



CARRERA DE INGENIERÍA EN SOFTWARE



MODELADO AGIL DE SOFTWARE

Autor:

Arteaga Molina Cesar Emanuel

Menoscal Santana Bryan Steven

Muñiz Rivas Leopoldo Miquel

Delgado Solorzano Jeremy Josué

Zambrano Lucas Justin Alejandro

Zambrano Pino Pedro Alexandre

Tema:

Primer entregable

Docente:

Ing. Israel Gómez

Curso:

5to semestre “B”

Periodo 2025(1)



ÍNDICE

1. EPICAS.....	1
1.1. Épica 1: Sistema de Autenticación y Perfiles de Usuario	1
1.1.1 Historia de Usuario: HU01 - Registro de Usuario.....	1
1.2.1 Historia de Usuario: HU02 - Inicio de Sesión	2
1.3.1 Historia de Usuario: HU03 - Recuperación de Contraseña	4
1.4.1 Historia de Usuario: HU04 - Perfil de Usuario	5
1.2. Épica 2: Reproductor de Audio con Playlists Básicas	7
1.1.2 Historia de Usuario: HU05 - Reproducción de Canciones	7
1.2.2 Historia de Usuario: HU06 - Creación de Playlists	8
1.3.2 Historia de Usuario: HU07 - Controles de Reproducción.....	10
1.3. Épica 3: Catálogo de Música Organizado.....	12
1.1.3 Historia de Usuario: HU08 - Categorización por Géneros.....	12
1.2.3 Historia de Usuario: HU09 - Búsqueda de Música	13
1.3.3 Historia de Usuario: HU10 - Visualización de Álbumes y Singles	15
1.4. Épica 4: Sistema de Suscripciones.....	16
1.1.4 Historia de Usuario: HU11 - Plan Gratuito.....	16
1.2.4 Historia de Usuario: HU12 - Suscripción Premium Mensual.....	18
1.5. Épica 5: Almacenamiento y Streaming.....	19
1.1.5 Historia de Usuario: HU13 - Solución de Almacenamiento	19
1.2.5 Historia de Usuario: HU14 - Streaming de Audio	21
1.6. Épica 6: Pagos.....	23
1.1.6 Historia de Usuario: HU15 - Pago de Suscripción	23
1.2.6 Historia de Usuario: HU16 - Gestión de Suscripciones	24
1.7. Épica 7: Modo Offline	26
1.1.7 Historia de Usuario: HU17 - Descarga de Canciones	26
1.8. Épica 8: Dashboard de Administración.....	27
1.1.8 Historia de Usuario: HU18 - Gestión de Usuarios	27
1.2.8 Historia de Usuario: HU19 - Gestión de Contenido Musical	29
1.3.8 Historia de Usuario: HU20 - Estadísticas de Uso.....	30
1.9. Épica 9: Plan Premium Anual	32
1.1.9 Historia de Usuario: HU21 - Suscripción Anual.....	32
1.10. Épica 10: Recomendaciones Básicas	34
1.1.10 Historia de Usuario: HU22 - Sugerencias Personalizadas	34
1.2.10 Historia de Usuario: HU23 - Playlists Populares	35



1.11.	Épica 11: Búsqueda Avanzada	37
1.1.11	Historia de Usuario: HU24 - Filtros Avanzados	37
1.2.11	Historia de Usuario: HU25 - Resultados Categorizados	38
1.12.	Resumen del Backlog	40
1.1.12	MVP (Must Have) - 187 puntos	40
1.2.12	Fase 2 (Should Have) - 42 puntos	40
1.3.12	Stack Tecnológico	40
1.4.12	Estimación Total	41
2.	Roadmap del proyecto	41
3.	Estimaciones del proyecto	42
4.	Priorización MoSCoW	46
5.	Sprint Planning, Daily Standups, Sprint Review y Retrospectives	46
6.	Burndown charts	47
7.	Otros gráficos	48
8.	Metodologías aplicadas	49
8.1.	Scrum	49

1. EPICAS

1.1.Épica 1: Sistema de Autenticación y Perfiles de Usuario

1.1.1 *Historia de Usuario: HU01 - Registro de Usuario*

Como usuario

Quiero poder registrarme en la plataforma

Para crear una cuenta personal

Criterios de Aceptación

Backend

- El sistema debe permitir la creación de cuentas con nombre, email y contraseña
- El email debe ser único en la base de datos
- La contraseña debe almacenarse de forma segura usando hash
- Debe enviarse confirmación por email
- Al registrarse correctamente, debe devolverse un mensaje de éxito
- En caso de error, se debe retornar un mensaje de error claro

Frontend

- Debe haber un formulario de registro con validación de campos
- Debe mostrar confirmación visual de registro exitoso
- El formulario debe validar formato de email y longitud de contraseña
- Debe incluir navegación hacia el login

Tasklist - Backend



- Crear modelo Usuario en Django
- Crear endpoint POST /api/auth/register
- Implementar validación de datos
- Configurar envío de emails de confirmación
- Hashear contraseñas con bcrypt
- Escribir tests unitarios

Tasklist - Frontend

- Crear componente RegisterComponent
- Implementar formulario reactivo con validaciones
- Conectar con API de registro
- Mostrar mensajes de éxito/error
- Agregar navegación al login

Estimación: 8 puntos | **Prioridad:** Must Have

1.2.1 Historia de Usuario: HU02 - Inicio de Sesión

Como usuario registrado

Quiero poder iniciar sesión

Para acceder a mi cuenta

Criterios de Aceptación

Backend

- El sistema debe autenticar usuarios con email y contraseña
- Debe generar y gestionar tokens de sesión/JWT



- Debe validar credenciales contra la base de datos
- Debe manejar sesiones de forma segura
- Debe retornar información del usuario autenticado

Frontend

- Debe tener un formulario de login con validación
- Debe incluir opción "recordarme" para mantener sesión
- Debe redirigir al dashboard después del login exitoso
- Debe mostrar errores de autenticación claramente
- Debe guardar token de sesión de forma segura

Tasklist - Backend

- [Crear endpoint POST /api/auth/login
- Implementar autenticación con JWT
- Validar credenciales de usuario
- Configurar manejo de sesiones
- Implementar refresh tokens
- Escribir tests de autenticación

Tasklist - Frontend

- Crear componente LoginComponent
- Implementar formulario con email y contraseña
- Agregar checkbox "recordarme"
- Gestionar tokens en localStorage/sessionStorage



- Implementar guards de autenticación
- Crear interceptor para añadir tokens a requests

