

Portada

Generador de Listas Etiquetadas

Integrantes: Diego Mercado Coello, Luis Eduardo Gonzalez Gloria, Federico Humberto Zaragoza Manuel

Materia: Desarrollo de Aplicaciones Web / ITESO

Fecha: 2025-05-10

Enlace a Render y Credenciales de Prueba

La aplicación está desplegada en Render: <https://generador-de-listas-etiquetadas.onrender.com>

Credenciales de prueba:

- Admin: `admin@example.com` / `admin123`
- Usuario común: `user@example.com` / `user123`

Descripción Breve del Proyecto

El Generador de Listas Etiquetadas es una aplicación web para crear, organizar y compartir listas de ítems categorizados mediante etiquetas y vistas personalizadas. Cada usuario puede gestionar sus propias listas e ítems, asignarles campos especiales (fecha, prioridad, notas), usar filtros por etiquetas, marcar favoritos o archivar, y compartir listas con otros. Los administradores, además, tienen un panel para gestionar usuarios y consultar estadísticas de uso.

Flujo de la Aplicación

Usuario Común

1. Registro – Login (JWT almacenado en LocalStorage)
2. Creación de nuevas listas (título, descripción)
3. Gestión de ítems dentro de cada lista (título, descripción, fecha, etiquetas, campos especiales, estado)
4. Filtrado y búsqueda por etiquetas e ítems
5. Definición de vistas personalizadas (conjuntos de filtros y ordenamientos)
6. Aplicación de vistas personalizadas para visualizar ítems.

- 7. Compartir listas con otros usuarios (asignando permisos de 'ver' o 'editar')
- 8. Marcar ítems/listas como favoritos o archivadas
- 9. Visualización de listas recientes, favoritas y archivadas en el dashboard.
- 10. Edición de perfil (nombre, email, contraseña).

Usuario Administrador

- 1. Login a panel de administración
- 2. Listado y gestión de usuarios (crear, editar, eliminar, cambiar rol)
- 3. Consulta de estadísticas globales (número de usuarios, listas creadas, actividad reciente)
- 4. Acceso a todas las funcionalidades de un usuario común para sus propias listas/ítems.

REST API Documentada

Método	Ruta	Descripción	Acceso	Parámetros (Body/Params)
Autenticación (/api/auth)				
POST	/register	Registrar nuevo usuario	Público	Body: { username, password, full_name, birthDate, role }
POST	/login	Iniciar sesión	Público	Body: { email, password }
GET	/profile	Obtener perfil de usuario actual	Privado	-
PUT	/profile	Actualizar perfil de usuario actual	Privado	Body: { nombre, email, fecha_nacimiento }
PUT	/updatepassword	Cambiar contraseña	Privado	Body: { currentPassword, newPassword }
Usuarios (/api/users)				

Método	Ruta	Descripción	Acceso	Parámetros (Body/Params)
GET	/	Listar usuarios (paginado)	Privado/Admin	Query: <code>page</code> , <code>lin</code>
GET	/stats	Estadísticas de usuarios	Privado/Admin	-
GET	/:id	Obtener usuario por ID	Privado/Admin	Params: <code>id</code>
PUT	/:id	Actualizar un usuario (admin)	Privado/Admin	Params: <code>id</code> , Body: <code>nombre</code> , <code>fecha_nacimient</code>
PUT	/:id/role	Cambiar rol de usuario	Privado/Admin	Params: <code>id</code> , Body:
DELETE	/:id	Eliminar usuario	Privado/Admin	Params: <code>id</code>
PUT	/me/lists/favorite/:listId	Toggle lista favorita	Privado	Params: <code>listId</code>
PUT	/me/lists/archive/:listId	Toggle lista archivada	Privado	Params: <code>listId</code>
Listas (/api/listas)				
GET	/	Listar listas del usuario (paginado)	Privado	Query: <code>page</code> , <code>lin</code>

