

**Desarrollo de interfaces**

**Tema 04 – Boletín 01**



Serpiente de color azul

Descripción generada automáticamente con confianza media

**i.E.S Carrillo salcedo**

**2º desarrollo de aplicaciones multiplataforma**

**curso 2024/2025**

**Realizado por Equipo 01:**

**Molero Marín, Juan**

**Rodríguez López, Julio**

**Martínez lorda, julián**

**Contenido**

[**1.- Despliegue Inicial del Proyecto:** 2](#_Toc184588955)

[**2.- Remodelación de la Base de Datos:** 4](#_Toc184588956)

[**3.- Arreglo de las Funcionalidades:** 10](#_Toc184588957)

[**3.1.- Creación de dos nuevos botones para exportar a PDF:** 10](#_Toc184588958)

[**3.2.- Filtración de Videojuegos por género:** 12](#_Toc184588959)

[**3.3.- Arreglo Rango Inválido en la Gráfica:** 14](#_Toc184588960)

[**3.4.- Modificación Label:** 16](#_Toc184588961)

[**4.- Bibliografía Empleada:** 17](#_Toc184588962)

# **1.- Despliegue Inicial del Proyecto:**

En este caso, para esta práctica nuestro cometido ha sido el de arreglar un proyecto proporcionado por el profesor, implementando también en el camino nuestra base de datos (videojuegos), y sustituyéndola por la de productos.

Para ello, lo primero ha sido descomprimir el proyecto y abrirlo con Visual Studio Code. Tras descomprimirlo, dentro de la carpeta de **‘deployment’** hemos tenido que crear nuestro entorno virtual, para ello hemos ejecutado el comando que venimos usando durante todo el curso:

|  |
| --- |
| **python -m venv .env** |

Una vez hemos creado nuestro entorno virtual dentro del directorio especificado, nuestro siguiente paso en esta práctica en concreto será el de activarlo y posteriormente preparar el entorno y levantar un contenedor en docker que permita la conexión con la base de datos.

Para ello, en este caso específico, se nos han proporcionado dos scripts, sin embargo, al intentar ejecutarlos saltaron fallos por la falta de una firma digital en los mismos, por lo que hemos tenido que modificar las políticas de ejecución de Windows antes de ejecutarlos mediante este comando:

**Set-ExecutionPolicy Unrestricted -Scope CurrentUser 🡪** Nos permitirá el uso de cualquier script en nuestro sistema

Una vez cambiada dicha política, es hora de ejecutar los scripts:

|  |
| --- |
| **.\activar\_venv.ps1** |

|  |
| --- |
| **.\preparar\_entorno\_y\_docker.ps1** |

**Nota 🡪** Antes de la ejecución de los scripts, se ha realizado una modificación del docker-compose.yml, tal y como se puede apreciar aquí:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Al ejecutar ambos scripts, ya tendremos el entorno listo para comenzar con la corrección del proyecto respectivamente.

A continuación vamos a realizar los cambios necesarios para poder poner en funcionamiento la aplicación.

# **2.- Remodelación de la Base de Datos:**

Una captura de pantalla de una computadora

Descripción generada automáticamenteEn nuestro caso, el proyecto se nos ha otorgado con una Base de Datos basadas en productos. Al desplegarlo de forma inicial, podremos observar lo siguiente:

Antes de ponernos manos a la obra para arreglar las funcionalidades principales de la ventana, hemos decidido previamente implementar nuestra base de datos de **‘Videojuegos’** ya, para ello, nuestro primer paso ha sido la creación de las entidades necesarias, en este caso concreto, videojuego, venta, género, usuario y rol:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Una vez hemos tenido las entidades creadas, ha sido momento de retocar el archivo SQL que inicializa la Base de Datos. Para comprobar que funciona, se realizaron previas comprobaciones dentro de DBeaver, tal y como se puede apreciar aquí:

Texto

Descripción generada automáticamente con confianza media

Tras comprobar que funciona, se ha implementado dentro del proyecto, tal y como podemos observar aquí.

