otros que deberemos desarrollar posteriormente.

Es la encargada de conectarse con el controlador anteriormente mencionado para poder transmitir los datos de las instancias a la vista, respectivamente.

### 1.3.4.- Custom Table View:

Módulo donde hemos configurado el aspecto visual de la tabla, es decir, su representación gráfica dentro de nuestra nueva ventana. Además de ello, hemos implementado otras funcionalidades a destacar, como por ejemplo el de filtrar, es decir, se ha manejado que se vaya filtrando lo que el usuario va a escribir dentro de la barra de búsqueda:

```
1 from PySide6.QtWidgets import QApplication, QMainWindow, QTableView, QHeaderView, QTableWidget, QTableWidgetItem
2 from PySide6.QtCore import Qt, QSortFilterProxyModel
3 from PySide6.QtGui import QStandardItemModel, QStandardItem
4
5 import customTableview.utils.styles as styles
6
7
8 class CustomTableView(QTableView):
9     def __init__(self, estilo=None, parent=None):
10         """Constructor de la clase CustomTableView"""
11         super().__init__(parent)
12
13         # CONFIGURACIÓN DE COMPONENTES
14         self.configurar_datos_tabla()
15         self.configurar_filtro_datos()
16         self.configurar_cabeceras()
17
18         # ESTILO
19         if estilo is not None:
20             self.setStyleSheet(estilo)
21
22     def configurar_datos_tabla(self):
23         """Configura la estructura de los datos de la tabla"""
24         # Crea un modelo de datos con 4 columnas
25         self.datos_tabla = QStandardItemModel(0, 4)
26         # Establece los nombres de las columnas de la tabla
27         self.datos_tabla.setHorizontalHeaderLabels([
28             "Titulo", "Genero", "Plataforma", "Desarrollador",
29             "Fecha de Lanzamiento", "Sinopsis"
30         ])
31
32     def configurar_filtro_datos(self):
33         """Configura el filtro de datos para la búsqueda y filtrado"""
34         self.filtro_datos = QSortFilterProxyModel()
35         self.filtro_datos.setSourceModel(self.datos_tabla)
36         self.filtro_datos.setFilterCaseSensitivity(Qt.CaseSensitivity.CaseInsensitive)
37         self.filtro_datos.setFilterKeyColumn(-1)
38         self.setModel(self.filtro_datos)
39
40     def configurar_cabeceras(self):
41         """Configura las cabeceras de la tabla"""
42         self.horizontalHeader().setSectionResizeMode(QHeaderView.ResizeMode.Interactive)
43         self.horizontalHeader().setStretchLastSection(True)
44
45         for i in range(0):
46             self.horizontalHeader().setSectionResizeMode(
47                 i, QHeaderView.ResizeMode.ResizeToContents if i in [
48                     0, 1, 2, 3, 4] else QHeaderView.ResizeMode.Stretch
49             )
50
51     def cargar_videojuegos_en_tabla(self, videojuegos):
52         """Carga una lista de objetos videojuegos en la tabla"""
53
54         Args:
55             videojuegos (list): Lista de objetos videojuegos a cargar en la tabla.
56         """
57         # Limpia cualquier fila existente en la tabla
58         self.datos_tabla.setRowCount(0)
59         # Recorre la lista de videojuegos y agrega cada uno como una fila en la tabla
60         for videojuego in videojuegos:
61             fila = [
62                 # Columna de titulo
63                 QStandardItem(videojuego.get_titulo()),
64                 # Columna de genero
65                 QStandardItem(videojuego.get_genero()),
66                 # Columna de plataforma
67                 QStandardItem(videojuego.get_plataforma()),
68                 # Columna de desarrollador
69                 QStandardItem(videojuego.get_desarrollador()),
70                 # Columna de fecha de lanzamiento
71                 QStandardItem(videojuego.get_fecha_lanzamiento()),
72                 # Columna de sinopsis
73                 QStandardItem(videojuego.get_sinopsis())
74             ]
75             # Añade el texto al centro para todas las columnas
76             for i, item in enumerate(fila):
77                 if i in [0, 1, 2, 3, 4]:
78                     item.setTextAlignment(Qt.AlignmentFlag.AlignCenter)
79             # Añade la fila al modelo de datos
80             self.datos_tabla.appendRow(fila)
81
82     def filtrar_tabla(self, texto):
83         """Añade un filtro al modelo de datos para mostrar solo las filas que coincidan con el texto"""
84
85         Args:
86             texto (str): Texto de búsqueda.
87         """
88         self.filtro_datos.setFilterFixedString(texto.strip())
```