Método	Ruta	Descripción	Acceso	Parámetros (Body/Params)
POST	/	Crear nueva lista	Privado	Body: { title, description, camposEspaciales }
GET	/:id	Obtener lista específica y sus ítems	Privado	Params: id
PUT	/:id	Actualizar lista	Privado	Params: id, Body: { description, camposEspaciales }
DELETE	/:id	Eliminar lista	Privado	Params: id
POST	/:listId/share	Compartir lista con colaborador	Privado	Params: listId, emailOrUsername, permiso }
DELETE	/:listId/share/:collabUserId	Quitar colaborador	Privado	Params: listId, collaboratorUserId
Ítems (/api/items)				
GET	/filtrar	Filtrar ítems (estado, etiq, orden)	Privado	Query: listId, etiquetas (CSV), etiqueta_logica, sortField, sort, page, limit
POST	/	Crear nuevo ítem	Privado	Body: { listId, title, description, etiquetas (array valoresCamposEspaciales) }
GET	/:id	Obtener ítem específico	Privado	Params: id

Método	Ruta	Descripción	Acceso	Parámetros (Body/Params)
PUT	/:id	Actualizar Ítem	Privado	Params: <code>id</code> , Body: <code>a actualizar</code> }
DELETE	/:id	Eliminar Ítem	Privado	Params: <code>id</code>
**Etiquetas (/api/etiquetas)				
GET	/	Listar etiquetas (paginado)	Privado	Query: <code>page</code> , <code>lin</code>
POST	/	Crear nueva etiqueta	Privado	Body: { <code>nombre</code> , <code>co</code>
PUT	/:id	Actualizar etiqueta	Privado	Params: <code>id</code> , Body: <code>nombre</code> , <code>color</code> }
DELETE	/:id	Eliminar etiqueta	Privado	Params: <code>id</code>
GET	/estadisticas	Estadísticas de uso de etiquetas	Privado	-
**Vistas Personalizadas (/api/vistas)				
GET	/	Listar vistas del usuario (paginado)	Privado	Query: <code>page</code> , <code>lin</code>

Método	Ruta	Descripción	Acceso	Parámetros (Body/Params)
POST	/	Crear nueva vista	Privado	Body: { nombre , ordenamientos }
GET	/:id	Obtener vista específica	Privado	Params: id
PUT	/:id	Actualizar vista	Privado	Params: id , Body: nombre , filtros , ordenamientos }
DELETE	/:id	Eliminar vista	Privado	Params: id


Screenshots de Pantallas Importantes

A continuación, se muestran algunas pantallas clave de la aplicación:


1. Dashboard Principal

 Dashboard Principal *Descripción: Vista principal donde el usuario gestiona sus listas.*


2. Detalle de Lista con Ítems y Campos Especiales

 Detalle de Lista *Descripción: Visualización de los ítems de una lista, incluyendo etiquetas y valores de campos especiales definidos.*

3. Panel de Administración - Gestión de Usuarios

 Panel de Administración *Descripción: Interfaz para que el administrador gestione los usuarios del sistema.*

4. Creación/Edición de Vistas Personalizadas

 Vistas Personalizadas *Descripción: Formulario para definir filtros y ordenamientos de una vista personalizada.*

5. Perfil

Distribución de Actividades

Diego (45%):

- **Front-end:** Desarrollo de componentes y lógica para la gestión de Ítems (CRUD completo, modal de edición), Etiquetas (gestor, selección en modal de ítem), Vistas Personalizadas (creación, listado, eliminación, aplicación en detalle de lista) y Campos Especiales (definición en lista, asignación y visualización en ítems). Integración de estas funcionalidades con la API.
- **Back-end:** Diseño e implementación de los modelos Mongoose para Ítems, Etiquetas, Vistas Personalizadas y la estructura de Campos Especiales en Listas/Ítems. Desarrollo de las rutas API REST y controladores para el CRUD completo de estas entidades, incluyendo la lógica de filtrado avanzado para ítems (estado, etiquetas con AND/OR, ordenamiento) y la lógica de persistencia para las relaciones (ítem-etiquetas).

