**✅🟡🔲 Check List Proyecto Musicalendaria**

**🧱 1. Servidor base funcional ✅**

* ✅ Configurado con Express.
* ✅ Middleware para parseo (express.json()), CORS, etc.
* ✅ Rutas básicas funcionando.
* 🟡 Simulación de datos o conexión real (¡**ahora estamos en esto**!).
* 🔲 Escalabilidad (índices, paginación).
* 🔲 Manejo de errores global (404, 500, etc).

**👥 2. Autenticación y roles 🟡**

* 🟡 Registro, login y logout con JWT.
* 🟡 Protección de rutas con JWT.
* 🔲 Middleware de autorización por rol:
  + Admin: gestiona artistas, eventos, calendario.
  + Artista: solo su info.
* 🔲 Guardado del token en localStorage / sessionStorage.
* 🔲 Validación de JWT en el frontend para vistas privadas.

**🧩 3. CRUD del backend 🔲**

* 🔲 Controladores y rutas para:
  + 🔲 Usuarios
  + 🔲 Artistas
  + 🔲 Eventos
  + 🔲 Calendario
* 🔲 Verificado con Postman.
* 🔲 Validaciones (emails, teléfonos, enlaces).
* 🔲 Validaciones específicas (prohibir artistas como Eminem, Dua Lipa, etc).

**🎨 4. Frontend conectado 🔲**

* 🔲 HTML + Tailwind estructurado.
* 🔲 Fetch/AJAX para backend (registro, login, CRUD).
* 🔲 Formularios: login, registro, carga de datos.
* 🔲 Vistas según tipo de usuario.
* 🔲 UI responsiva.
* 🔲 Validaciones de enlaces.

**🛡️ 5. Protección en frontend 🔲**

* 🔲 Validar JWT antes de mostrar vistas.
* 🔲 Redirección según permisos.
* 🔲 Control de sesión (renovación de token).

**🗃️ 6. Base de datos real 🟡**

* ✅ Usamos **MySQL con Uniform Server**.
* ✅ Conexión con módulo mysql2.
* 🟡 Creamos carpeta models/ con archivos para simular datos.
* 🔲 Relaciones:
  + Un artista tiene muchos eventos.
  + Eventos ligados a plataformas/redes.
* 🔲 Datos de prueba reales.
* 🔲 Índices, optimización.
* 🔲 Respaldo automático.

**✅ 7. Pruebas, limpieza y documentación 🔲**

* 🔲 Flujo completo probado (registro → login → panel → CRUD → logout).
* 🔲 Postman listo.
* 🔲 README con instalación y uso.
* 🔲 Código limpio y organizado.

**⚙️ 8. Funciones adicionales recomendadas 🔲**

* 🔲 Notificaciones por correo.
* 🔲 Alertas en UI.
* 🔲 Búsqueda y filtrado de artistas/eventos.
* 🔲 Integración con APIs de streaming (Spotify, etc).
* 🔲 Manejo de fechas/eventos (validación, calendario dinámico).

**🧹 9. Mantenimiento y escalabilidad 🔲**

* 🔲 Optimización para crecimiento.
* 🔲 Plan de mantenimiento y seguridad.

.

**Check List Completo para el Proyecto Musicalendaria**

Este checklist está basado en los requerimientos funcionales y no funcionales del proyecto **Musicalendaria**. Lo he dividido en secciones para que puedas guiarte fácilmente y verificar tu progreso.

**🧱 1. Servidor Base Funcional**

* **✅ Configurado con Express.**
* **✅ Middleware para parseo (**[express.json()](vscode-file://vscode-app/c:/Users/Usuario/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ")**), CORS, etc.**
* **✅ Rutas básicas funcionando.**
* **✅ Conexión real a MySQL usando mysql2.**
* **🔲 Escalabilidad:**
  + Implementar paginación e índices en las consultas.
* **🔲 Manejo de errores global:**
  + Crear un middleware para manejar errores 404 y 500.