Para finalizar, también vamos a mostrar su contenido dentro del directorio utils, en este caso nuevamente serán dos módulos (**estilos.py** y **etiqueta.py**):

```
estilos.py
1 DARK_GREY = "#2E2E2E"
2 SOFT_WHITE = "#D9D9D9"
3 ACCENT_BLUE = "#4A90E2"
4 DARK_GREEN = "#1B5E20"
5 ACCENT_ORANGE = "#FF9800"
6 DEEP_RED = "#C0392B"
7 SOFT_YELLOW = "#FFEB3B"
8 PURPLE_ACCENT = "#9C27B0"
9 TEAL_COLOR = "#00897B"
10 BORDER_COLOR = ACCENT_BLUE
11 SHADOW_COLOR = "#121212"
12 PASTEL_YELLOW = "#FFF5CC"
13 HEADER_BLUE = "#3498DB"

14 # Estilo para el campo de texto
15 ESTILO_CAMPO_TEXTO = {
16     "color": (SOFT_WHITE), /* Color del texto */
17     "background-color": (DARK_BACKGROUND), /* Color de fondo */
18     "border": 1px solid (BORDER_COLOR), /* Borde del campo */
19     "border-radius": 5px; /* Bordes redondeados */
20     "padding": 5px; /* Espaciado interno */
21 }

22 # Estilo para la tabla personalizada
23 ESTILO_TABLA = {
24     "background-color": (DARK_BACKGROUND), /* Color de fondo de la tabla */
25     "color": (SOFT_WHITE), /* Color del texto */
26     "gridline-color": (BORDER_COLOR), /* Color de las líneas de la cuadrícula */
27     "border": 1px solid (BORDER_COLOR), /* Borde opcional */
28     "selection-color": (DARK_BACKGROUND), /* Color del texto en la celda seleccionada */
29 }

30 # QHeaderView: section
31 {
32     background-color: (HEADER_BLUE); /* Color de fondo del encabezado */
33     color: (SOFT_WHITE); /* Color del texto del encabezado */
34     padding: 5px; /* Espaciado interno en las secciones del encabezado */
35     border: 1px solid (BORDER_COLOR); /* Borde de las secciones del encabezado */
36 }

37 # QTableWidgetItem: section
38 {
39     background-color: (HEADER_BLUE); /* Color de fondo del botón de la sección */
40 }

etiquetas.py
1 # Archivo: utils/etiquetas.py
2 # Constante que define el título de la ventana principal
3 TITULO_VENTANA_PRINCIPAL = "Barra de Búsqueda con Tabla de Videojuegos"
4 # Constante que define el texto del tooltip para el campo de búsqueda
5 TOOLTIP_CAMPO_BUSQUEDA = "Búsqueda por código de producto o nombre o stock o número"
6 # Constante que define el texto de marcador de posición (placeholder) para el campo de búsqueda
7 PLACEHOLDER_CAMPO_BUSQUEDA = TOOLTIP_CAMPO_BUSQUEDA
```

