# Visado 1

## Objetos en Javascript (ES6)

Modele (diseñe e implemente) una aplicación similar a Spotify, que llamaremos UNQfy

En UNQfy existe una gran cantidad de temas musicales (*tracks)* los cuales siempre pertenecen a un álbum. Un álbum tiene un sólo artista como autor pero un artista puede ser autor de múltiples albumes. Cada track tiene asociado uno o más géneros, que son strings. También existen playlists, que son conjuntos de tracks que pueden pertenecer a diferentes álbumes.

En UNQfy, además de las típicas operaciones de alta, baja y modificación de todos estos elementos (tracks, albums, artistas), es posible:

- realizar búsquedas de temas.
- Debe ser posible recuperar todas las canciones que fueron interpretadas por un determinado artista, y
- todas las canciones que se correspondan con un determinado género.

También se desea la opción de autogenerar una Playlist en base a una lista de géneros, es decir, rellenar una playlist con canciones de determinados géneros y con una duración máxima.

Para operar UNQfy vamos a usar, en principio, la línea de comando. Más abajo se explican los detalles pero a grandes rasgos la idea es tener una serie de comandos que permitan alterar e inspeccionar el modelo de objetos de UNQfy.

Nota 1: Todas las entidades, albumes, tracks y artistas deben tener una propiedad ID, la cual no puede repetirse.

Nota 2: Al eliminar un artista, track o album, se deberá chequear que los tracks involucrados que pertenezcan a una playlist sean borrados de la playlist también.

## TAREAS (realice las tareas en el orden indicado)

1. Realice el diseño y documente mediante un diagrama de clases UML. Recomendación: usar UMLet
2. **Valide el diseño con los docentes de la materia**.
3. Implemente utilizando clases con la sintaxis ECMA Script 6.
4. Utilice los test cases provistos para validar parte de su implementación.
5. Utilizando la consola se desea invocar comandos para interactuar con UNQfy. Implemente un programa que acepte como parámetros:
    a. El nombre de un comando a ejecutarse y

b. Los argumentos necesarios para llevar a cabo ese comando.

Por ejemplo el comando que permite dar de alta a un artista se ejecutaría de la siguiente forma.

```
node main.js addArtist name "Peter Tosh"
```

Donde:

- *addArtist* es el comando
- *name* es el nombre de un parámetro
- *"Peter Tosh"* es el valor del parametro name

Un ejemplo más complejo es la creación de un tema, donde hay que enviar más parámetros.

```
node main.js addTrack title "Legalize it" album "Greatest Hits" duration
500
```

Estos son solo ejemplos ud. puede decidir omitir los nombres de los parámetros en cuyo caso la línea de comando tendría la forma:

```
node main.js addTrack "Legalize it" "Greatest Hits" 500
```

Los comandos que deberá implementar deben permitir invocar la siguiente funcionalidad en UNQfy (las tareas con fondo verde son un plus para llegar a la nota 10 en el visado, pero no son obligatorias):

I. Ingresar tracks, álbumes y artistas. Tenga en cuenta que para dar de alta un álbum el artista debe existir, y para agregar un track el álbum al que pertenece debe existir. Si no existen debe reportar en la consola que no se pudo completar la operación, indicando claramente el error.
II. Eliminar Tracks, albumes, artistas y playlists. Tenga en cuenta que cuando elimine un artista, album o track, deberá eliminar los tracks de las playlists que los usen. Esto quiere decir que el modelo de objetos debe mantener su consistencia.
III. Buscar (e imprimir en pantalla) tracks, álbumes o artistas por matching parcial del string recibido como parámetro contra el nombre de los objetos.
IV. Buscar (e imprimir en pantalla) todos los tracks de un determinado artista.
V. Buscar (e imprimir en pantalla) todos los tracks que poseen un género particular.

VI. Crear una Playlist (con un nombre determinado) de una determinada duración máxima, para un género determinado, y pedir que UNQfy la rellene automáticamente.
VII. Listar en pantalla el contenido de una Playlist, Artist, Album, Track
VIII. Modelar a los usuarios de UNQfy. Un usuario por ahora lo único que sabe hacer es escuchar un tema. Es posible preguntarle a un usuario qué temas ha escuchado y retorna una lista sin repeticiones de los temas. Tambíen se le puede consultar cuántas veces escuchó un tema, retornará un entero.
IX. "Escuchar un tema". Esto significa que cada vez que un track es escuchado por un usuario, se mantiene un registro de dicha “escucha”.
X. Armar automáticamente e imprimir en pantalla la lista "This is ..." . Esta lista contiene los 3 temas más escuchados de un artista dado. Tenga en cuenta que esta lista siempre es calculada "on the fly".
XI. Utilizando su implementación de Unqfy:
    a. Lanzar una excepción con un mensaje apropiado cuando se intente agregarse un artista cuando ya existe uno con el mismo nombre.
    b. Lanzar una excepción con un mensaje apropiado cuando se intente agregar un track que ya existe en un album.
    c. Maneje ambas excepciones

## Notas:

1. Considere la existencia de UNQfy como un objeto, deberá aplicar el patrón Facade para simplificar las operaciones y pasar los tests cases.
2. Tenga en cuenta que, para poder “mantener” el estado entre las diferentes ejecuciones de UNQfy, su programa debe seguir la siguiente lógica en la ejecución de cada comando:
    a. Cargar de disco el estado de UNQfy, utilizando la función `getUNQfy` provista por la cátedra.
    b. Ejecutar el comando pasado por parámetro en la línea de comandos.
    c. Guardar el nuevo estado del programa, utilizando la función `saveUNQfy` provista por la cátedra.
3. En la descripción de los diferentes comandos a implementar se ha omitido intencionalmente algunos detalles, como por ejemplo los parámetros para las búsquedas o el formato de la impresión en pantalla. Ud. debe aplicar sentido común maximizando siempre la legibilidad del código y la usabilidad de UNQfy.

# Requerimientos de la entrega:

- El programa debe tener al menos un módulo node JS que implemente la lógica de UNQfy. Por ejemplo, si su módulo se llama `unqfy` será importado así:

```
const unqfy = require('./unqfy.js');
```

- Implementar el manejo via línea de comando en otro archivo JS que use el módulo UNQfy.
- La entrega se deberá realizar subiendo el contenido a github.
- Debe incluir en el readme.md de github una referencia al diagrama de clases, para poder verlo en pantalla.
- En el readme.md de github debe mostrar como se invoca cada comando desde la CLI.
- Incluya un script de linea de comando que permita ejercitar toda la funcionalidad de lo entregado.
- Incluir comandos de prueba.

## Importante:

Para validar ud. mismo su entrega considere que a la hora de corregir los docentes seguirán los siguientes pasos:

1. Clonar su repositorio de GitHub
2. npm install
3. npm test - los tests deben pasar!
4. Leer la documentacion del readme.md de github.
5. Usar los comandos desde la CLI
6. Revisar el código fuente

# Recomendaciones

La implementación de los comandos de UNQfy es lo suficientemente interesante para aplicar técnicas de reuso de código o de encapsulamiento de funcionalidad (como patrones de diseño). Si aplica algún patrón por favor documéntelo (indique qué patrón, y que roles juegan las clases de su modelo).

Debe respetar las buenas práctica de programación orientada a objetos: no repetir código, distribuir de manera adecuada las responsabilidades, no violar encapsulamiento, etc.

# Checkpoint (martes 15/9)

1. Diseño completo UML
2. Implementacion de punta a punta de “addArtists” (debe funcionar desde la linea de comando) y getArtist