



Proyecto Catálogo de películas
Manual de uso

2CV5

Alumnos:

Falcón Vivanco Angel Aarón
Gómez Valentino Marco Antonio

Programación Avanzada

Profesor: Jorge Adrián Garduño Medina

Índice

Introducción y requisitos.....	pág-2
Restricciones y más requisitos.....	pág-3-7
Metodología y descripción.....	pág-8
Bibliotecas.....	pág-9
Ejecución.....	pág-10

Introducción y requisitos

El objetivo de este manual es proporcionar una guía completa para instalar, configurar y utilizar el proyecto Catálogo de Películas, desarrollado en Python.

La función de nuestro programa es tener acceso a un catálogo de películas de diferentes épocas y géneros, en donde se pueda ver una sinopsis y por supuesto el año de salida de determinada película.

Dicho esto para poder instalarlo son necesarios los siguientes requisitos:

- Python 3.7+ instalado en tu sistema.
- Biblioteca Tkinter (incluida por defecto en la mayoría de las distribuciones de Python).

Restricciones y más requisitos

Este programa no cuenta con restricciones para su uso ya que es de software libre, osea se puede modificar y distribuir libremente. Existe un repositorio en la plataforma Github al que puedes acceder con el siguiente link:

https://github.com/marcovalentino/proyecto_cat-logo_pel-culas.git

El programa cuenta con una licencia GNU General Public License v2.0, la cuál en pocas palabras:

- Puedes usar, copiar, modificar y distribuir este software de manera gratuita.
- Cualquier distribución o modificación debe estar bajo la misma licencia GPL v2.0.
- Se requiere que mantengas los avisos de copyright y licencia originales.

Para más información sobre esta licencia, puedes consultar:

<https://www.gnu.org/licenses/old-licenses/gpl-2.0.html>

Esta licencia también está especificada en el repositorio de Github.

Pensamos que la mejor forma de hacer crecer el proyecto es hacerlo de uso libre para que así pueda ser modificado y enriquecido en contenido.

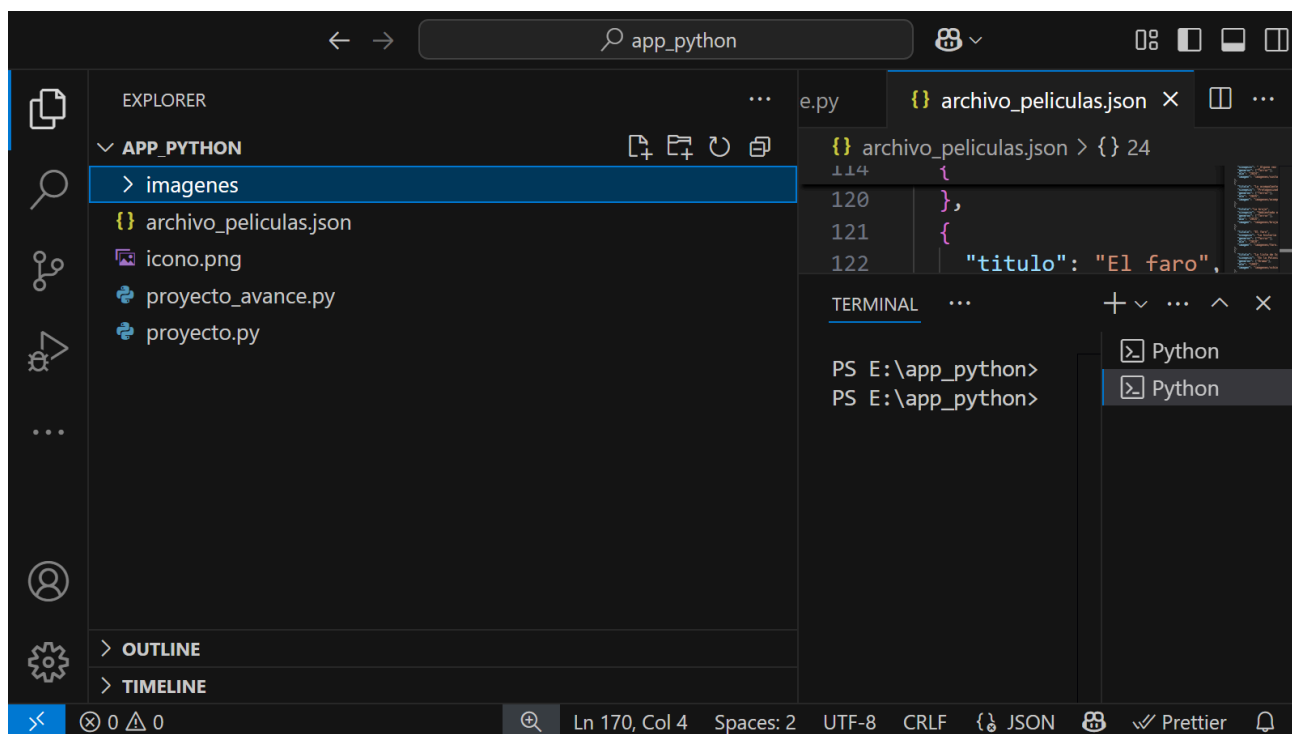
Nuestro proyecto necesita de una base de datos, la cuál está hecha en formato de archivo .json (este está especificado en el repositorio de Github).