## 2.- Eventos y Señales de los Componentes:

Estos tres componentes van a ser integrados dentro de una nueva ventana de Búsqueda, tal y como ya hemos ido explicando. Para ello, en nuestro proyecto principal han tenido que ser importados, tal y como vamos a explicar ahora un poco más adelante en el apartado del empaquetado. El funcionamiento verdaderamente es muy simple: La nueva ventana la vamos a crear mediante una nueva instancia de QMainWindow, a la cual se dirigirá el usuario tras loguearse dentro de la ventana anterior. A continuación, vamos a explicarlo con más profundidad:

```
1 from PySide6.QtWidgets import QMainWindow, QVBoxLayout, QWidget, QPushButton
2 from PySide6.QtCore import Slot
3 from views.custom_search_bar import CustomSearchBar
4
5 # CustomSearchBar
6 from custom_search_bar.controllers.custom_search_bar_controller import CustomSearchBarController
7
8 # CustomButton
9 from custom_button.views.custom_button import CustomButton
10
11 # CustomTableview
12 from custom_tableview.views.custom_tableview import CustomTableview
13
14 # CustomTableviewController
15 from custom_tableview.controllers.custom_tableview_controller import CustomTableviewController
16
17 # CustomSearchBarController
18 import utils.estilos as estilos
19
20 class BúsquedaWindow(QMainWindow):
21     def __init__(self, ventana_login=None):
22         super().__init__()
23         self.ventana_login = ventana_login
24         self.setWindowTitle("Búsqueda")
25         self.setGeometry(100, 100, 800, 600)
26
27     def configurar_ventana(self):
28         self.setWindowTitle(TITULO_VENTANA_PRINCIPAL)
29         self.setStyleSheet(ESTILOS)
30
31     def crear_widgets(self):
32         self.search_bar = CustomSearchBar(estilos.ESTILO_CAMPO_TEXTO)
33         self.table = CustomTableview(estilos.ESTILO_TABLA)
34         self.button = CustomButton("Buscar")
35         self.button.clicked.connect(self.buscar)
36
37     def asociar_controladores(self):
38         self.search_bar_controller = CustomSearchBarController(self.search_bar, self.table.filtrar_tabla)
39         self.table_controller = CustomTableviewController(self.table)
40         self.button_controller = CustomButtonController(self.button)
41
42     def configurar_layout(self):
43         self.layout = QVBoxLayout()
44         self.layout.addWidget(self.search_bar)
45         self.layout.addWidget(self.table)
46         self.layout.addWidget(self.button)
47         self.setLayout(self.layout)
```



```

45 def configurar_layout(self):
46     """Configura el layout principal de la ventana"""
47     layout_principal = QHBoxLayout()
48     layout_principal.addWidget(self.search_bar)
49     layout_principal.addWidget(self.table)
50     layout_principal.addWidget(self.button)
51
52     widget = QWidget()
53     widget.setLayout(layout_principal)
54     self.setCentralWidget(widget)
55
56
57 @QtCore.pyqtSlot()
58 def volver_al_login(self):
59     """Todo que se ejecuta al hacer clic en 'Volver a Login'"""
60     print("Botón 'Volver a Login' presionado") # Descomentar
61     try:
62         self.show() # Se llama a la ventana de búsqueda
63         if self.ventana_login.is_not_none():
64             self.ventana_login.show() # Descomentar la ventana de login
65         else:
66             print("Referencia a ventana_login no válida") # Descomentar
67     except Exception as e:
68         print(f"Error al volver a la ventana de login: {e}")
69