**👥 2. Autenticación y Roles**

* **✅ Registro y login con JWT.**
* **✅ Protección de rutas con JWT usando el middleware**[protect](vscode-file://vscode-app/c:/Users/Usuario/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o ")**.**
* **🟡 Middleware de autorización por rol:**
  + Implementado parcialmente con [authorize](vscode-file://vscode-app/c:/Users/Usuario/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html" \o "), pero falta aplicarlo en más rutas específicas.
* **🔲 Logout:**
  + Implementar una ruta para invalidar el token (opcional, ya que el logout puede manejarse en el frontend).
* **🔲 Guardado del token en localStorage / sessionStorage (frontend).**
* **🔲 Validación de JWT en el frontend para vistas privadas.**

**🧩 3. CRUD del Backend**

* **🟡 Controladores y rutas para:**
  + **🟡 Usuarios:** Registro y login están implementados, pero falta un CRUD completo.
  + **🔲 Artistas:** Crear modelo, controlador y rutas para gestionar artistas.
  + **🔲 Eventos:** Crear modelo, controlador y rutas para gestionar eventos.
  + **🔲 Calendario:** Crear modelo, controlador y rutas para gestionar el calendario.
* **🔲 Verificado con Postman:**
  + Probar todas las rutas y funcionalidades.
* **🔲 Validaciones de datos:**
  + Validar emails, teléfonos, enlaces, etc.
* **🔲 Validaciones específicas:**
  + Prohibir artistas como Eminem, Dua Lipa, Catriel o Paco Amoroso.

**🎨 4. Frontend Conectado**

* **🔲 HTML + Tailwind estructurado.**
* **🔲 Fetch/AJAX para interactuar con el backend (registro, login, CRUD).**
* **🔲 Formularios:**
  + Crear formularios para login, registro y carga de datos.
* **🔲 Vistas según tipo de usuario:**
  + Mostrar vistas diferentes para administradores y artistas.
* **🔲 UI responsiva.**
* **🔲 Validaciones de enlaces en el frontend.**

**🛡️ 5. Protección en el Frontend**

* **🔲 Validar JWT antes de mostrar vistas privadas.**
* **🔲 Redirección según permisos:**
  + Redirigir usuarios no autorizados a páginas de error o login.
* **🔲 Control de sesión:**
  + Implementar renovación de token o manejo de expiración.

**🗃️ 6. Base de Datos Real**

* **✅ Usar MySQL con Uniform Server.**
* **✅ Conexión con módulo mysql2.**
* **✅ Crear carpeta models/ para gestionar datos.**
* **🔲 Relaciones:**
  + Definir relaciones entre tablas (por ejemplo, un artista tiene muchos eventos).
* **🔲 Datos de prueba reales:**
  + Insertar datos de prueba en la base de datos.
* **🔲 Índices y optimización:**
  + Optimizar consultas y agregar índices en las tablas.
* **🔲 Respaldo automático:**
  + Configurar un sistema de respaldo para la base de datos.

**✅ 7. Pruebas, Limpieza y Documentación**

* **🔲 Flujo completo probado:**
  + Probar el flujo completo desde el registro hasta el CRUD y logout.
* **🔲 Postman listo:**
  + Crear una colección de Postman con todas las rutas y ejemplos de solicitudes.
* **🔲 README con instalación y uso:**
  + Documentar cómo instalar y usar el proyecto en el archivo [README.md](vscode-file://vscode-app/c:/Users/Usuario/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html).
* **🔲 Código limpio y organizado:**
  + Revisar y limpiar el código, agregar comentarios y eliminar redundancias.

**⚙️ 8. Funciones Adicionales Recomendadas**

* **🔲 Notificaciones por correo:**
  + Enviar correos de confirmación o notificaciones (opcional).
* **🔲 Alertas en la UI:**
  + Mostrar mensajes de éxito o error en el frontend.
* **🔲 Búsqueda y filtrado:**
  + Implementar búsqueda y filtrado de artistas y eventos.
* **🔲 Integración con APIs de streaming:**
  + Conectar con APIs como Spotify, YouTube, etc.
* **🔲 Manejo de fechas/eventos:**
  + Validar fechas y crear un calendario dinámico.

**🧹 9. Mantenimiento y Escalabilidad**

* **🔲 Optimización para crecimiento:**
  + Asegurar que el sistema pueda manejar más usuarios y datos.
* **🔲 Plan de mantenimiento y seguridad:**
  + Documentar un plan para mantener y asegurar el sistema.
* UTILIZAR ARQUITECTURA MVVM