Luis (30%):

- **Front-end:** Desarrollo de los componentes principales del Dashboard (listado de listas, creación, edición básica, eliminación), la interfaz para marcar/desmarcar Favoritos y Archivados, y la lógica para compartir listas (modal de invitación, visualización de colaboradores). Conexión de estas funcionalidades con la API.
- **Back-end:** Diseño e implementación del modelo Mongoose para Listas. Desarrollo de las rutas API REST y controladores para el CRUD de Listas, y los endpoints específicos para la gestión de Favoritos/Archivados (actualizando el modelo User) y el sistema de Compartir (invitar, quitar colaborador, actualizar array de colaboradores en Lista).

Federico (25%):

- **Front-end:** Desarrollo de las páginas de Registro y Login, incluyendo la lógica de validación de formularios y la interacción con la API para autenticación. Implementación de la gestión de sesión (token JWT en localStorage) y la redirección basada en roles. Creación del Panel de Administración (UI para listar, editar rol, eliminar usuarios) y la página de Perfil de Usuario (visualización, cambio de contraseña). Implementación de la lógica de visibilidad condicional de elementos para administradores.
- **Back-end:** Diseño e implementación del modelo Mongoose para Usuarios. Desarrollo de las rutas API REST y controladores para la autenticación (registro, login), gestión de perfiles (obtener, actualizar perfil/contraseña) y las operaciones de administración de usuarios (CRUD completo, cambio de rol). Implementación de la autenticación JWT (generación y verificación de tokens) y middlewares de protección de rutas y autorización por roles. Configuración inicial del servidor Express y conexión a MongoDB. Desarrollo del script Seeder.

Lo que SÍ Funcionó y lo que Quedó Pendiente

Sí Funcionó:

- Autenticación completa: Registro, login (con token JWT), y logout.
- Gestión de Perfil: Visualización de datos y cambio de contraseña funcional.
- CRUD completo de Listas: Crear, ver, editar (título, descripción, campos especiales), eliminar. Incluye marcar/desmarcar como Favorito/Archivado.
- CRUD completo de Ítems: Crear, ver, editar (título, descripción, estado, etiquetas, valores de campos especiales), eliminar.
- CRUD completo de Etiquetas: Crear, ver, eliminar desde su gestor.
- Campos Especiales: Definición a nivel de Lista y asignación/visualización de valores a nivel de Ítem.
- Vistas Personalizadas: Creación, listado, eliminación y aplicación funcional de vistas para filtrar y ordenar ítems.
- Compartir Listas: Funcionalidad MVP para invitar usuarios (por email/username) a colaborar en listas con permisos de 'ver' o 'editar', y quitarles el acceso. Los usuarios invitados pueden ver las listas compartidas.
- Panel de Administración: Visualización de usuarios, cambio de roles, eliminación de usuarios, edición básica de perfil de otros usuarios y visualización de estadísticas básicas.
- Paginado en todos los endpoints de listado del backend.
- Manejo de errores y validaciones robustas en el backend y feedback mejorado en el frontend.
- Despliegue estable y funcional en Render con MongoDB Atlas.

Pendiente:

- **Drag & Drop Avanzado de Ítems**
- **Compartir Listas (UI Avanzada):** Mejorar la interfaz de compartir, quizás con sugerencias de usuarios, y mostrar claramente quién tiene acceso y con qué permiso fuera del modal.
- **Perfil de Usuario Completo:** subida de foto de perfil.

Temas Legales / UX

Banner de Cookies

Se recomienda un banner de cookies. Aunque el uso principal de `localStorage` es para el token de autenticación (esencial para el funcionamiento), si en el futuro se añadieran funcionalidades como recordar preferencias de UI, analíticas de terceros o trackers, el

consentimiento del usuario sería imperativo para cumplir con regulaciones como GDPR o CCPA. Implementarlo desde el inicio fomenta la transparencia.

Políticas de Privacidad

Esencial. La aplicación recolecta y almacena datos personales como emails, nombres, y el contenido de las listas creadas por los usuarios. Una política de privacidad clara debe detallar qué información se recolecta, cómo se usa, cómo se almacena y protege, con quién se podría compartir (si aplica, ej. servicios de terceros), y los derechos del usuario sobre sus datos (acceso, rectificación, eliminación), cumpliendo con normativas vigentes como GDPR.