Estimación: 5 puntos | **Prioridad:** Must Have

1.3.1 Historia de Usuario: HU03 - Recuperación de Contraseña

Como usuario

Quiero poder recuperar mi contraseña

Para acceder a mi cuenta si la olvido

Criterios de Aceptación

Backend

- El sistema debe generar tokens de recuperación únicos y temporales
- Debe enviar emails con enlace de recuperación
- Debe validar tokens de recuperación
- Debe permitir el restablecimiento seguro de contraseña
- Los tokens deben expirar después de un tiempo determinado

Frontend

- Debe tener interfaz para iniciar el proceso de recuperación
- Debe mostrar formulario para ingresar email
- Debe tener página para restablecer contraseña con token
- Debe confirmar visualmente el envío del email
- Debe validar la nueva contraseña antes de enviarla

Tasklist - Backend



- Crear endpoint POST /api/auth/forgot-password
- Crear endpoint POST /api/auth/reset-password
- Generar tokens de recuperación seguros
- Configurar plantillas de email
- Implementar expiración de tokens
- Validar y actualizar contraseñas

Tasklist - Frontend

- Crear componente ForgotPasswordComponent
- Crear componente ResetPasswordComponent
- Implementar formulario de solicitud de recuperación
- Implementar formulario de nueva contraseña
- Manejar parámetros de URL para tokens
- Mostrar mensajes de confirmación

Estimación: 5 puntos | **Prioridad:** Must Have

1.4.1 Historia de Usuario: HU04 - Perfil de Usuario

Como usuario

Quiero tener un perfil básico

Para personalizar mi información

Criterios de Aceptación

Backend

- El sistema debe almacenar información básica del perfil



- Debe permitir actualización de datos personales
- Debe validar datos antes de guardarlos
- Debe mantener historial de cambios importantes
- Debe proteger la información sensible

Frontend

- Debe mostrar campos para información personal básica
- Debe tener opción para editar perfil
- Debe validar campos antes del envío
- Debe mostrar confirmación de cambios guardados
- Debe permitir cambio de avatar/foto

Tasklist - Backend

- Extender modelo Usuario con campos de perfil
- Crear endpoint GET /api/user/profile
- Crear endpoint PUT /api/user/profile
- Implementar validaciones de perfil
- Añadir soporte para avatars
- Crear tests de actualización de perfil

Tasklist - Frontend

- Crear componente ProfileComponent
- Implementar formulario de edición de perfil
- Mostrar información actual del usuario

- Agregar subida de avatar
- Implementar validaciones del lado cliente
- Crear navegación en el menú principal

Estimación: 5 puntos | **Prioridad:** Must Have

1.2. Épica 2: Reproductor de Audio con Playlists Básicas

1.1.2 Historia de Usuario: HU05 - Reproducción de Canciones

Como usuario

Quiero poder reproducir canciones individuales

Para disfrutar de la música

Criterios de Aceptación

Backend

- El sistema debe servir archivos de audio de forma eficiente
- Debe implementar streaming de audio sin interrupciones
- Debe manejar diferentes formatos de audio
- Debe controlar acceso según tipo de suscripción
- Debe registrar reproducciones para estadísticas

Frontend

- Debe tener interfaz de reproductor funcional
- Debe mostrar información de la canción actual
- Debe manejar carga y buffering de audio



- Debe sincronizar estado entre componentes
- Debe funcionar en diferentes navegadores

Tasklist - Backend

- Configurar Supabase Storage para archivos de audio
- Crear endpoint GET /api/songs/{id}/stream
- Implementar control de acceso por suscripción
- Optimizar delivery de archivos de audio
- Registrar reproducciones en base de datos
- Manejar metadatos de canciones

Tasklist - Frontend

- Crear servicio AudioPlayerService
- Crear componente AudioPlayerComponent
- Implementar controles de reproducción básicos
- Manejar estados de carga y error
- Sincronizar con estado global de la app
- Optimizar para rendimiento de audio

Estimación: 13 puntos | **Prioridad:** Must Have

1.2.2 Historia de Usuario: HU06 - Creación de Playlists

Como usuario

Quiero crear playlists personales

Para organizar mi música favorita



Criterios de Aceptación

Backend

- El sistema debe permitir crear, editar y eliminar playlists
- Debe gestionar relaciones entre playlists y canciones
- Debe permitir reordenar canciones en playlists
- Debe validar permisos de acceso a playlists
- Debe mantener integridad de datos al eliminar canciones

Frontend

- Debe tener interfaz para crear, editar y eliminar playlists
- Debe permitir agregar/quitar canciones de playlists
- Debe mostrar lista de playlists del usuario
- Debe permitir reordenar canciones con drag & drop
- Debe confirmar acciones destructivas

Tasklist - Backend

- Crear modelos Playlist y PlaylistSong
- Crear endpoints CRUD para playlists
- Implementar endpoint para gestionar canciones en playlist
- Añadir validaciones de permisos
- Crear endpoint para reordenar canciones
- Escribir tests para operaciones de playlist

Tasklist - Frontend



- Crear componente PlaylistManagerComponent
- Crear componente CreatePlaylistComponent
- Implementar lista de playlists del usuario
- Agregar funcionalidad drag & drop
- Crear modales de confirmación
- Integrar con el reproductor de audio

Estimación: 8 puntos | **Prioridad:** Must Have

1.3.2 Historia de Usuario: HU07 - Controles de Reproducción

Como usuario

Quiero tener controles básicos de reproducción

Para controlar mi experiencia de escucha

Criterios de Aceptación

Backend

- El sistema debe mantener estado de reproducción del usuario
- Debe sincronizar posición de reproducción entre sesiones
- Debe manejar cola de reproducción
- Debe gestionar modo repetición y aleatorio
- Debe optimizar la carga de la siguiente canción

Frontend

- Debe mostrar botones de play, pause, skip (anterior/siguiente)
- Debe tener control de volumen funcional



- Debe mostrar barra de progreso interactiva
- Debe mostrar tiempo transcurrido y total
- Debe persistir configuraciones entre sesiones

Tasklist - Backend

- Crear modelo UserPlaybackState
- Crear endpoints para gestionar estado de reproducción
- Implementar cola de reproducción
- Añadir soporte para modos repeat/shuffle
- Optimizar pre-carga de canciones
- Sincronizar estado entre dispositivos

Tasklist - Frontend

- Extender AudioPlayerComponent con todos los controles
- Implementar barra de progreso interactiva
- Crear control de volumen con slider
- Añadir botones de skip y modo repeat/shuffle
- Persistir configuraciones en localStorage
- Añadir atajos de teclado

Estimación: 5 puntos | **Prioridad:** Must Have

1.3. Épica 3: Catálogo de Música Organizado

1.1.3 Historia de Usuario: HU08 - Categorización por Géneros

Como usuario

Quiero ver la música categorizada por géneros

Para encontrar fácilmente lo que me gusta

Criterios de Aceptación

Backend

- El sistema debe tener un catálogo de géneros musicales
- Debe categorizar canciones por géneros principales
- Debe permitir búsqueda y filtrado por género
- Debe mantener consistencia en la categorización
- Debe optimizar consultas de género