El archivo JSON debe contener una lista de objetos con la siguiente estructura para cada película:

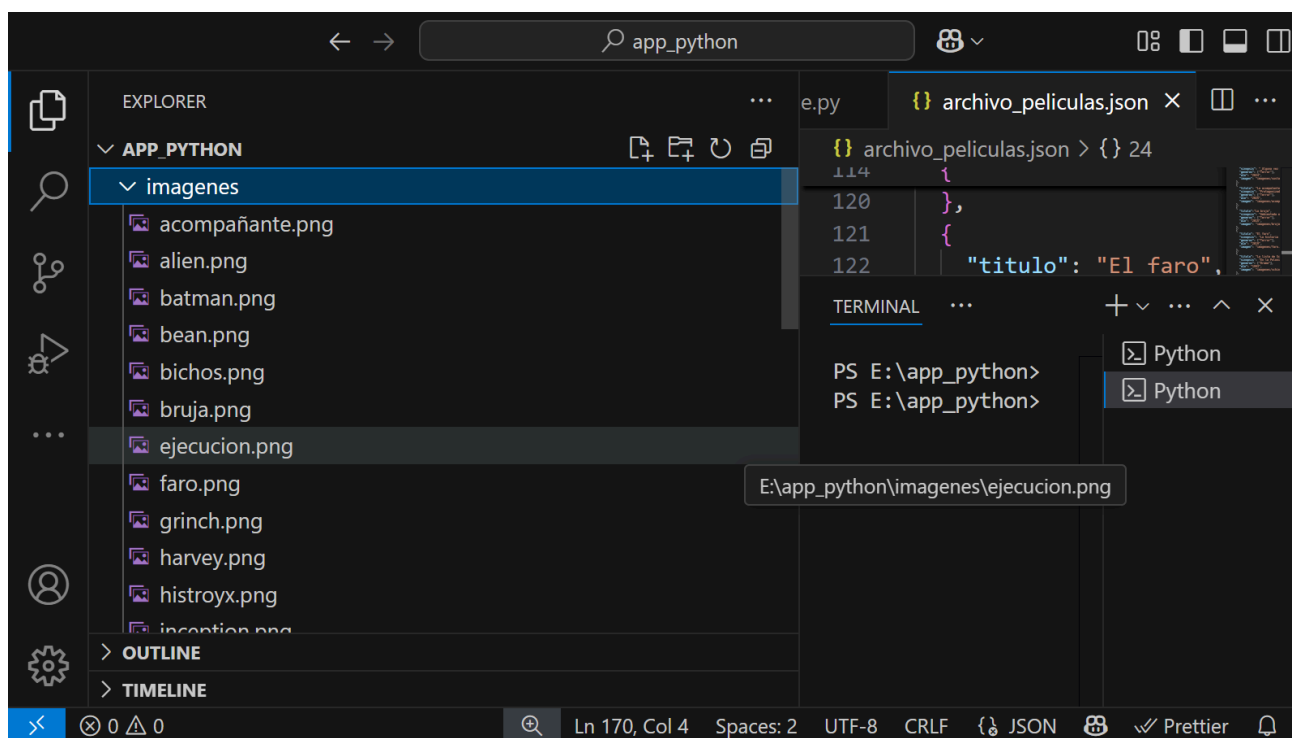
```
{
  "archivo_peliculas.json": {
    "1": [
    2      {
    3        "titulo": "Inception",
    4        "sinopsis": "Dom Cobb es un ladrón con una extraña habilidad para entrar a los sueños de la",
    5        "generos": ["Ciencia ficción"],
    6        "año": "2010",
    7        "imagen": "imagenes/inception.png"
    8      },
    9      {
   10       "titulo": "Mad Max",
   11       "sinopsis": "En un páramo postapocalíptico, una mujer se rebela contra un gobernante tiráni",
   12       "generos": ["Ciencia ficción"],
   13       "año": "2015",
   14       "imagen": "imagenes/madmax.png"
   15     },
   16     {
   17       "titulo": "Interstellar",
   18       "sinopsis": "Un equipo de exploradores viaja a través de un agujero de gusano en el espacio",
   19       "generos": ["Ciencia ficción"],
   20       "año": "2014",
   21       "imagen": "imagenes/interstellar.png"
   22     }
   23   ]
  }
}
```