```

6 Añadimos los widgets al layout

7 Metodo para volver al login

Cabe destacar que es en el constructor donde todo posteriormente se llevará a cabo, menos el botón de volver al Login que se le asocia al botón en el momento de crear el widget o componente. Además de todo esto, también hemos tenido que implementar una funcionalidad nueva dentro de la clase Loguin para que el usuario, tras iniciar sesión, pueda ser redirigido a esta ventana:

```

7 class VentanaLogin(QtCore.QObject):
8     """Ventana de login"""
9     __slots__ = ("self", "parent", "search_bar", "table", "button")
10     def __init__(self, parent=None):
11         super().__init__(parent)
12         self.search_bar = CustomSearchBar()
13         self.table = CustomTable()
14         self.button = CustomButton()
15         self.configurar_layout()
16
17     def configurar_layout(self):
18         """Configura el layout principal de la ventana"""
19         layout_principal = QHBoxLayout()
20         layout_principal.addWidget(self.search_bar)
21         layout_principal.addWidget(self.table)
22         layout_principal.addWidget(self.button)
23
24         widget = QWidget()
25         widget.setLayout(layout_principal)
26         self.setCentralWidget(widget)
27
28     @QtCore.pyqtSlot()
29     def volver_al_login(self):
30         """Todo que se ejecuta al hacer clic en 'Volver a Login'"""
31         print("Botón 'Volver a Login' presionado") # Descomentar
32         try:
33             self.show() # Se llama a la ventana de búsqueda
34             if self.ventana_login.is_not_none():
35                 self.ventana_login.show() # Descomentar la ventana de login
36             else:
37                 print("Referencia a ventana_login no válida") # Descomentar
38         except Exception as e:
39             print(f"Error al volver a la ventana de login: {e}")
40
41
42 @QtCore.pyqtSlot()
43 def iniciar_sesion(self):
44     """Todo que se ejecuta al hacer clic en 'Iniciar Sesión'"""
45     try:
46         self.ventana_login.is_not_none()
47         self.ventana_login = VentanaLogin(self)
48         self.ventana_login.show()
49     except Exception as e:
50         print(f"Error al iniciar la ventana de login: {e}")
51
52
53 @QtCore.pyqtSlot()
54 def registrar_usuario(self):
55     """Todo que se ejecuta al hacer clic en 'Registrar Usuario'"""
56     try:
57         self.ventana_registro.is_not_none()
58         self.ventana_registro = VentanaRegistro(self)
59         self.ventana_registro.show()
60     except Exception as e:
61         print(f"Error al registrar el usuario: {e}")
62
63
64 @QtCore.pyqtSlot()
65 def buscar_usuario(self):
66     """Todo que se ejecuta al hacer clic en 'Buscar Usuario'"""
67     try:
68         self.ventana_busqueda.is_not_none()
69         self.ventana_busqueda = VentanaBusqueda(self)
70         self.ventana_busqueda.show()
71     except Exception as e:
72         print(f"Error al buscar el usuario: {e}")
73

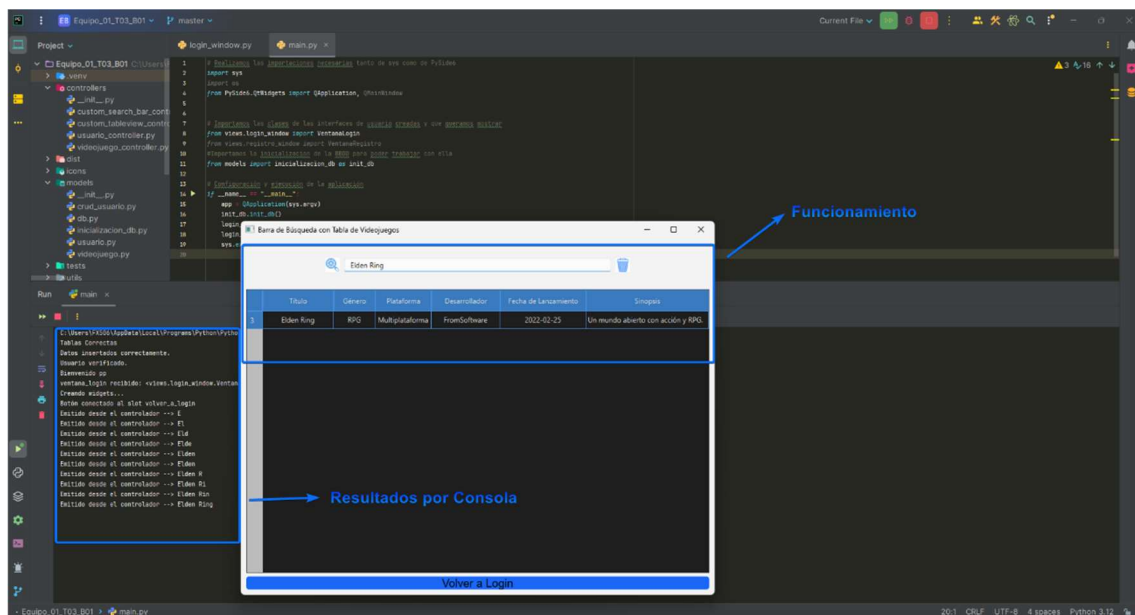
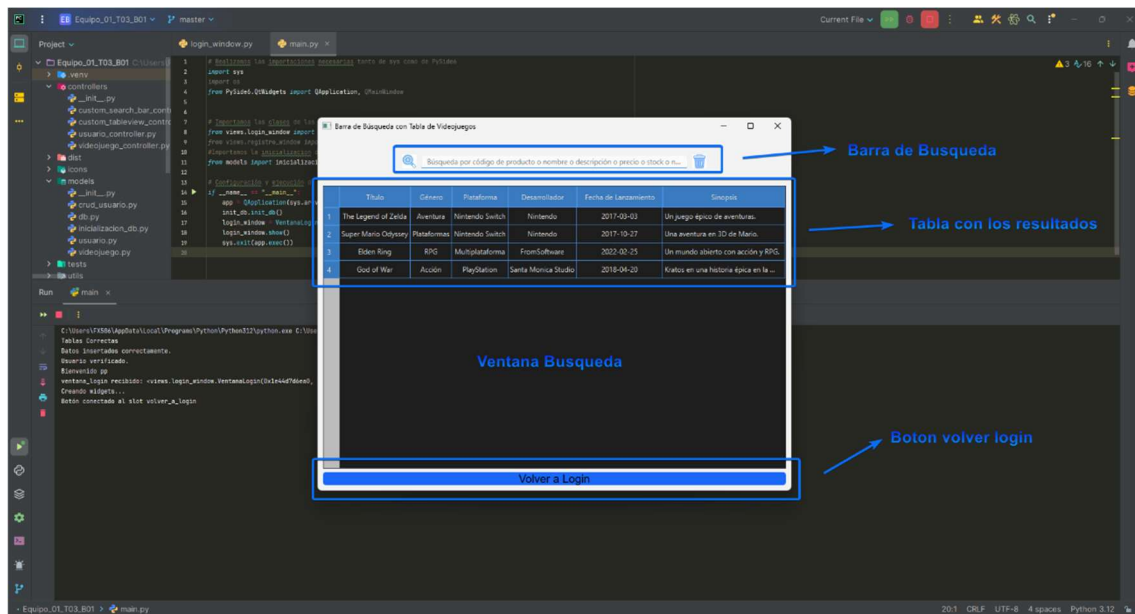
```