Frontend

- Debe mostrar interfaz de navegación intuitiva entre categorías
- Debe listar géneros disponibles
- Debe mostrar canciones/álbumes por género
- Debe permitir navegación rápida entre géneros
- Debe mostrar estadísticas por género

Tasklist - Backend

- Crear modelo Genre
- Relacionar canciones con géneros



- Crear endpoint GET /api/genres
- Crear endpoint GET /api/genres/{id}/songs
- Implementar filtros por género en búsqueda
- Optimizar queries con índices

Tasklist - Frontend

- Crear componente GenresComponent
- Crear componente GenreDetailComponent
- Implementar navegación por géneros
- Mostrar grid de géneros con imágenes
- Integrar con búsqueda global
- Añadir breadcrumbs de navegación

Estimación: 8 puntos | **Prioridad:** Must Have

1.2.3 Historia de Usuario: HU09 - Búsqueda de Música

Como usuario

Quiero poder buscar artistas y canciones

Para encontrar música específica

Criterios de Aceptación

Backend

- El sistema debe implementar búsqueda de texto completo
- Debe buscar en artistas, canciones, álbumes y playlists
- Debe retornar resultados relevantes ordenados por relevancia



- Debe optimizar velocidad de búsqueda
- Debe manejar búsquedas con errores tipográficos

Frontend

- Debe tener barra de búsqueda funcional y accesible
- Debe mostrar sugerencias mientras el usuario escribe
- Debe categorizar resultados (artistas, canciones, álbumes)
- Debe mostrar resultados de forma clara y navegable
- Debe mantener historial de búsquedas

Tasklist - Backend

- Implementar búsqueda con PostgreSQL Full-Text Search
- Crear endpoint GET /api/search
- Optimizar índices de búsqueda
- Implementar paginación de resultados
- Añadir filtros de búsqueda
- Implementar auto-completado

Tasklist - Frontend

- Crear componente SearchComponent
- Implementar barra de búsqueda con autocomplete
- Crear página de resultados SearchResultsComponent
- Mostrar resultados categorizados
- Implementar filtros de búsqueda

- Guardar historial de búsquedas

Estimación: 8 puntos | **Prioridad:** Must Have

1.3.3 Historia de Usuario: HU10 - Visualización de Álbumes y Singles

Como usuario

Quiero visualizar álbumes y singles

Para explorar el catálogo musical

Criterios de Aceptación

Backend

- El sistema debe organizar canciones en álbumes
- Debe servir información completa de artistas y canciones
- Debe optimizar carga de metadatos y portadas
- Debe manejar singles como álbumes de una canción
- Debe proporcionar información de fechas de lanzamiento

Frontend

- Debe mostrar vista de álbumes con portadas
- Debe mostrar información detallada de artistas
- Debe listar canciones con duración y número de pista
- Debe permitir reproducir álbum completo
- Debe mostrar enlaces relacionados

Tasklist - Backend

- Crear modelos Artist, Album
- Relacionar canciones con álbumes y artistas
- Crear endpoints para álbumes y artistas
- Optimizar carga de portadas desde Supabase
- Implementar metadatos enriquecidos
- Crear endpoint para reproducir álbum completo

Tasklist - Frontend

- Crear componente AlbumListComponent
- Crear componente AlbumDetailComponent
- Crear componente ArtistDetailComponent
- Mostrar portadas de álbumes responsivas
- Implementar vista de lista de canciones
- Añadir botón "reproducir álbum"

Estimación: 5 puntos | **Prioridad:** Must Have

1.4. Épica 4: Sistema de Suscripciones

1.1.4 Historia de Usuario: HU11 - Plan Gratuito

Como usuario

Quiero acceder a un plan gratuito

Para probar la plataforma con limitaciones

Criterios de Aceptación

Backend



- El sistema debe definir claramente las funcionalidades limitadas
- Debe permitir registro sin requerir tarjeta de crédito
- Debe controlar acceso basado en tipo de suscripción
- Debe implementar límites (reproducciones, saltos, calidad)
- Debe mostrar promociones para upgrade

Frontend

- Debe permitir registro sin información de pago
- Debe mostrar claramente las limitaciones del plan gratuito
- Debe incluir llamadas a la acción para upgrade
- Debe mostrar anuncios o mensajes promocionales
- Debe bloquear funciones premium visualmente

Tasklist - Backend

- Crear modelo Subscription con tipos de plan
- Implementar middleware para verificar suscripción
- Definir límites para usuarios gratuitos
- Crear sistema de contador de uso
- Implementar lógica de restricciones
- Crear endpoints para información de plan

Tasklist - Frontend

- Crear servicio SubscriptionService
- Implementar guards para funciones premium



- Mostrar badges de "Premium" en funciones bloqueadas
- Crear modales de upgrade
- Mostrar información del plan actual
- Implementar banners promocionales

Estimación: 5 puntos | **Prioridad:** Must Have

1.2.4 Historia de Usuario: HU12 - Suscripción Premium Mensual

Como usuario

Quiero poder suscribirme a un plan Premium mensual

Para acceder a todas las funcionalidades

Criterios de Aceptación

Backend

- El sistema debe integrar con Stripe o Kushki para pagos
- Debe crear suscripciones recurrentes mensuales
- Debe actualizar el estado de suscripción del usuario
- Debe manejar webhooks de pago
- Debe desbloquear funcionalidades premium

Frontend

- Debe mostrar proceso de suscripción claro
- Debe detallar todos los beneficios del plan Premium
- Debe integrar formulario de pago seguro
- Debe confirmar suscripción exitosa

- Debe permitir gestión de la suscripción

Tasklist - Backend

- Integrar Stripe/Kushki SDK
- Crear endpoints para crear suscripción
- Implementar webhooks de pago
- Actualizar estado de usuario tras pago exitoso
- Crear sistema de facturación
- Manejar fallos de pago

Tasklist - Frontend

- Crear componente SubscriptionComponent
- Integrar Stripe/Kushki Elements
- Mostrar planes y precios claramente
- Implementar formulario de pago
- Crear página de confirmación
- Añadir gestión de suscripción en perfil

Estimación: 8 puntos | **Prioridad:** Must Have

1.5. Épica 5: Almacenamiento y Streaming

1.1.5 Historia de Usuario: HU13 - Solución de Almacenamiento

Como administrador

Quiero implementar una solución de almacenamiento

Para guardar los archivos de audio



Criterios de Aceptación

Backend

- El sistema debe integrar con Supabase Storage
- Debe gestionar subida eficiente de archivos grandes
- Debe organizar archivos por artista/álbum
- Debe implementar compresión y optimización
- Debe manejar backup y redundancia