- **titulo:** (String) Nombre de la película.
- **sinopsis:** (String) Descripción breve; puede tener varias líneas.
- **generos:** (Array) Lista de géneros.
- **año:** (Integer) Año de estreno.
- **imagen:** (String) Nombre del archivo PNG asociado, ubicado en la misma carpeta.

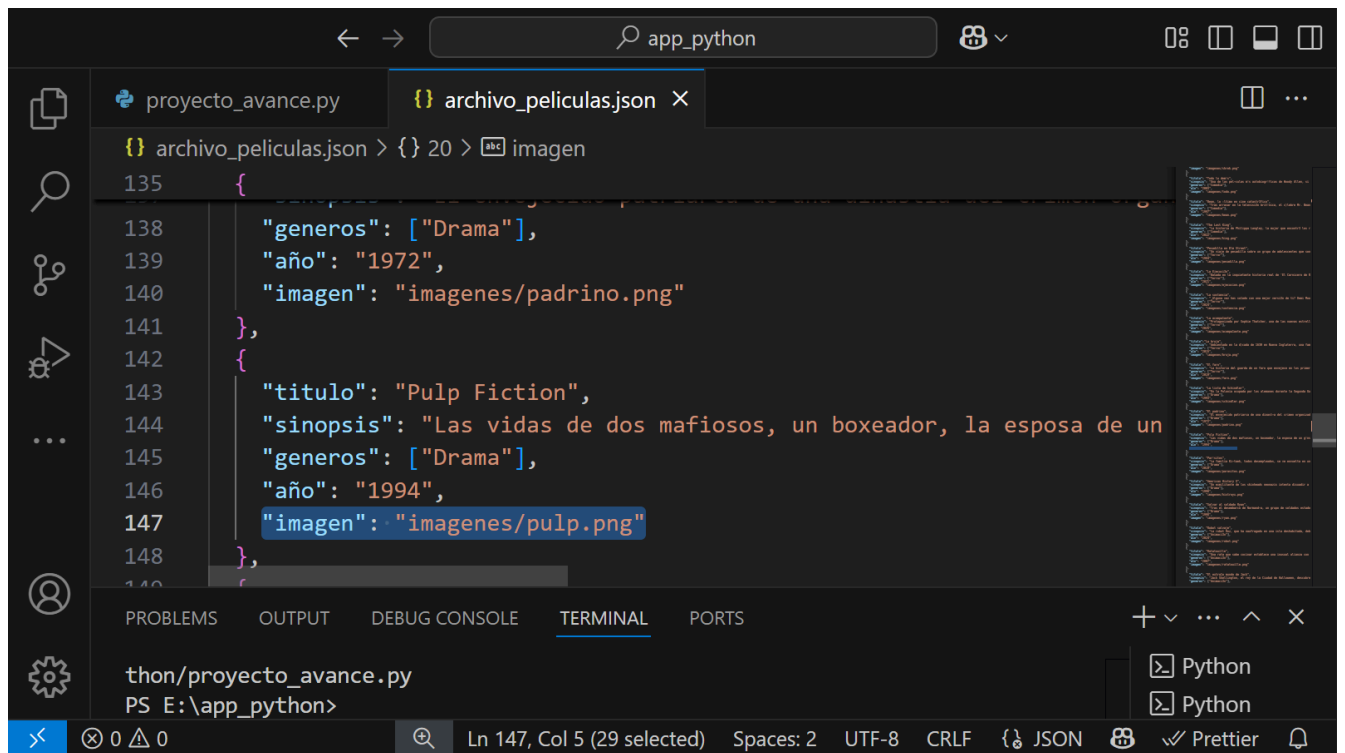
También es necesario agregar una carpeta que contenga las imágenes en formato png.