Todo se conecta con el boton de login

Se llamará en el método iniciar sesion

Nueva Funcionalidad implementada en Login

Nuestro resultado final ha sido el de poder navegar con nuestros usuarios creados a una nueva ventana de búsqueda, la cual tendrá tanto una barra para filtrar, como una tabla donde se mostrarán los resultados, así como un botón para volver:



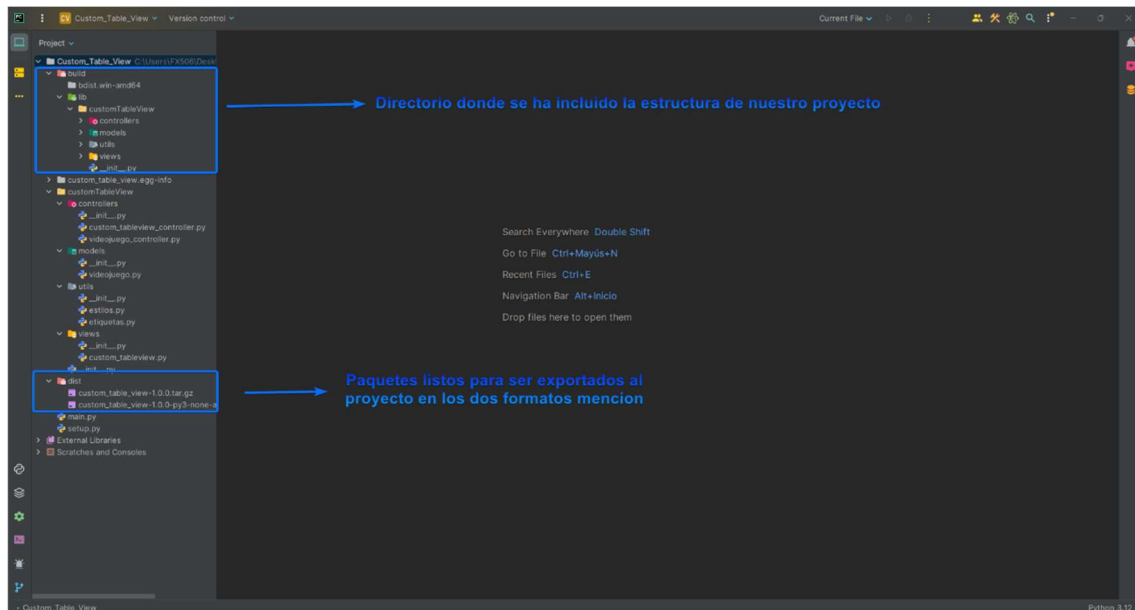
### 3.- Empaquetado de los Componentes:

