📘 INFORME FINAL DE TESTING Y VALIDACIÓN DEL SISTEMA DE TARJETAS SEMANA 3

🧩 1. Objetivo del testing. Validar el correcto funcionamiento del sistema de tarjetas (cards) en un entorno FastAPI + frontend con drag & drop, asegurando:

- Creación de tarjetas
- Lectura de tarjetas
- Actualización (título, estado, list_id, order)
- Eliminación
- Reordenamiento automático
- Movimiento entre columnas
- Persistencia visual en frontend
- Manejo de errores

🧪 2. Testing realizado. A continuación se listan todas las pruebas realizadas, su objetivo y el resultado obtenido.

2.1 Crear tarjeta – POST /cards/
Objetivo: Validar que se puede crear una tarjeta correctamente.
Resultado:

- La tarjeta se crea con title, list_id, status="todo" y order=0.
- El backend responde con código 200 y devuelve la tarjeta creada.

2.2 Leer tarjetas – GET /cards/
Objetivo: Comprobar que el backend devuelve todas las tarjetas existentes.
Resultado:

- El endpoint devuelve un array con todas las tarjetas.
- Los campos id, list_id, status y order se muestran correctamente.

2.3 Actualizar tarjeta – PUT /cards/{id}
Objetivo: Validar que se pueden actualizar los campos de una tarjeta.
Resultado:

- Se actualiza correctamente:
  - ➔ title
  - ➔ status
  - ➔ list_id
  - ➔ order

- El backend responde con código 200 y devuelve la tarjeta actualizada.

2.4 Eliminar tarjeta – DELETE /cards/{id}
Objetivo: Confirmar que una tarjeta puede eliminarse correctamente.
Resultado:

- La tarjeta se elimina de la base de datos.
- El backend responde con "Card deleted".

2.5 Reordenamiento dentro de columnas.
Objetivo: Validar que el backend gestiona el orden de forma automática.
Resultado:

- Al cambiar el order de una tarjeta, el backend recalcula los órdenes de todas las tarjetas de la lista.
- No quedan duplicados.
- No quedan huecos.
- El orden final es siempre una secuencia limpia: 0, 1, 2, ….

2.6 Movimiento entre columnas (list_id).
Objetivo: Verificar que arrastrar una tarjeta entre columnas actualiza su list_id.
Resultado:

- El list_id cambia correctamente.
- El frontend refleja el movimiento sin errores.

2.7 Simulación de errores.
Objetivo: Validar el comportamiento del sistema ante situaciones incorrectas.
Errores probados:

- Falta de token → 401
- Card no encontrada → 404
- Movimiento a tablero ajeno → 403

Resultado:

- ➔ El backend responde con los códigos esperados.
- ➔ No se producen fallos inesperados.

2.8 Revisión en frontend.
Objetivo: Validar la experiencia de usuario.
Resultado:

- El estilo visual es correcto.
- El movimiento es fluido.
- No hay saltos entre columnas.
- Tras refrescar la página, el orden y la columna se mantienen.

🐞 3. Errores detectados y soluciones.

3.1 Duplicados de order.
Problema:
➔ Dos tarjetas podían quedar con el mismo order.
Solución aplicada:
➔ Se implementó lógica de reordenamiento automático en update_card.
Estado:✔️ Resuelto

3.2 Orden incorrecto al mover tarjetas entre listas.
Problema:
➔ Al cambiar list_id, el order no se recalculaba en la nueva lista.
Solución recomendada:
➔ Aplicar reordenamiento automático también cuando cambia list_id.
Estado: 🔧 Pendiente (mejora técnica)

3.3 Huecos en el orden tras borrar una tarjeta.
Problema:
➔ Al eliminar una tarjeta intermedia, podían quedar órdenes como 0, 1, 3.
Solución recomendada:
➔ Recalcular los órdenes de la lista en el endpoint DELETE.
Estado: 🔧 Pendiente (mejora técnica)

🚀 4. Mejoras recomendadas.

Mejoras:
1. Reordenar tras borrar.
2. Reordenar al cambiar list_id entre listas.
3. Validar order negativo o fuera de rango.
4. Tests automatizados con pytest.
5. Feedback visual en frontend.
6. Accesibilidad drag & drop.

Descripción:
1. Evitar huecos en el orden.
2. Evitar duplicados en la nueva lista.
3. Seguridad y consistencia.
4. Garantizar estabilidad.
5. Mostrar confirmación de movimiento.
6. Mejorar UX.

Rol:

1. Backend.
2. Backend.
3. Backend.
4. Backend.
5. Frontend.
6. Frontend.

 5. Conclusión general.

El sistema ha sido probado exhaustivamente y cumple con los requisitos funcionales:

- ➔ Crear, leer, actualizar y borrar tarjetas.
- ➔ Mover entre columnas.
- ➔ Reordenar dentro de columnas.
- ➔ Persistencia visual.
- ➔ Manejo de errores.
- ➔ Fluidez en frontend.

📑 5. Conclusión general

El sistema ha sido probado exhaustivamente y cumple con los requisitos funcionales:

Crear, leer, actualizar y borrar tarjetas

Mover entre columnas

Reordenar dentro de columnas

Persistencia visual

Manejo de errores

Fluidez en frontend