Frontend

- Debe proporcionar interfaz de administración para subida
- Debe mostrar progreso de subida de archivos
- Debe validar formatos de audio soportados
- Debe permitir edición de metadatos
- Debe mostrar uso de almacenamiento

Tasklist - Backend

- Configurar Supabase Storage buckets
- Crear endpoints para subida de archivos
- Implementar procesamiento de metadatos
- Crear sistema de organización de archivos
- Implementar compresión automática
- Añadir validaciones de formato y tamaño

Tasklist - Frontend



- Crear componente de administración FileUploadComponent
- Implementar drag & drop para archivos
- Mostrar barra de progreso de subida
- Crear editor de metadatos
- Implementar validaciones del lado cliente
- Mostrar estadísticas de almacenamiento

Estimación: 13 puntos | **Prioridad:** Must Have

1.2.5 Historia de Usuario: HU14 - Streaming de Audio

Como usuario

Quiero que el streaming de audio sea fluido

Para disfrutar de la música sin interrupciones

Criterios de Aceptación

Backend

- El sistema debe implementar buffering adecuado
- Debe adaptarse a diferentes velocidades de conexión
- Debe servir audio en múltiples calidades
- Debe optimizar entrega mediante CDN
- Debe manejar reconexión automática

Frontend

- Debe pre-cargar siguiente canción en cola
- Debe ajustar calidad según conexión



- Debe mostrar estado de buffering
- Debe manejar interrupciones de red
- Debe sincronizar reproducción entre pestañas

Tasklist - Backend

- Implementar streaming adaptativo
- Configurar CDN para archivos de audio
- Crear múltiples calidades de audio
- Implementar range requests para audio
- Optimizar headers de caché
- Monitorear latencia de streaming

Tasklist - Frontend

- Implementar buffering inteligente
- Crear detector de velocidad de conexión
- Implementar cola de pre-carga
- Manejar estados de red
- Sincronizar estado entre pestañas
- Mostrar indicadores de calidad de streaming

Estimación: 13 puntos | **Prioridad:** Must Have

1.6. Épica 6: Pagos

1.1.6 *Historia de Usuario: HU15 - Pago de Suscripción*

Como usuario premium

Quiero poder pagar mi suscripción

Para acceder a las funcionalidades completas

Criterios de Aceptación

Backend

- El sistema debe integrar con Stripe Test Mode o Kushki
- Debe procesar pagos de forma segura
- Debe manejar diferentes métodos de pago
- Debe generar facturas automáticamente
- Debe manejar fallos y reintento de pagos

Frontend

- Debe tener interfaz de pago clara e intuitiva
- Debe mostrar información de seguridad
- Debe soportar múltiples métodos de pago
- Debe confirmar transacciones exitosas
- Debe manejar errores de pago graciosamente

Tasklist - Backend

- Configurar webhooks de Stripe/Kushki
- Crear endpoints para procesar pagos



- Implementar generación de facturas
- Manejar estados de pago (pendiente, exitoso, fallido)
- Crear sistema de reintento automático
- Implementar logging de transacciones

Tasklist - Frontend

- Integrar Stripe/Kushki checkout
- Crear formulario de información de pago
- Mostrar opciones de método de pago
- Implementar manejo de errores
- Crear página de confirmación de pago
- Añadir cargando durante procesamiento

Estimación: 13 puntos | **Prioridad:** Must Have

1.2.6 Historia de Usuario: HU16 - Gestión de Suscripciones

Como administrador

Quiero gestionar las suscripciones y renovaciones

Para mantener el control de los usuarios premium

Criterios de Aceptación

Backend

- El sistema debe tener panel de gestión de suscripciones
- Debe enviar notificaciones de renovación/caducidad
- Debe manejar cancelaciones y reactivaciones



- Debe generar reportes de suscripciones
- Debe automatizar procesos de facturación

Frontend

- Debe mostrar dashboard de suscripciones activas
- Debe permitir búsqueda y filtrado de suscripciones
- Debe mostrar métricas de renovación y cancelación
- Debe permitir acciones administrativas
- Debe generar reportes exportables

Tasklist - Backend

- Crear endpoints de administración de suscripciones
- Implementar sistema de notificaciones
- Crear jobs programados para renovaciones
- Implementar métricas y analytics
- Crear sistema de reportes
- Manejar cancelaciones y reembolsos

Tasklist - Frontend

- Crear SubscriptionAdminComponent
- Implementar tabla de suscripciones con filtros
- Crear dashboard de métricas
- Implementar acciones administrativas
- Crear generador de reportes

- Añadir notificaciones en tiempo real

Estimación: 8 puntos | **Prioridad:** Must Have

1.7. Épica 7: Modo Offline

1.1.7 *Historia de Usuario: HU17 - Descarga de Canciones*

Como usuario premium

Quiero descargar canciones

Para escucharlas sin conexión

Criterios de Aceptación

Backend

- El sistema debe validar suscripción premium para descargas
- Debe gestionar límites de descargas por usuario
- Debe servir archivos optimizados para descarga
- Debe rastrear descargas para royalties
- Debe manejar expiración de contenido descargado

Frontend

- Debe mostrar botón de descarga para usuarios premium
- Debe gestionar contenido descargado localmente
- Debe permitir reproducción offline
- Debe mostrar estado de descarga
- Debe sincronizar contenido cuando vuelva la conexión

Tasklist - Backend



- Crear endpoint para validar descargas
- Implementar límites de descarga por usuario
- Crear sistema de tokens de descarga
- Implementar rastreo de descargas
- Manejar expiración de contenido
- Optimizar archivos para descarga

Tasklist - Frontend

- Implementar Service Worker para offline
- Crear gestor de descargas DownloadManager
- Implementar almacenamiento local de audio
- Crear interfaz de gestión de descargas
- Implementar reproductor offline
- Manejar sincronización online/offline

Estimación: 13 puntos | **Prioridad:** Must Have

1.8.Épica 8: Dashboard de Administración

1.1.8 Historia de Usuario: HU18 - Gestión de Usuarios

Como administrador

Quiero gestionar usuarios

Para mantener control sobre la plataforma

Criterios de Aceptación

Backend



- El sistema debe proporcionar endpoints CRUD para usuarios
- Debe implementar filtros de búsqueda avanzados
- Debe manejar roles y permisos de usuario
- Debe generar reportes de actividad de usuarios
- Debe permitir suspensión y reactivación de cuentas