**Nota: se implementaron consultas adicionales para verificar su funcionalidad:**

Texto

Descripción generada automáticamente

Una vez tenemos la base de datos, es momento de modificar la clase del controlador proporcionada, para modificar la base de datos que va a tratar respectivamente. Para ello, se implementaron los siguientes cambios dentro de la clase:

Texto

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

# **3.- Arreglo de las Funcionalidades:**

## **3.1.- Creación de dos nuevos botones para exportar a PDF:**

Uno de los requerimientos especificados ha sido la implementación de un botón que exportará los resultados obtenidos tanto en la gráfica como en la tabla de la vista a formato PDF. Para ello, hemos tomado la decisión en nuestro caso de implementar dos, separando la obtención de dichos resultados en dos documentos distintos:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Además de ello, como se puede apreciar en la primera captura, cada uno de ellos está conectado a su método para poder exportar en el formato especificado de forma concreta, los cuáles podemos apreciar a continuación:

Texto

Descripción generada automáticamente

Implementando todos estos cambios, al poner en marcha las posteriores funcionalidades que se van a explicar, se podrán generar resultados como los siguientes:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Importante 🡪** Se ha explicado el proceso para exportarlo, el cual cabe recalcar que necesitará la instalación de la librería **fpdf**. Además de todo ello, para generar los documentos, en las clases **widgets** se ha tenido que instalar también la librería **reportlab.**

## **3.2.- Filtración de Videojuegos por género:**

Una de las funcionalidades principales que debíamos de conseguir dentro de nuestra Aplicación era poder filtrar los resultados, en nuestro caso concreto hemos escogido el género del videojuego.

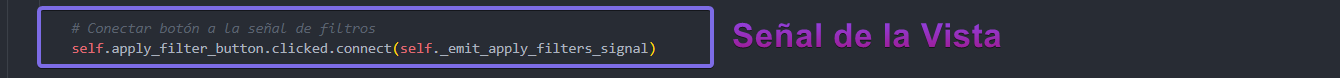
Para ello, se ha preseleccionado todas las categorías existentes dentro de nuestra Base de Datos **(Acción, Aventuras, RPG y Deportes),** así como una categoría extra denominada **‘Todos’,** el cual nos proporcionará todos los videojuegos concretamente.

Texto

Descripción generada automáticamenteTodo esto se ha conseguido mediante la implementación de modificaciones dentro de la función **apply\_filters,** situada en el controlador concretamente. Los cambios han sido los siguientes:

Captura de pantalla de computadora

Descripción generada automáticamenteDicho método se conecta con la vista mediante una señal, respectivamente:

Posteriormente, en la vista, también nos encontramos con su señal correspondiente, y con un método que nos permite aplicar los filtros para que todo funcione:

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

De la misma forma, se puede escribir dentro de la barra de búsqueda, y en función del input se irán obteniendo los resultados que coincidan, de forma independiente a la columna de la que se trate:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

## **3.3.- Arreglo Rango Inválido en la Gráfica:**

Al iniciar la aplicación, se estaba obteniendo el siguiente fallo tanto al intentar aplicar cualquier filtro o intentar usar la barra de búsqueda:

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

Para solucionarlo, se ha tenido que modificar la clase Custom Chart Widget, concretamente los métodos **set data y configure axes:**

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Una vez realizados estos cambios, la aplicación ha solventado ese error, permitiendo escribir y filtrar sin ningún tipo de error en el formato:

Captura de pantalla de computadora

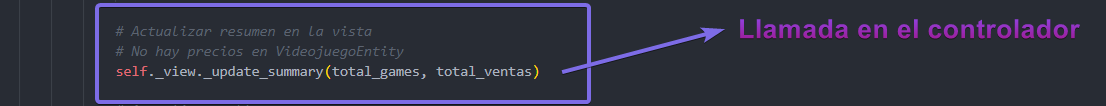
Descripción generada automáticamente

## **3.4.- Modificación Label:**

La vista principal nos traía incorporada ya una Label, sin embargo, ésta no poseía funcionalidad. En nuestro caso concreto, para que funcione, hemos implementado ciertos cambios dentro de la vista y del controlador, de tal forma que cuando filtras te mostrará el total de productos y el importe total de los mismos:

**Texto

Descripción generada automáticamenteCambios realizados en la Vista 🡪 Métodos init summary y update summary:**

Posteriormente, **update summary** será llamado en el controlador:

Gráfico

Descripción generada automáticamente con confianza bajaCon esta implementación, al realizar cualquier búsqueda o aplicar nuestro filtro, obtendremos el resultado deseado:

# **4.- Bibliografía Empleada:**

*Python Tutorial*. (s. f.). W3schools.com. Recuperado 8 de diciembre de 2024, de <https://www.w3schools.com/python/>

*Qt for Python*. (s. f.). Doc.qt.io. Recuperado 8 de diciembre de 2024, de <https://doc.qt.io/qtforpython-6/>

*Stack Overflow - Where Developers Learn, Share, & Build Careers*. (s. f.). Stack Overflow. Recuperado 8 de diciembre de 2024, de <https://stackoverflow.com/>