1. Estructura del Proyecto

1.1. Clases Principales

- `Usuario` : Maneja el registro e inicio de sesión.
- `JuegoRetro` : Representa un videojuego con atributos (`Nombre` , `Plataforma` , `Año`).
- `ColeccionJuegos` : Administra la lista de juegos del usuario.
- `APIHandler` : Conecta con la API de **Retro.gg** para obtener datos de los juegos.
- `MuseoRetro` : Implementa una interfaz visual estilo arcade en consola.

---

2. Aplicación de Conceptos del Trabajo Final

Concepto	Aplicación en el Proyecto
POO (Clases y Métodos)	Cada entidad será una clase con métodos específicos (ej. `AgregarJuego()` , `BuscarJuego()`).
Herencia, Agregación, Composición	`JuegoRetro` puede tener agregación con otras clases (`DetallesJuego`). `Usuario` contiene `ColeccionJuegos` por composición.
Encapsulamiento	Todos los atributos serán privados o protegidos, con acceso mediante propiedades.
Listas	`ColeccionJuegos` usará una `List<JuegoRetro>` .
Interfaces	Se definirá una interfaz `IInterfazConsola` para gestionar la visualización arcade del museo.
Enumeraciones (enum)	Se usará un `enum Plataforma` (`NES` , `SNES` , `PS1` , etc.).
Serialización en JSON	Se guardarán los datos de la colección en archivos JSON.
Conexión a API externa	`APIHandler` consumirá datos de **Retro.gg**.
Diagrama UML	Se diseñará para visualizar las relaciones entre clases.
Menú interactivo	Implementación por consola con opciones claras.

---

3. Funcionalidades del Menú

- **Registro de usuario** (`Crear cuenta`, `Iniciar sesión`)
- **Gestión de colección** (`Agregar juego`, `Eliminar juego`, `Ver colección`)
- **Búsqueda automática** (`Consultar API de Retro.gg para obtener información`)
- **Exploración en Museo Retro** (Vista arcade en consola)
- **Guardar/Leer datos** (`Persistencia en JSON`)

---

4. Pasos de Desarrollo

1. **Definir el modelo de clases** (`Usuario`, `JuegoRetro`, `ColeccionJuegos`, etc.).
2. **Implementar el sistema de persistencia** (JSON).
3. **Conectar con la API de Retro.gg** (`APIHandler`).
4. **Crear la visualización del "Museo Retro"** (diseño ASCII, colores en consola).
5. **Estructurar el menú interactivo y pruebas de usuario**.

Diagrama UML - Esquema

- `Usuario` (Clase)
  - **Atributos:** `Nombre`, `Correo`, `Contraseña`
  - **Métodos:** `Registrarse()`, `IniciarSesion()`
  - **Relacionado con:** `ColeccionJuegos` (Composición: un usuario tiene una colección)
- `ColeccionJuegos` (Clase)
  - **Atributos:** `Lista<JuegoRetro>`
  - **Métodos:** `AgregarJuego()`, `EliminarJuego()`, `BuscarJuego()`, `MostrarColeccion()`
  - **Relacionado con:** `JuegoRetro` (Agregación: la colección contiene múltiples juegos)
- `JuegoRetro` (Clase)
  - **Atributos:** `Nombre`, `Plataforma (enum)`, `Año`
  - **Métodos:** `MostrarDetalles()`
  - **Relacionado con:** `APIHandler` (Composición: puede obtener datos de una API)
- `APIHandler` (Clase)
  - **Métodos:** `BuscarJuegoEnAPI()`
  - **Relacionado con:** `JuegoRetro`
- `MuseoRetro` (Clase - Implementa `IInterfazConsola`)
  - **Métodos:** `MostrarEstiloArcade()`
  - **Relacionada con:** `ColeccionJuegos` (Visualización)
- `IInterfazConsola` (Interfaz)
  - **Métodos:** `MostrarEstiloArcade()`