

# Sistema de Reorden de Canciones Pendiente\_Lazy

## Resumen

Se ha implementado un sistema completo que permite a los usuarios gestionar el orden de sus propias canciones en estado `pendiente_lazy`. Ahora los usuarios pueden mover sus canciones hacia arriba y hacia abajo antes de que sean aprobadas por el admin, lo que permite una mejor control del flujo de reproducción.

## Diagrama del Flujo de la Cola Lazy



### 3■■ ADMIN APRUEBA CANCIONES (SISTEMA LAZY)

■ Admin Panel → Canciones → Cola Lazy

■ [Botón] Aprobar siguiente canción pendiente\_lazy

■ Sistema TOMA la primera canción pendiente\_lazy

■ y la cambia a: APROBADO

■ ¿IMPORTANTE! ¿Qué canción es "primera"?

- • Se respeta el orden\_manual del usuario
- • La canción que el usuario puso primero se aprueba primero
- • Se mantiene el orden relativo

■ La canción aprobada entra en la COLA PRINCIPAL

■ (se integra al sistema de "Cola Justa")

- • Espera su turno según categoría de mesa
- • ORO: 3 canciones por turno
- • PLATA: 2 canciones por turno
- • BRONCE: 1 canción por turno

#### 4■■ SINCRONIZACIÓN CON COLA JUSTA

■ REPRODUCIÉNDOSE AHORA

■ "Canción Actual" - Usuario X

■ [■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■] 2:45 / 3:45 ■

▼ (cuando termina)

■ SIGUIENTE EN LA COLA

- (Algoritmo Cola Justa respeta orden\_manual)

- 1. Canciones CON orden\_manual (admin las priorizó)
- 2. Luego, Round Robin por mesas
  - - Toma N canciones de cada mesa
  - - Respeta el orden que el usuario estableció

■ Estado: REPRODUCIENDO

■ Player inicia: "Bohemian Rhapsody" - Usuario X

[■■■■■■■■■■■■■■■■■■■■] 0:30 / 6:00

■ La canción que el usuario reordenó está

■ sonando en el momento correcto ■

- Cuando termina → Estado: CANTADA

■ Si es karaoke: Muestra puntuación

■ Se guarda en historial del usuario

## 5 ■ EJEMPLO PRÁCTICO: FLUJO COMPLETO

PASO 1: Usuario "Toscana-Usuariol" agrega 4 canciones

■ Canción 1: Aprobada (Se reproduce en 2 min)

■■ Estado: APROBADO

■ Canción 2: Pendiente Lazy (En cola esperando)

■ Estado: PENDIENTE\_LAZY

```
■ ■ orden_manual: 1
```

■ Canción 3: Pendiente Lazy (En cola esperando)

```
■■ orden_manual: 2
```

■■ Estado: PENDIENTE\_LAZY

```
■■ order_manual: 3
```

[illegible]

Después de reordenar:

■ Canción 2: orden\_manual = 3

[illegible]

= Canción 4

[illegible]

[LUEGO] Canción 2 (Usuario X) - PENDIENTE\_LAZY

[illegible]

■ [LUEGO] Canción 2 (Usuario X) - PENDIENTE\_LAZY

↑ ¡La que el usuario puso en primer lugar!

■ [LUEGO] Canción 2 (Usuario X) - PENDIENTE\_LAZY

- Se guarda en la BD (campo orden\_manual)

manteniendo ese orden relativo

- Si el usuario mueve una canción, todos lo ven

#### 4. SEGURIDAD Y AISLAMIENTO:

- Los parámetros incluyen `usuario_id`
- Una canción solo puede moverse si pertenece al usuario
- Los usuarios de otras mesas no interfieren

## Cambios Realizados

### 1. Backend - Funciones CRUD (crud.py)

Se agregaron dos nuevas funciones para manejar el movimiento de canciones:

```
move_lazy_song_up(db: Session, cancion_id: int, usuario_id: int)
```

- Mueve una canción `pendiente_lazy` hacia arriba en la cola personal del usuario
- Solo funciona para canciones del usuario actual
- Utiliza el campo `orden_manual` para mantener el orden
- Retorna la canción actualizada o `None` si no existe

```
move_lazy_song_down(db: Session, cancion_id: int, usuario_id: int)
```

- Mueve una canción `pendiente_lazy` hacia abajo en la cola personal del usuario
- Solo funciona para canciones del usuario actual
- Mantiene la sincronización con el algoritmo de cola justa
- Retorna la canción actualizada o `None` si no existe