Para poder usar estos tres componentes en nuestro proyecto primero ha sido necesario, tras elaborarlos en proyectos por separado como ya habíamos indicado, el proceso de empaquetarlos tanto en formatos comprimido (**tar.gz**) como en formato de instalación de módulos para Python (**whl**). Cabe destacar que para que este proceso resulte exitoso, deberemos incluir en cada directorio que deseamos exportar como un paquete distinto el archivo '**\_\_init\_\_.py**' respectivamente.

Para ello, en cada uno de los proyectos por separado, tras finalizarlos hemos tenido que ejecutar por consola el siguiente comando desde la raíz del directorio:

**python setup.py sdist bdist\_wheel**

Tras ejecutarlo, se nos incluirá en la estructura del proyecto lo siguiente:



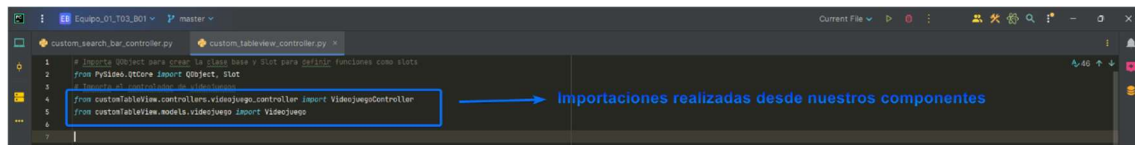
Una vez los tengamos todos listos, es hora de crear un nuevo directorio 'dist' dentro de nuestro proyecto, que será donde los hemos trasladados para ser instalados de forma posterior. Una vez allí tendremos que ejecutar los siguientes comandos:

**pip install .\dist\custom\_button-1.0.0-py3-none-any.whl**

**pip install .\dist\custom\_search\_bar-1.0.0-py3-none-any.whl**

**pip install .\dist\custom\_table\_view-1.0.0-py3-none-any.whl**

Tras ejecutarlos, cada componente será una librería nueva dentro de nuestro proyecto, y podrán ser importados dentro de cualquier clase para poder ser usados, tal y como apreciamos en el siguiente ejemplo:

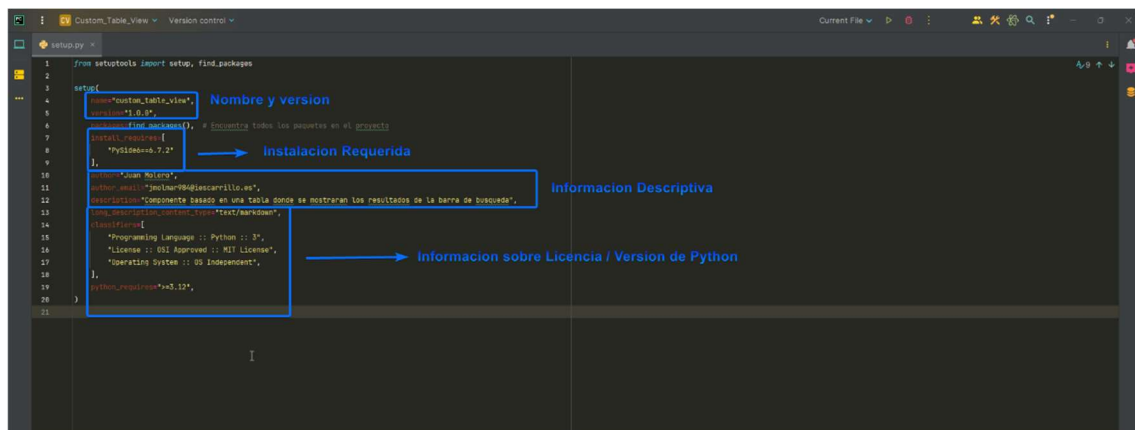


### 3.1.- Archivo setup.py:

Para que todo este proceso haya sido un éxito, en la raíz de los proyectos individuales de cada uno de los componentes hemos tenido que incluir el archivo **'setup.py'**, archivo clave dentro del proceso al poseer toda la información y configuraciones necesarias para que el mismo se lleve a cabo.