Frontend

- Debe tener interfaz de CRUD de usuarios completa
- Debe mostrar tabla de usuarios con paginación
- Debe incluir filtros y búsqueda avanzada
- Debe permitir acciones en lote
- Debe mostrar estadísticas de usuarios

Tasklist - Backend

- Crear endpoints de administración de usuarios
- Implementar sistema de roles y permisos
- Crear filtros avanzados de búsqueda
- Implementar acciones administrativas
- Crear sistema de auditoría
- Generar reportes de usuarios

Tasklist - Frontend

- Crear UserManagementComponent



- Implementar tabla con filtros avanzados
- Crear modales para edición de usuarios
- Implementar acciones en lote
- Crear dashboard de estadísticas
- Añadir sistema de confirmaciones

Estimación: 8 puntos | **Prioridad:** Should Have

1.2.8 Historia de Usuario: HU19 - Gestión de Contenido Musical

Como administrador

Quiero gestionar el contenido musical

Para mantener actualizado el catálogo

Criterios de Aceptación

Backend

- El sistema debe permitir subida de nuevas canciones/álbumes
- Debe proporcionar herramientas de edición de metadatos
- Debe validar calidad y formato de archivos
- Debe gestionar derechos y licencias
- Debe implementar moderación de contenido

Frontend

- Debe tener interfaz para subida masiva de contenido
- Debe permitir edición de metadatos en lote
- Debe mostrar preview de audio antes de publicar



- Debe gestionar categorización y etiquetado
- Debe mostrar estadísticas de contenido

Tasklist - Backend

- Crear endpoints para gestión de contenido
- Implementar validación de archivos de audio
- Crear sistema de metadatos enriquecidos
- Implementar moderación automática
- Crear sistema de licencias y derechos
- Generar reportes de contenido

Tasklist - Frontend

- Crear ContentManagementComponent
- Implementar subida masiva con drag & drop
- Crear editor de metadatos en lote
- Implementar preview de audio
- Crear sistema de categorización
- Mostrar estadísticas de catálogo

Estimación: 8 puntos | **Prioridad:** Should Have

1.3.8 Historia de Usuario: HU20 - Estadísticas de Uso

Como administrador

Quiero ver estadísticas básicas de uso

Para entender el comportamiento de los usuarios



Criterios de Aceptación

Backend

- El sistema debe recopilar métricas de uso en tiempo real
- Debe generar gráficos y reportes analíticos
- Debe proporcionar insights de comportamiento de usuarios
- Debe mostrar tendencias de contenido popular
- Debe exportar datos para análisis externo

Frontend

- Debe mostrar dashboard con métricas clave
- Debe incluir gráficos interactivos de uso
- Debe permitir filtrado por períodos de tiempo
- Debe mostrar comparativas históricas
- Debe ser responsive para diferentes dispositivos

Tasklist - Backend

- Implementar sistema de analytics
- Crear endpoints para métricas
- Generar reportes automáticos
- Implementar métricas en tiempo real
- Crear sistema de exportación
- Optimizar consultas de analytics

Tasklist - Frontend



- Crear AnalyticsComponent
- Integrar Chart.js para gráficos
- Implementar filtros de fecha
- Crear widgets de métricas
- Implementar exportación de reportes
- Añadir actualizaciones en tiempo real

Estimación: 8 puntos | **Prioridad:** Should Have

1.9.Épica 9: Plan Premium Anual

1.1.9 Historia de Usuario: HU21 - Suscripción Anual

Como usuario

Quiero poder suscribirme a un plan anual con descuento

Para ahorrar dinero

Criterios de Aceptación

Backend

- El sistema debe ofrecer suscripción anual con descuento
- Debe calcular correctamente el precio con descuento
- Debe manejar facturación anual
- Debe aplicar promociones automáticamente
- Debe gestionar renovaciones anuales

Frontend

- Debe mostrar opción de suscripción anual visiblemente



- Debe destacar el ahorro vs plan mensual
- Debe mostrar desglose de precios
- Debe incluir términos y condiciones claros
- Debe confirmar el cambio de plan

Tasklist - Backend

- Configurar plan anual en Stripe/Kushki
- Implementar lógica de descuentos
- Crear endpoints para suscripción anual
- Manejar cambios de plan mensual a anual
- Implementar prorrateo de pagos
- Crear notificaciones de renovación anual

Tasklist - Frontend

- Actualizar SubscriptionComponent con opción anual
- Mostrar calculadora de ahorros
- Crear toggle entre mensual/anual
- Implementar upgrade de plan
- Mostrar beneficios del plan anual
- Añadir confirmación de cambio de plan

Estimación: 5 puntos | **Prioridad:** Should Have

1.10. Épica 10: Recomendaciones Básicas

1.1.10 Historia de Usuario: HU22 - Sugerencias Personalizadas

Como usuario

Quiero recibir sugerencias basadas en mi historial

Para descubrir nueva música

Criterios de Aceptación

Backend

- El sistema debe implementar algoritmo básico de recomendación
- Debe analizar historial de reproducciones del usuario
- Debe considerar géneros favoritos y artistas escuchados
- Debe actualizar recomendaciones periódicamente
- Debe evitar recomendar contenido ya conocido

Frontend

- Debe mostrar sección "Recomendado para ti"
- Debe actualizar recomendaciones dinámicamente
- Debe permitir feedback sobre recomendaciones
- Debe mostrar razón de la recomendación
- Debe integrar con el reproductor principal

Tasklist - Backend

- Implementar algoritmo de collaborative filtering básico
- Crear modelo de preferencias de usuario



- Generar recomendaciones basadas en similitud
- Crear endpoint GET /api/recommendations
- Implementar sistema de feedback
- Crear job programado para actualizar recomendaciones

Tasklist - Frontend

- Crear RecommendationsComponent
- Mostrar carrusel de canciones recomendadas
- Implementar sistema de like/dislike
- Mostrar explicación de recomendaciones
- Integrar con reproductor
- Crear página dedicada de recomendaciones

Estimación: 13 puntos | **Prioridad:** Should Have

1.2.10 Historia de Usuario: HU23 - Playlists Populares

Como usuario

Quiero ver playlists populares por género

Para explorar música nueva

Criterios de Aceptación

Backend

- El sistema debe mantener playlists curadas por género
- Debe actualizar popularidad basada en reproducciones
- Debe permitir creación de playlists editoriales



- Debe proporcionar métricas de popularidad
- Debe actualizar contenido periódicamente

Frontend

- Debe mostrar playlists curadas por género
- Debe destacar playlists más populares
- Debe permitir navegación por categorías
- Debe mostrar estadísticas de popularidad
- Debe integrar con sistema de reproducción

Tasklist - Backend

- Crear modelo CuratedPlaylist
- Implementar algoritmo de popularidad
- Crear endpoints para playlists populares
- Implementar sistema editorial
- Crear métricas de engagement
- Automatizar actualización de rankings