### 2. Backend - Endpoints API (canciones.py)

Se agregaron dos nuevos endpoints POST para las acciones de mover:

```
POST /api/v1/canciones/{cancion_id}/mover-arriba
```

- Parámetros: `cancion_id` (path), `usuario_id` (query)
- Respuesta: Objeto `Cancion` actualizado
- Acción: Mueve canción hacia arriba en la cola lazy del usuario

```
POST /api/v1/canciones/{cancion_id}/mover-abajo
```

- Parámetros: `cancion_id` (path), `usuario_id` (query)
- Respuesta: Objeto `Cancion` actualizado
- Acción: Mueve canción hacia abajo en la cola lazy del usuario

Ambos endpoints: - Notifican a los clientes via WebSocket cuando hay cambios - Retornan 404 si la canción no existe o no pertenece al usuario - Solo funcionan con canciones en estado `pendiente_lazy`

### 3. Frontend - Interfaz de Usuario (static/app\_bees.js)

#### Actualización de `createSongItemHTML(song, isMyList)`

- Ahora genera botones de flecha hacia arriba (■) y abajo (■) para canciones en estado `pendiente_lazy`
- Los botones solo aparecen en "Mis Canciones" y solo para canciones `pendiente_lazy`
- Los botones están contenidos en un div `.song-move-buttons`

#### Nuevos Handlers de Eventos

- `handleMoveSongUp(event)`: Maneja clicks en botón "■"
- `handleMoveSongDown(event)`: Maneja clicks en botón "■"
- Ambos realizan llamadas AJAX a los nuevos endpoints
- Muestran notificaciones de éxito/error al usuario
- Actualizan la lista de canciones tras completar la acción

#### Event Listeners (DOMContentLoaded)

- Se agregaron listeners para los eventos click en los botones de mover
- Asociados al elemento `#my-song-list`

### 4. Frontend - Estilos CSS (static/styles\_bees.css)

#### Nuevos Estilos para Botones de Mover

```
.song-move-buttons { }  
.move-up-btn { }  
.move-down-btn { }
```

Características: - Botones azules (#2196F3) con contraste blanco - Animación de escala al pasar el mouse - Estado deshabilitado con opacidad 0.6 - Gap de 8px entre botones

#### Estilo para Estado Pendiente\_Lazy

```
.status-pendiente_lazy {  
  background: #FF9800; /* Naranja */  
  color: blanco;  
  animation: pulse 2s infinite;  
}
```

El estado `pendiente_lazy` ahora se ve como **naranja** con animación de pulso para destacar visualmente que está en cola de espera.

## Flujo de Funcionamiento

1. **Usuario agrega canción:** Si ya hay una canción aprobada, va a `pendiente_lazy`
2. **En "Mis Canciones":** Las canciones en `pendiente_lazy` muestran botones ■ ■ ■

3. **Usuario mueve canción:** El orden se actualiza en la BD usando `orden_manual`
4. **Sincronización automática:** El algoritmo de cola justa respeta el orden manual del usuario
5. **Admin aprueba:** Cuando el admin aprueba una canción lazy, su posición se mantiene respecto a sus hermanas

## Integración con Cola Justa

Las funciones de mover aprovechan el sistema existente de `orden_manual`: - El campo `orden_manual` ya está soportado en `get_cola_completa_con_lazy()`  
- Las canciones del mismo usuario mantienen su orden relativo cuando se aprueban - El algoritmo de mesa (ORO/PLATA/BRONCE) continúa funcionando normalmente

## Pruebas Recomendadas

1. Agregar múltiples canciones desde el mismo usuario → deben ir a `pendiente_lazy`
2. Mover una canción hacia arriba → debe cambiar su posición
3. Mover una canción hacia abajo → debe cambiar su posición
4. Mover varias veces la misma canción → debe mantener coherencia
5. Actualizar la página → el orden debe persistir
6. Admin aprueba una canción → debe mantener su posición relativa respecto a hermanas
7. Diferentes usuarios mueven canciones → no deben interferir entre sí

## Archivos Modificados

- ■ `crud.py` - Funciones de movimiento
- ■ `canciones.py` - Nuevos endpoints
- ■ `static/app_be.es.js` - Handlers y renderizado
- ■ `static/styles_be.es.css` - Estilos de botones y estado

## Notas Importantes

- El movimiento solo afecta a las canciones `pendiente_lazy` del usuario actual
- Los cambios se sincronizan automáticamente vía WebSocket
- El campo `orden_manual` utiliza decimales (0.5, 1.5, etc.) para permitir inserciones entre canciones
- La animación de pulso en status `pendiente_lazy` ayuda a que el usuario note que está en cola de espera