Para poder realizarlo, lo primero ha tenido que ser importar dentro del mismo la librería **'setuptools'**, de la cual hemos obtenido el propio **setup** o empaquetado, así como el **find\_packages()**, mediante el cual se localizarán todos los paquetes que deseamos empaquetar, gracias al **'\_\_init\_\_'** ya mencionado.

En nuestro caso, esta ha sido la estructura de dicho archivo:



#### 4.- Pruebas / Test Unitarios:

Para la realización de nuestras pruebas unitarias, nos hemos decantado por **pytest-qt**, herramienta o complemento de Pytest que nos permitirá la realización de las mismas.

En nuestro caso particular, hemos realizado un total de 3 pruebas para cada componente, con lo que tenemos 9 pruebas en total.

##### 4.1.- Pruebas Custom Button – test\_custom\_button.py:

```
1 import pytest
2 from PySide6.QtWidgets import QApplication
3 from PySide6.QtCore import Qt
4 from PySide6.QtGui import QColor, QPalette
5
6 from views.custom_button import CustomButton
7
8 # ===== Fixture para preparar QApplication =====
9
10
11 @pytest.fixture(scope="module", autouse=True)
12 def qt_app():
13     """
14     Crea una instancia global de QApplication para usar en las pruebas.
15     Esta instancia es necesaria para que los widgets de Qt funcionen correctamente.
16     """
17     app = QApplication.instance() or QApplication(
18         []
19     ) # QApplication o una QApplication
20     yield app
21     app.quit() # Finaliza la aplicación después de las pruebas
22
23 # ===== Fixture para CustomButton =====
24
25
26 def test_custom_button_creation():
27     """
28     Verifica que el botón personalizado se cree con las propiedades correctas.
29     - El texto es correcto.
30     - El botón es visible después de crearlo.
31     """
32     button = CustomButton(text="Probar")
33     button.show()
34     assert button.text() == "Probar" # Comprueba el texto del botón
35     assert button.isVisible() is True # Verifica que el botón está visible
36
37
38 def test_custom_button_hover_event(qt_app):
39     """
40     Verifica que el color del botón cambia correctamente al pasar el mouse sobre él.
41     """
42     button = CustomButton()
43     qt_app.addWidget(button) # Añade el botón al sistema de pruebas de Qt
44     button.show()
45     # Espera a que el botón esté completamente renderizado
46     qt_app.waitExposed(button)
47     qt_app.mouseMove(button) # Simula mover el cursor sobre el botón
48     qt_app.mouseClick() # Pasa el mouse para procesar el evento de hacer clic
49     # Verifica que el color del fondo coincide con hover_color
50     assert button.palette().color(button.backgroundRole()) == QColor(button.hover_color)
51
52
53 def test_custom_button_initial_style(qt_app):
54     """
55     Verifica que el botón personalizado se inicializa con los estilos correctos.
56     - Color de fondo inicial: blanco.
57     - Color del texto inicial: negro.
58     """
59     button = CustomButton(
60         text="Probar",
61         bg_color="#FFFFFF", # Fondo blanco
62         font_color="#000000" # Texto negro
63     )
64     qt_app.addWidget(button)
65     button.show()
66
67     # Verifica el color de fondo inicial
68     background_color = button.palette().color(button.backgroundRole())
69     assert background_color == QColor("#FFFFFF"), (
70         f"Color de fondo esperado: #FFFFFF, pero se obtuvo: {
71             background_color.name()}"
72     )
73
74     # Verifica el color del texto inicial
75     text_color = button.palette().color(QPalette.ColorRole.ButtonText)
76     assert text_color == QColor("#000000"), (
77         f"Color de texto esperado: #000000, pero se obtuvo: {
78             text_color.name()}"
79     )
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
250
```