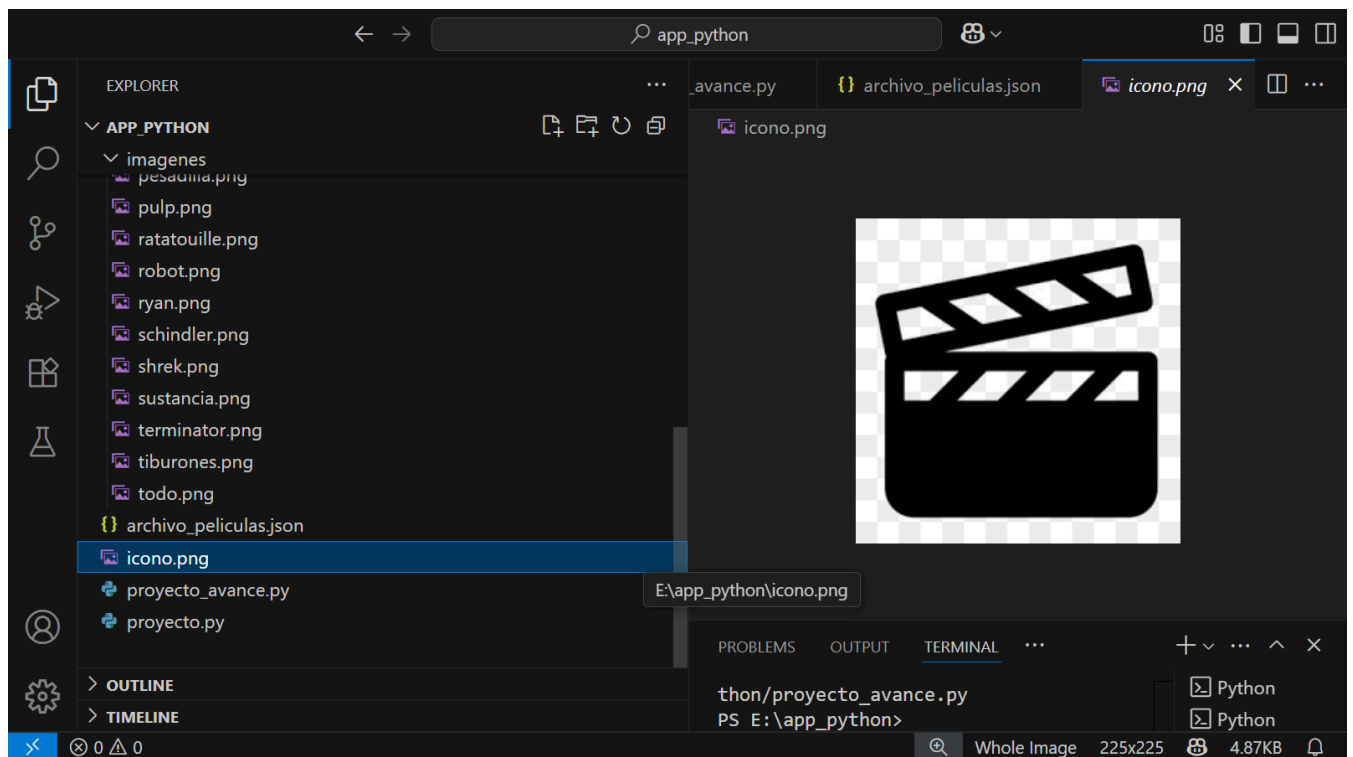
Dentro de la carpeta



En el archivo .json se tiene que agregar correctamente la dirección de la imagen que corresponde a cada película (la carpeta ya cuenta con las direcciones correctas pero si se desea agregar más películas es necesario colocar la dirección de cada imagen).

