**Informe del análisis utilizados en el desafío #2.**

**Alejandro Bedoya Zuluaga**

**Jeronimo Herrera Espinosa**

**1.Introducción**

En este desafío, abordaremos principalmente la temática y el modelo de Programación Orientada a Objetos (POO), el propósito de este será el de crear desde cero una aplicación de reproducción de música, en las cuales se tendrán que contemplar varias funciones y clases primordiales para su funcionamiento, y las cuales serán profundizadas más adelante.

Este informe detalla el análisis que decidimos aplicar, las clases junto con sus atributos y métodos podrán ser visualizados en el Diagrama de clases estilo UML simplificado.

**2.Descripción del problema**

En este reto se nos darán unos datos/listas acerca de los artistas que aparecerán en la aplicación, junto con sus respectivos álbumes y las respectivas canciones que salen en estos, a su vez.

Para el funcionamiento de esta aplicación se deberá dar la posibilidad de iniciar sesión como un usuario existente, el cual podrá ser premium o estándar, para estas dos clases hay las siguientes ventajas/condiciones:

**Premium:** los usuarios premium pueden poseer una lista de favoritos personalizada, a su vez, pueden elegir seguir la lista de favoritos de otro usuario, podrán escuchar sus canciones en 390kbpsy podrán escuchar todas sus canciones sin publicidad.

**Estándar:** los usuarios estándar, tendrán que ver un anuncio cada dos canciones, estos mensajes publicitarios podrán ser de tres tipos y los cuales tendrán distinta prioridad, de menor a mayor, los tipos son C, B y AAA, también los usuarios estándar tendrán que escuchar las canciones en 128kbps.

**3.Analisis utilizado**

Para este desafío, se decidieron implementar ciertas clases primordiales, las cuales son:

**Plataforma:** la cual nos servirá como la base del todo y la primera interacción con los usuarios, ya que esta tiene relación de referencia con las clases Usuarios, a su vez en esta clase se encuentran los métodos utilizados para cargar los datos que estarán guardados en archivos externos de texto plano (.txt).

**Usuarios:** En esta clase se encontrará guardados, como lo indican su nombre los usuarios, junto con, en sus atributos, su tipo (premium y estándar), junto con los otros ya mencionados en el diagrama.

**Artista:** Esta clase modela al artista de las canciones registradas en la plataforma, en esta se almacena información relevante sobre el artista tales como el país, edad, id, nombre… y demás información relevante que pueda servir de referencia para la canción.

Esta clase mantiene una relación de uno a muchos con la clase álbum, ya que un artista puede tener muchos álbumes los cuales a su vez contendrán muchas canciones.

**Álbum:** Similar a la clase artista, contine información relevante sobre los álbumes de los artistas, estas a su vez se relacionan de uno a muchos con la clase canción, ya que un álbum puede tener varias canciones.

**Créditos:** Esta es una clase la cual tiene una relación de pertenencia con la clase **canción,** debido a que, como es dicho en el enunciado del desafío cada clase debe contener unos créditos en donde se dan a conocer los productores, músicos, compositores y su código.

**listaFavoritos:** Esta es la clase es debido a la funcionalidad de los usuarios premium, los cuales como se dijo ya, pueden poseer una lista de canciones favoritas con un máximo de hasta 10000 canciones, en esta los usuarios pueden crear playlist de favoritos y seguir a otra lista de favoritos de otro usuario.

**Reproductor:**  La clase Reproductor se encarga de gestionar todas las funciones relacionadas con la reproducción de contenido musical dentro de la plataforma. Entre sus principales responsabilidades se incluyen iniciar, pausar, detener y reanudar la reproducción de canciones, así como avanzar o retroceder entre pistas. Además, puede interactuar con listas de reproducción o listas de favoritos para reproducir secuencias de canciones en orden o de forma aleatoria.  
Esta clase actúa como el componente funcional que controla la experiencia de escucha del usuario, permitiendo la gestión dinámica del flujo de audio y garantizando una interacción fluida entre el usuario, las canciones y las listas disponibles en el sistema.

Para este desafío haremos uso de las librerías externas <string> y <random>, las cuales nos servirán para el funcionamiento del programa, como por ejemplo la reproducción aleatoria de las canciones dentro de una lista.

**4.Conclusión**

Para concluir, en este desafío se adoptó un enfoque basado en la Programación Orientada a Objetos (POO), mediante la creación de diversos módulos y clases que permiten estructurar el programa de forma más clara y organizada. Este enfoque contribuye a mejorar la legibilidad del código y facilita la interacción entre los distintos objetos, promoviendo así un diseño modular, reutilizable y fácil de mantener.

Además, la implementación de las clases y sus relaciones permite representar de manera fiel los componentes principales del sistema, reflejando cómo interactúan entre sí dentro de la plataforma. Gracias a esta estructura, el desarrollo se vuelve más escalable y flexible, facilitando la incorporación de nuevas funcionalidades sin afectar el funcionamiento general del programa.