#### 4.2.- Pruebas Custom Search Bar – test\_custom\_search\_bar.py:

```
1  > import sys
2
3  # tests fixture para custom_search_bar
4
5
6
7
8
9
10 @pytest.fixture(scope="module", autouse=True)
11 def qt_app():
12     """
13     Crea una instancia global de Application para los widgets de Qt.
14     """
15     app = QApplication.instance() or QApplication()
16     yield app
17     app.quit()
18
19 # tests fixture para CustomSearchBar
20
21
22 def test_search_bar_creation():
23     """
24     Verifica que la barra de búsqueda se crea correctamente.
25     - tiene un placeholder en el campo de texto.
26     - los iconos de búsqueda y cancelar están configurados.
27     """
28     search_bar = CustomSearchBar()
29     # El placeholder no está vacío
30     assert search_bar campo_texto.placeholderText() != ""
31     # El campo de búsqueda está configurado
32     assert search_bar.icono_búsqueda.pixmap() is not None
33     # El icono de cancelar está configurado
34     assert search_bar.icono_cancelar.pixmap() is not None
35
36
37 def test_search_bar_placeholder():
38     """
39     Comprueba que el texto del placeholder (marcador de posición) es correcto.
40     """
41     search_bar = CustomSearchBar()
42     expected_placeholder = (
43         "Búsqueda por código de producto o nombre o descripción o precio o stock o número de ventas"
44     )
45     assert search_bar campo_texto.placeholderText() == expected_placeholder
46
47
48 def test_search_bar_keyboard_shortcuts():
49     """
50     Verifica que los atajos de teclado están configurados correctamente:
51     - Ctrl+B para buscar.
52     - Ctrl+E para cancelar.
53     - Ctrl+Q para salir.
54     """
55     search_bar = CustomSearchBar()
56     assert search_bar.shortcut_buscar.key().toString() == "Ctrl+B"
57     assert search_bar.shortcut_cancelar.key().toString() == "Ctrl+E"
58     assert search_bar.shortcut_salir.key().toString() == "Ctrl+Q"
```

En la barra de búsqueda hemos verificado que se crea e inicia correctamente, que el texto del placeholder (marcador de posición) sea el correcto y que los atajos de teclado estén bien configurados.





Tras ejecutarlos, hemos obtenido resultados positivos con cada uno de ellos, tal y como se va a mostrar a continuación:

### Resultados Custom Table View:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01> pytest .\tests\test_custom_tableview.py
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0
PySide6 6.7.2 -- Qt runtime 6.7.2 -- Qt compiled 6.7.2
rootdir: C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01
plugins: qt-4.4.0
collected 3 items

tests\test_custom_tableview.py ... [100%]

===== 3 passed in 0.10s =====
PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01>
```

### Resultados Custom Search Bar:

```
PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01> pytest .\tests\test_custom_search_bar.py
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0
PySide6 6.7.2 -- Qt runtime 6.7.2 -- Qt compiled 6.7.2
rootdir: C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01
plugins: qt-4.4.0
collected 3 items

tests\test_custom_search_bar.py ... [100%]

===== 3 passed in 0.11s =====
PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01>
```

### Resultados Custom Button:

```
PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01> pytest .\tests\test_custom_button.py
===== test session starts =====
platform win32 -- Python 3.12.6, pytest-8.3.3, pluggy-1.5.0
PySide6 6.7.2 -- Qt runtime 6.7.2 -- Qt compiled 6.7.2
rootdir: C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01
plugins: qt-4.4.0
collected 3 items

tests\test_custom_button.py ... [100%]

===== 3 passed in 0.07s =====
PS C:\Users\FX506\Desktop\2DAM\Desarrollo\Interfaces\Tema_3\Tarea1_Tema3_01\Equipo_01_T03_B01>
```

## 5.- Bibliografía:

*pytest-qt* — *pytest-qt documentation*. (s. f.). Readthedocs.io. Recuperado 22 de noviembre de 2024, de <https://pytest-qt.readthedocs.io/en/latest/intro.html>

*Qt for Python*. (s. f.). Doc.qt.io. Recuperado 22 de noviembre de 2024, de <https://doc.qt.io/qtforpython-6/>

Atoche Ortega, J. (2024). *Tema 03: Desarrollo de Componentes Visuales Personalizados*. Dpto de Informática, IES Carrillo Salcedo.

**6.- Enlace a Repositorio Github:**

[https://github.com/jmolmar-dev/Equipo\\_01\\_T03\\_B01.git](https://github.com/jmolmar-dev/Equipo_01_T03_B01.git)