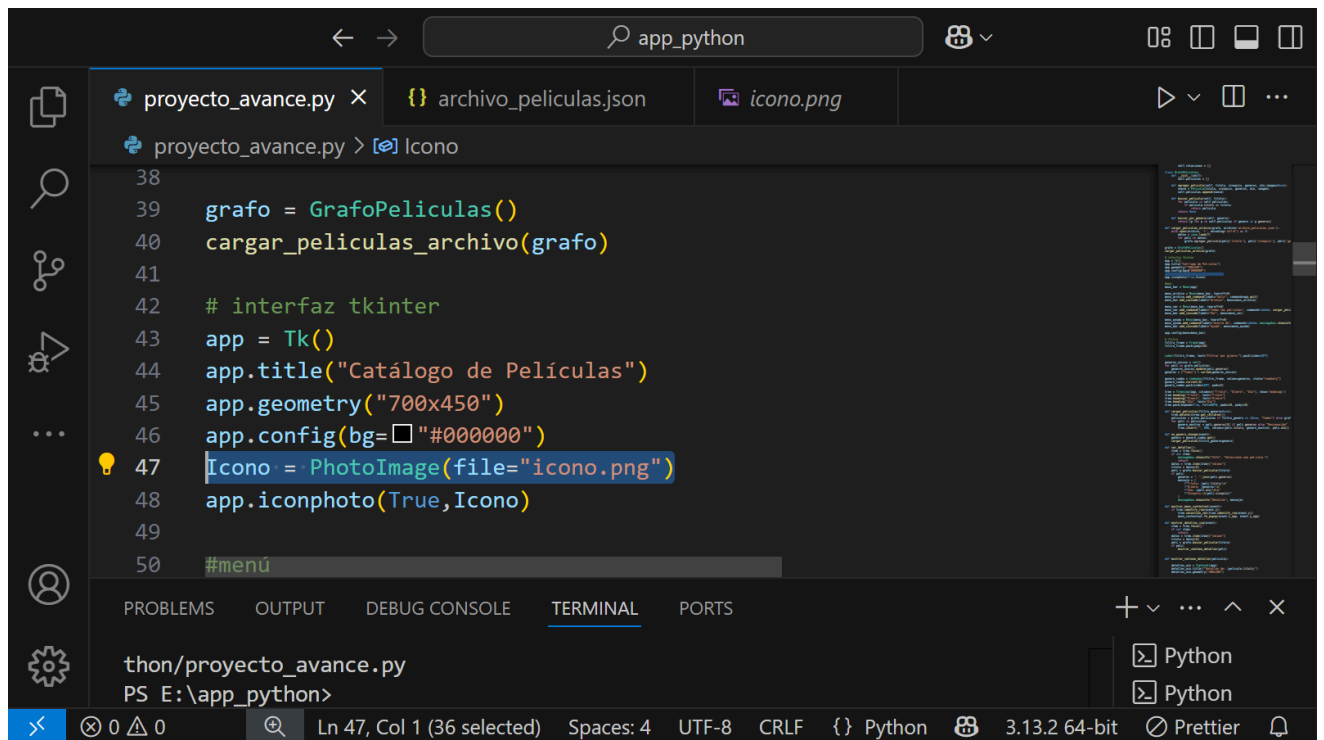


Al igual que se necesita de una imagen png para el icono.



Al igual que para la imagen de cada película este ícono ya está agregado en la carpeta pero si deseas cambiarlo recuerda el nombre que utilices para tenerlo en el código.

Este se encuentra en la interfaz tkinter.



```
38
39 grafo = GrafoPelículas()
40 cargar_peliculas_archivo(grafo)
41
42 # interfaz tkinter
43 app = Tk()
44 app.title("Catálogo de Películas")
45 app.geometry("700x450")
46 app.config(bg="#000000")
47 Icono = PhotoImage(file="icono.png")
48 app.iconphoto(True, Icono)
49
50 #menú
```

thon/proyecto_avance.py
PS E:\app_python>

Ln 47, Col 1 (36 selected) Spaces: 4 UTF-8 CRLF {} Python 3.13.2 64-bit Prettier

Metodología y descripción

El enfoque que utilizamos fue en cascada (waterfall), un desarrollo secuencial y estructurado.

El código funciona de la siguiente manera:

1. Clases principales:
 1. Pelicula: contiene los atributos titulo, sinopsis, generos, año e imagen.
 2. GrafoPelículas: administra la lista de objetos Pelicula y proporciona un método para cargarlas desde un archivo JSON.
2. Interfaz gráfica:
 1. Se crea una ventana principal (Tk) que muestra una lista de botones, uno por cada película.
 2. Cada botón ejecuta una función que abre una nueva ventana emergente (Toplevel) al hacer clic.
3. Función de detalles:
 1. mostrar_ventana_detalle(pelicula): esta función crea una nueva ventana para mostrar los detalles: el título, el año, los géneros, la sinopsis y la imagen PNG asociada (usando Pillow y ImageTk.PhotoImage).
 2. Dentro de esa ventana se colocan widgets Label para mostrar el contenido textual, y un Label con la imagen redimensionada.
4. Flujo de ejecución:
 1. Se instancia un objeto **GrafoPelículas** y se cargan las películas desde **películas.json**.
 2. Se genera un botón por cada película en la ventana principal.
 3. Al hacer clic en un botón, se llama a **mostrar_ventana_detalle**, que abre una nueva ventana con la información detallada.