Tasklist - Frontend

- Crear PopularPlaylistsComponent
- Mostrar ranking de playlists
- Implementar filtros por género
- Crear vista de playlist popular
- Mostrar métricas de popularidad



- Integrar con sistema de favoritos

Estimación: 8 puntos | **Prioridad:** Should Have

1.11. Épica 11: Búsqueda Avanzada

1.1.11 Historia de Usuario: HU24 - Filtros Avanzados

Como usuario

Quiero usar filtros avanzados en mis búsquedas

Para encontrar música específica

Criterios de Aceptación

Backend

- El sistema debe soportar múltiples filtros combinados
- Debe permitir filtrado por artista, álbum, año, género
- Debe optimizar consultas complejas
- Debe manejar rangos de fechas y duración
- Debe proporcionar auto-completado para filtros

Frontend

- Debe mostrar interfaz de filtros por artista, álbum, año
- Debe permitir combinación de múltiples filtros
- Debe mostrar resultados en tiempo real
- Debe guardar filtros aplicados
- Debe permitir resetear filtros fácilmente

Tasklist - Backend



- Extender endpoint de búsqueda con filtros
- Optimizar queries con índices compuestos
- Implementar filtros por rango de fechas
- Crear auto-completado para artistas/álbumes
- Implementar paginación eficiente
- Añadir logging de búsquedas

Tasklist - Frontend

- Extender SearchComponent con filtros avanzados
- Crear componente FilterPanel
- Implementar sliders para año y duración
- Agregar auto-complete para artistas
- Crear chips de filtros activos
- Implementar búsqueda en tiempo real

Estimación: 8 puntos | **Prioridad:** Should Have

1.2.11 Historia de Usuario: HU25 - Resultados Categorizados

Como usuario

Quiero ver resultados categorizados

Para navegar mejor por los resultados de búsqueda

Criterios de Aceptación

Backend

- El sistema debe separar resultados por categorías



- Debe ordenar por relevancia dentro de cada categoría
- Debe proporcionar conteo de resultados por categoría
- Debe optimizar consultas para múltiples categorías
- Debe manejar paginación por categoría

Frontend

- Debe mostrar interfaz clara de resultados categorizados
- Debe separar en pestañas: artistas, álbumes, canciones
- Debe mostrar contador de resultados por categoría
- Debe permitir navegación rápida entre categorías
- Debe mantener estado de búsqueda entre categorías

Tasklist - Backend

- Modificar endpoint de búsqueda para categorización
- Implementar conteo eficiente por categoría
- Optimizar paginación por categoría
- Crear índices específicos por tipo
- Implementar ordenamiento por relevancia
- Añadir métricas de búsqueda

Tasklist - Frontend

- Crear SearchResultsTabsComponent
- Implementar navegación por pestañas
- Mostrar contadores de resultados



- Crear componentes específicos por categoría
- Implementar paginación independiente
- Añadir estado persistente de búsqueda

Estimación: 5 puntos | **Prioridad:** Should Have

1.12. Resumen del Backlog

1.1.12 MVP (Must Have) - 187 puntos

Épica 1: Sistema de Autenticación y Perfiles (23 pts)

Épica 2: Reproductor de Audio con Playlists (26 pts)

Épica 3: Catálogo de Música Organizado (21 pts)

Épica 4: Sistema de Suscripciones (13 pts)

Épica 5: Almacenamiento y Streaming (26 pts)

Épica 6: Pagos (21 pts)

Épica 7: Modo Offline (13 pts)

1.2.12 Fase 2 (Should Have) - 42 puntos

Épica 8: Dashboard de Administración (24 pts)

Épica 9: Plan Premium Anual (5 pts)

Épica 10: Recomendaciones Básicas (21 pts)

Épica 11: Búsqueda Avanzada (13 pts)

1.3.12 Stack Tecnológico

Frontend: Angular



Backend: Django Rest Framework

Base de Datos: PostgreSQL

Almacenamiento: Supabase Storage

Pagos: Stripe Test Mode o Kushki

1.4.12 Estimación Total

Total de puntos: 229 puntos

Sprint de 1 semana (20 pts): ~3 sprints (3 semanas)

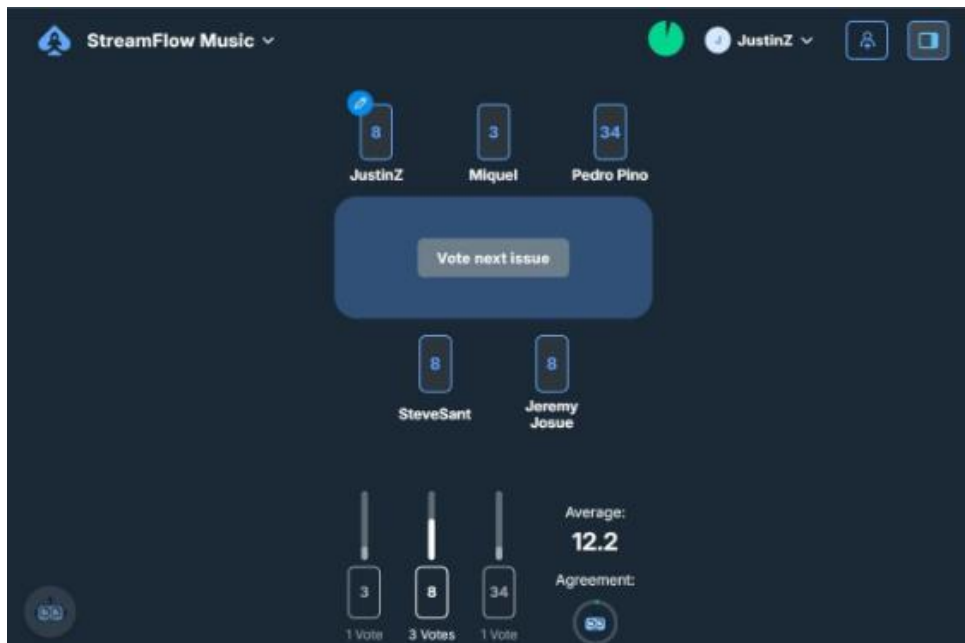
MVP: ~1 sprints (1 semana)