Términos y Condiciones

Fundamentales para establecer las reglas de uso del servicio. Para "Generador de Listas Etiquetadas", algunos puntos clave a incluir serían:

1. **Uso Aceptable del Servicio:** Prohibir usos indebidos, spam, contenido ilegal, etc.
Definir las responsabilidades del usuario al usar la plataforma.
2. **Propiedad del Contenido del Usuario:** Aclarar que el usuario retiene la propiedad de los datos que ingresa en sus listas, pero otorga a la plataforma los derechos necesarios para operar el servicio (ej. almacenar, mostrar).
3. **Limitación de Responsabilidad:** Establecer hasta qué punto la plataforma es responsable por pérdidas de datos, interrupciones del servicio, o daños indirectos.
Común en servicios SaaS.
4. **Modificaciones al Servicio y Términos:** Reservar el derecho de modificar la aplicación o los términos, y cómo se notificarán dichos cambios a los usuarios.
5. **Terminación de Cuentas:** Especificar las condiciones bajo las cuales una cuenta puede ser suspendida o terminada (ej. por violación de los términos, inactividad prolongada, etc.) y el procedimiento para ello.

Reflexión Individual

Diego

Trabajar en los módulos de Ítems, Etiquetas, Vistas Personalizadas y Campos Especiales fue un reto interesante y muy completo. Aprender a modelar estas entidades en MongoDB, especialmente las relaciones muchos-a-muchos (como Ítem-Etiquetas mediante un array de ObjectIds) y la estructura flexible de `valoresCamposEspeciales`, fue muy formativo. Implementar el paginado eficiente con Mongoose (`skip`, `limit` y `countDocuments`) y la lógica de filtrado avanzado en el backend para las Vistas Personalizadas me ayudó a comprender mejor cómo construir APIs robustas. En el frontend, el desafío principal fue la sincronización del estado, la gestión de los modales para edición y la renderización dinámica de componentes basados en datos de la API. El flujo de definir campos, luego asignar

valores y finalmente verlos aplicados, requirió una cuidadosa coordinación entre el frontend y el backend. Este proyecto reforzó lo visto en clase sobre APIs REST, diseño de modelos NoSQL y la importancia de la interactividad en el frontend, además de la necesidad de un debugging sistemático cuando las cosas no salen como se espera.

Luis

Mi enfoque principal estuvo en el módulo de Listas y la interfaz del Dashboard, incluyendo la gestión de Favoritos/Archivados y la funcionalidad de Compartir. Fue mi primera experiencia integrando de forma tan directa el frontend con un backend RESTful, manejando el estado del usuario y sus listas. Diseñar la interfaz del Dashboard para que fuera intuitiva y permitiera una fácil visualización y acceso a las acciones de las listas fue un aprendizaje importante. La implementación de la lógica de Favoritos/Archivados, actualizando los arrays en el modelo `User`, y luego el sistema de Compartir, con la gestión de colaboradores y permisos, me enseñó mucho sobre cómo manejar relaciones y autorización en el backend. Conectar los botones del frontend para estas acciones, actualizar el `localStorage` y reflejar los cambios en la UI de forma reactiva fue un desafío que me ayudó a entender mejor la importancia de un buen manejo de estado en el cliente. Lo visto en clase sobre JWT y protección de rutas fue crucial para el backend de estas funcionalidades.

Federico

Mi responsabilidad principal fue la autenticación de usuarios, la gestión de roles y el panel de administración. Implementar el registro, login con hash de contraseñas (bcrypt) y la generación/verificación de tokens JWT fue un gran aprendizaje, aplicando directamente los conceptos de seguridad vistos en clase. Diseñar el flujo de autorización con middlewares para proteger rutas y diferenciar entre roles 'user' y 'admin' fue un reto clave. Construir el panel de administración, con la capacidad de listar, editar y eliminar usuarios, y mostrar estadísticas básicas, me dio una buena perspectiva de las funcionalidades backend que soportan la gestión del sistema. El mayor desafío fue asegurar que la autenticación fuera segura y que el control de acceso por roles se aplicara consistentemente en todas las rutas protegidas. Esta experiencia consolidó mi entendimiento de los principios de las API REST y la importancia de un backend bien estructurado y seguro.