Para la correcta ejecución del programa es necesario contar con el archivo .json y el código en la misma carpeta.

Al ejecutarse mostrará la lista de películas a la izquierda y al hacer clic en alguna abrirá una ventana nueva que la muestre.

La carpeta también cuenta con un archivo llamado `proyecto.py`, el cual no es más que la estructura de grafo que se utilizó para la elaboración del programa. También puedes acceder a este.

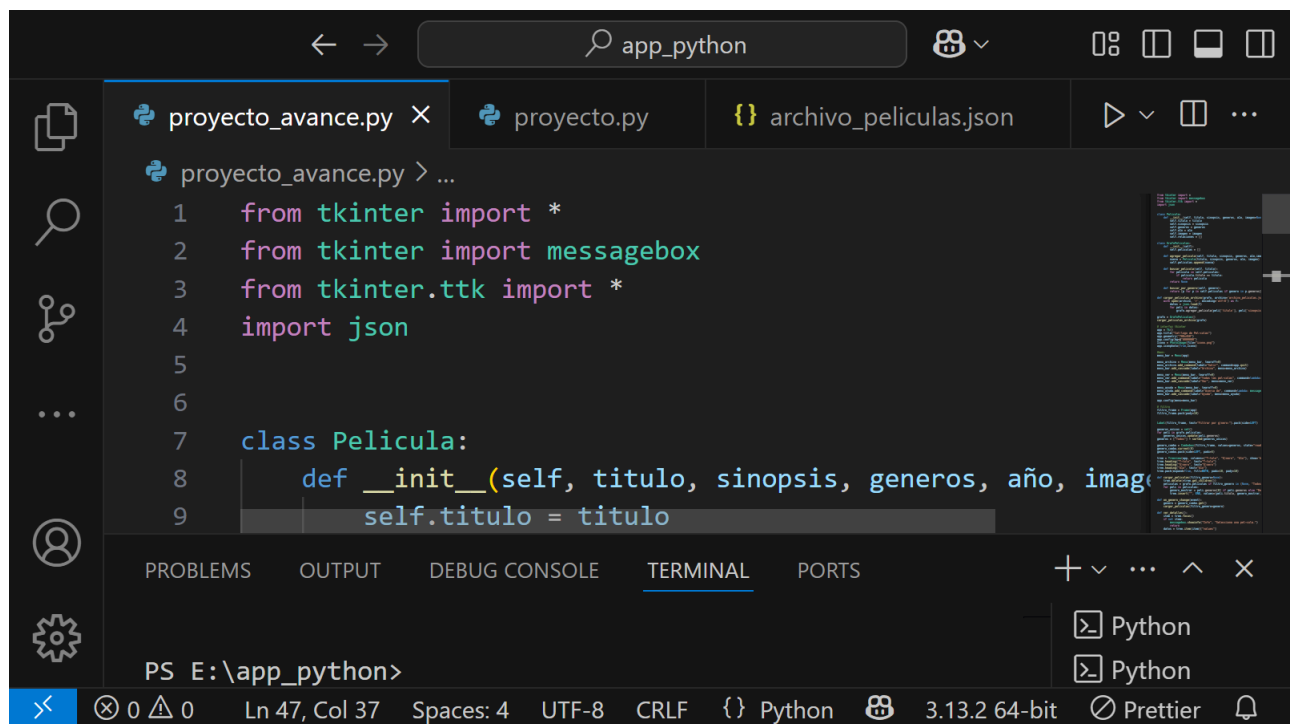
Bibliotecas

En este manual hemos utilizado palabras como `tkinter`, `ventana tk` o `archivo .json`, para que sea posible la ejecución del código se necesitan de algunas librerías, las cuales son las siguientes:

tkinter: para la creación de la interfaz gráfica (incluida en Python por defecto).

json: para manejar la carga de datos desde el archivo `peliculas.json`.

Se estructuraron de la siguiente forma.