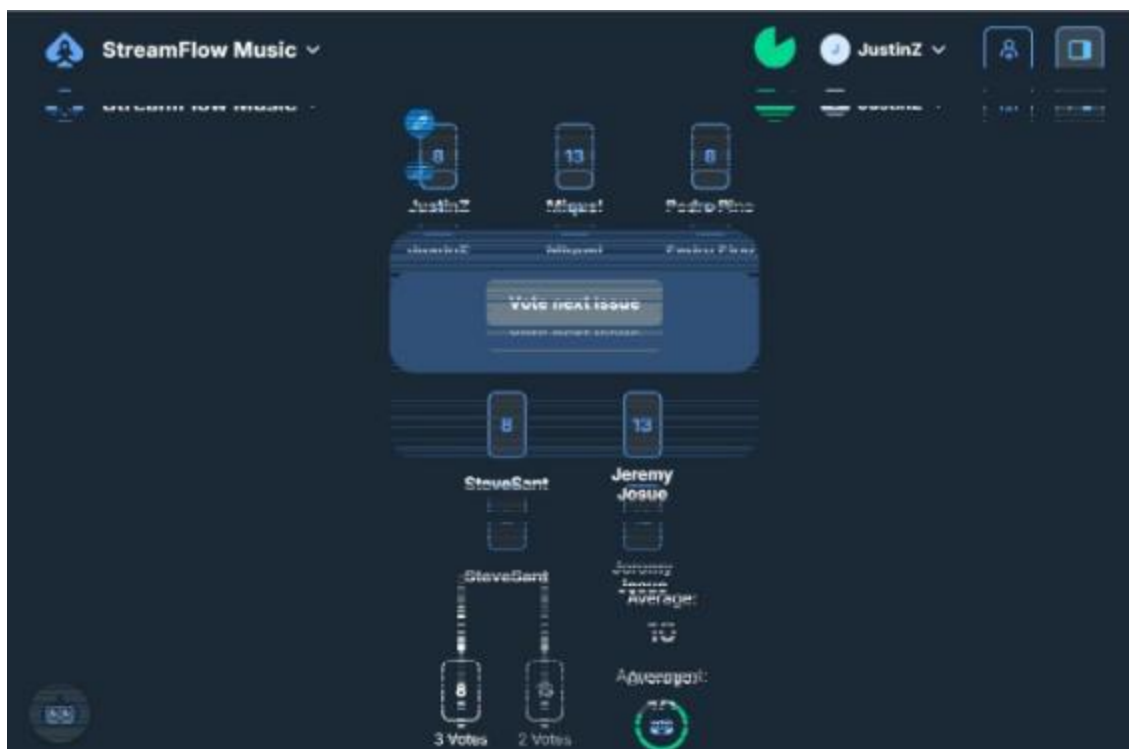
2. Roadmap del proyecto



3. Estimaciones del proyecto

<https://planningpokeronline.com/knOLBnSn70CrPZdVuebC/>
<https://planningpokeronline.com/knOLBnSn70CrPZdVuebC/>





