Elaborado por: Ingeniero Nazir Rosas Diplomado Python Sabatino.

🧾 Proyecto Integrador – Diplomado en Python

🎵 "PyMusic – Tu Biblioteca Musical en Consola"

🎯 Objetivo del Proyecto

Desarrollar una aplicación de **consola en Python** que simule una versión simplificada de Spotify, permitiendo al usuario:

- Gestionar una biblioteca de canciones
- Crear y consultar playlists
- Buscar, filtrar y simular reproducción de música
  Este proyecto integra estructuras de datos, funciones, control de flujo y manejo de archivos.

📦 ¿Qué deben entregar?

1. Código `.py` o Google Colab
2. Archivo `.json` o `.txt` con la base de datos de canciones
3. Capturas de pantalla de la ejecución
4. Documento en PDF explicando cómo funciona el programa (o README si usan GitHub)

🛠️ Requisitos Técnicos

✅ Requerimientos mínimos (60%)

- Menú interactivo desde consola
- Agregar canciones (título, artista, género, duración)
- Almacenar datos en listas de diccionarios
- Buscar por título o artista
- Filtrar por género

Elaborado por: Ingeniero Nazir Rosas Diplomado Python Sabatino.

- Crear playlists (listas dentro de diccionarios)
- Simular reproducción con `time.sleep()` o barra de texto
- Guardar y cargar datos desde archivo `.json` o `.txt`

✨ **Extras opcionales (20%)**

- Contador de reproducciones por canción
- Ranking top 3 más escuchadas
- Likes/dislikes
- Eliminar o editar canciones
- Carga automática al iniciar
- Guardar playlists por usuario

🧩 Estructuras de datos que deben utilizar

- `Listas`: para el catálogo general y playlists
- `Diccionarios`: para representar canciones y organizar playlists
- `Sets`: para evitar géneros duplicados o búsqueda rápida
- `Archivos`: lectura y escritura en `.json` o `.txt`
- `Funciones`: para modularizar el código
- (Opcional) `Tuplas` para listas inmutables de opciones

💻 Importante

- Todo debe ejecutarse desde **la consola** (terminal)
- Deben de clonar el codigo base desde este repositorio https://github.com/Barto12/PyMusic/tree/main

Elaborado por: Ingeniero Nazir Rosas Diplomado Python Sabatino.

Guía para Alumnos – Proyecto PyMusic (Git desde consola)

Paso 1. Clona el repositorio del profesor

git clone https://github.com/Barto12/PyMusic.git

cd PyMusic

git checkout develop

git checkout -b rama-tu-nombre

Ejemplo:

git checkout -b rama-maria

Paso 4. Haz tus cambios en los archivos `.py`, `.json`, etc.

Luego guarda y confirma los cambios:

git add .

git commit -m "feat: agregué función de reproducción"

Paso 5. Sube tu rama a GitHub (solo la primera vez con `-u`)

git push -u origin rama-tu-nombre

Paso 6. Si el profesor hace cambios en `develop`, actualiza tu rama

Asegúrate de estar en tu rama:

git checkout rama-tu-nombre

Luego trae los últimos cambios de `develop`:

git pull origin develop

Esto mezcla los nuevos cambios del profesor con tu rama sin necesidad de `merge`.

Paso 7. Sigue trabajando y subiendo tus avances

Cada vez que avances:

git add .

git commit -m "avance del proyecto"

git push

Ejemplo completo en consola:

```
git clone https://github.com/Barto12/PyMusic.git
cd PyMusic
git checkout develop
git checkout -b rama-juan
# (Haces tus cambios)
git add .
git commit -m "avance 1"
git push -u origin rama-juan

# (Después, si hay cambios nuevos en develop)
git checkout rama-juan
git pull origin develop
git push
```

- **No se permite uso de interfaces gráficas** ni librerías externas como Tkinter, Pygame o similares
- Todo debe construirse con Python puro

📅 Fechas clave

- **Entrega final:** Viernes 21 de junio de 2025
- **Presentación del proyecto:** Durante la última clase

🧑‍⚖️ Rúbrica de Evaluación

Criterio	Puntos
Requerimientos mínimos funcionales	30
Uso correcto de estructuras de datos	20

Elaborado por: Ingeniero Nazir Rosas Diplomado Python Sabatino.

Código modular, funciones claras y organizadas	15
Lectura/escritura de archivos	10
Flujo del programa (interacción, validación, menú)	10
Creatividad / funcionalidad extra	10
Presentación del proyecto (capturas, explicación)	5
Total	**100**