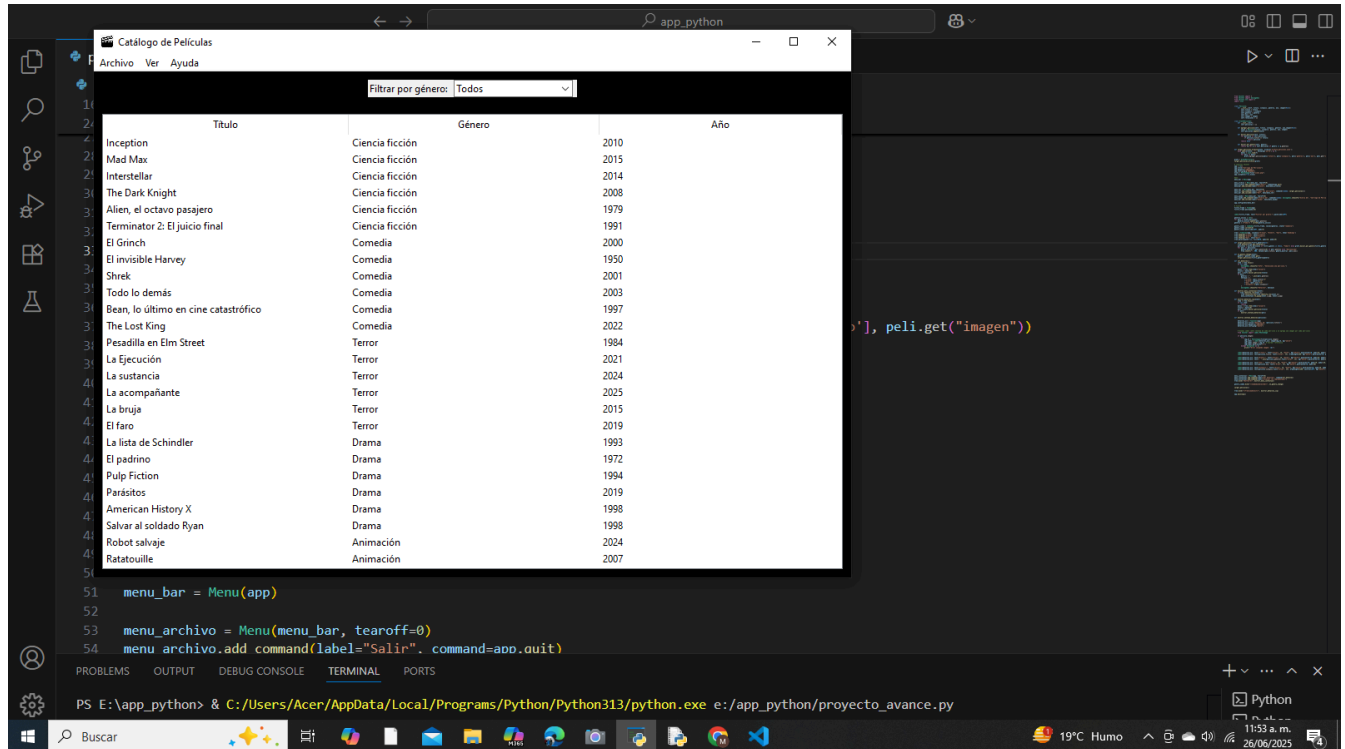
The screenshot shows a code editor with a dark theme. The top bar displays the search path `app_python`. The editor has three tabs: `proyecto_avance.py` (active), `proyecto.py`, and `archivo_peliculas.json`. The active tab shows the following Python code:

```
1 from tkinter import *
2 from tkinter import messagebox
3 from tkinter.ttk import *
4 import json
5
6
7 class Pelicula:
8     def __init__(self, titulo, sinopsis, generos, año, imagen):
9         self.titulo = titulo
```

The bottom of the editor shows the `TERMINAL` tab with the prompt `PS E:\app_python>`. The status bar at the bottom indicates the current position is `Ln 47, Col 37`, with settings for `Spaces: 4`, `UTF-8`, `CRLF`, `{ } Python`, `3.13.2 64-bit`, and `Prettier`.

Ejecución

Sí se siguieron correctamente las instrucciones y los archivos cuentan con cada especificación, entonces debe de aparecer lo siguiente en pantalla.



Nota: Como se puede ver en las imágenes nosotros utilizamos un interprete de código llamado Visual Studio Code, pero este no es necesario para la ejecución del programa.