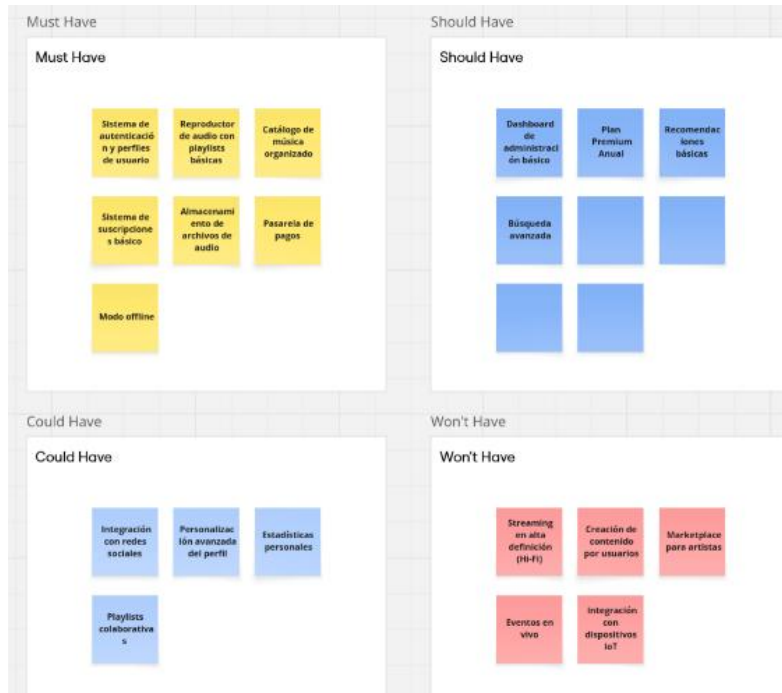




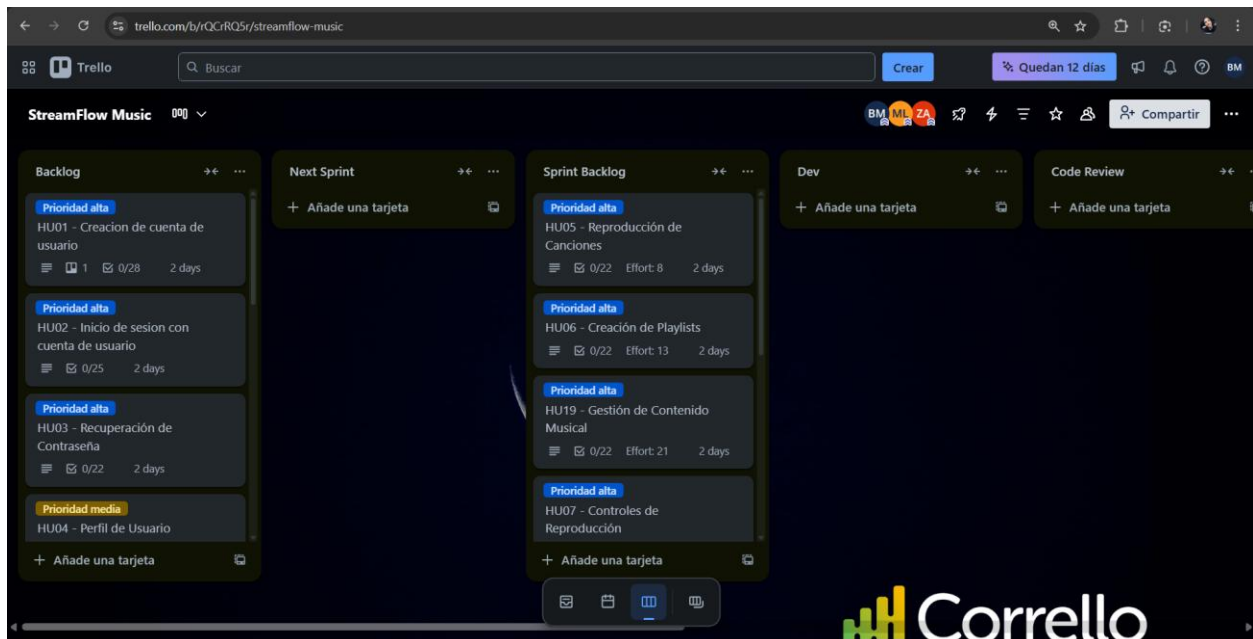


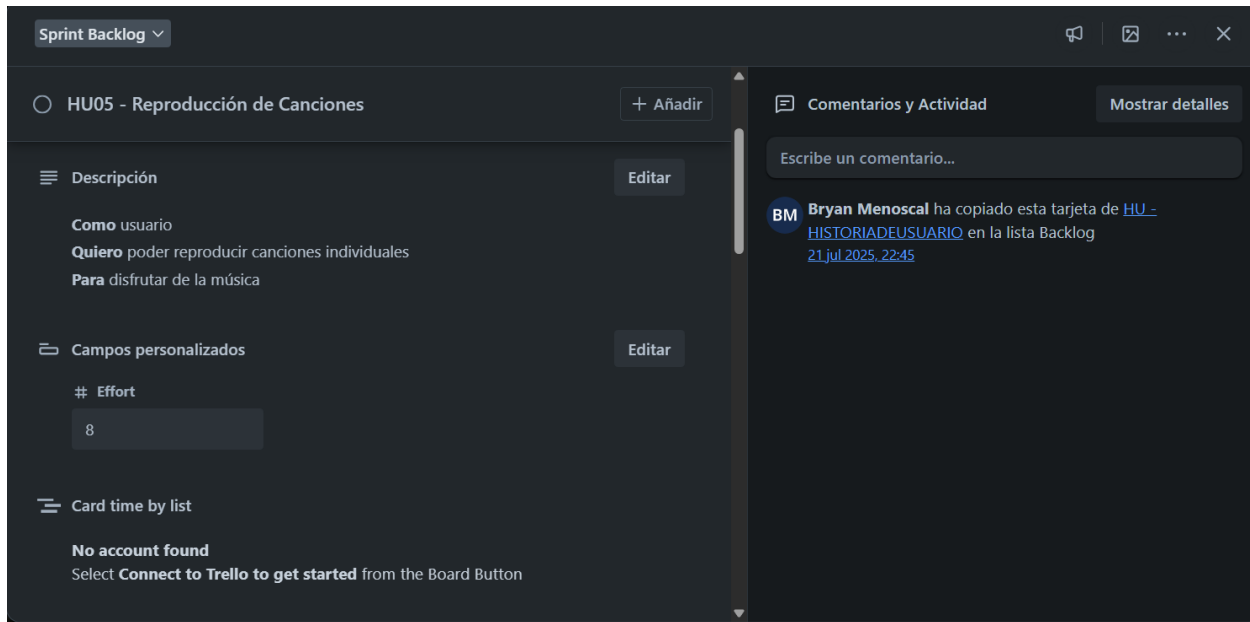
4. Priorización MoSCoW

https://miro.com/app/board/uXiVJbqGVug=



5. Sprint Planning, Daily Standups, Sprint Review y Retrospectives





Sprint Backlog

HU05 - Reproducción de Canciones + Añadir

Descripción Editar

Como usuario
Quiero poder reproducir canciones individuales
Para disfrutar de la música

Campos personalizados Editar

Effort
8

Card time by list

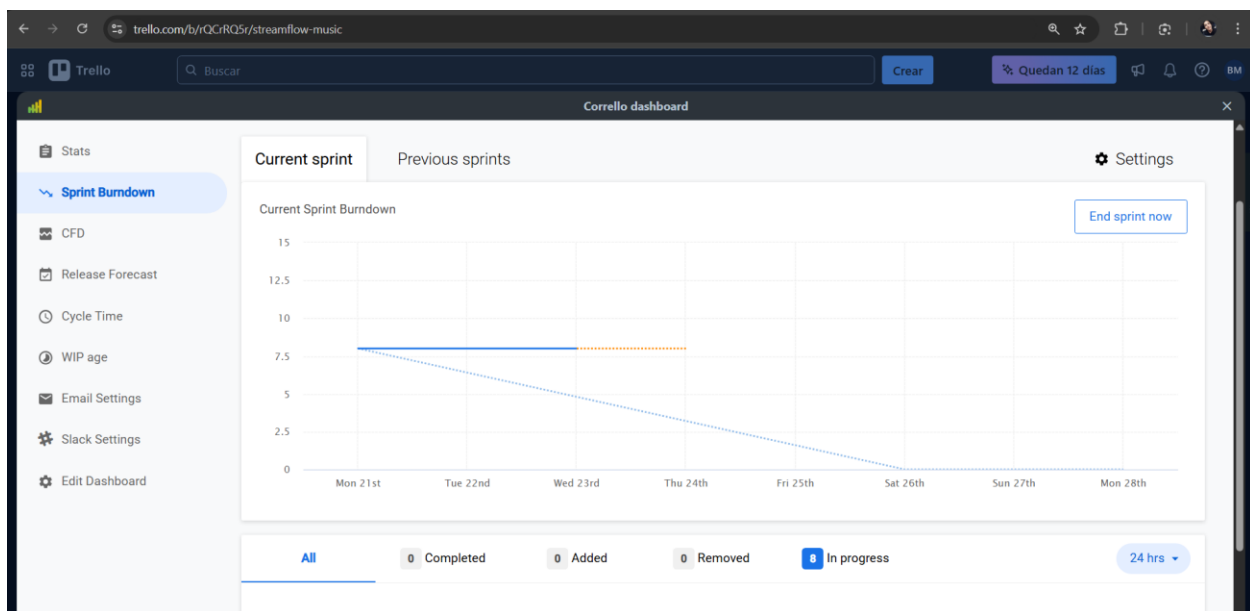
No account found
Select **Connect to Trello to get started** from the Board Button

Comentarios y Actividad Mostrar detalles

Escribe un comentario...

BM Bryan Menoscal ha copiado esta tarjeta de [HU - HISTORIADEUSUARIO](#) en la lista Backlog
21.jul.2025, 22:45

6. Burndown charts





7. Otros gráficos



8. Metodologías aplicadas

8.1.Scrum

Nombre	Rol	Observacion
Bryan Menoscal	Product Owner	Responsable de definir y priorizar requisitos
Pedro Zambrano	Scrum Master	Facilita el proceso y elimina impedimentos
Justin Zambrano	Developer	Desarrollo, pruebas, integración
Cesar Arteaga	Developer	Desarrollo, diseño, documentación
Jeremy Delgado	Developer	Desarrollo, QA, soporte técnico
Miquel Muñiz	Developer	Desarrollo, automatización, devops

- **Ceremonias:** Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective (y el Sprint como contenedor).
- **Artefactos:** Product Backlog, Sprint Backlog, Increment.

