



Tablero Kanban configurable con almacenamiento web

El objetivo de esta práctica es desarrollar una aplicación web en JavaScript que permita crear y gestionar un **tablero Kanban configurable**, utilizando **almacenamiento web**, **manipulación del DOM**, **eventos** y **drag & drop**.

Con esta práctica se pretende consolidar los conocimientos del tema de **almacenamiento web en el lado cliente**, integrándolos con contenidos vistos anteriormente como el DOM y los eventos.

Descripción general

La aplicación deberá comenzar mostrando un **formulario de configuración inicial**, a través del cual el usuario definirá las características de su tablero Kanban.

A partir de esta configuración, la aplicación generará dinámicamente el tablero y permitirá gestionar las tareas de forma persistente mediante localStorage.

El tablero deberá conservar su estado, aunque se recargue la página o se cierre el navegador.

Configuración inicial del tablero

La **configuración inicial del tablero deberá realizarse obligatoriamente mediante un formulario HTML**, gestionado y generado dinámicamente con JavaScript.

El formulario deberá permitir al usuario indicar:

1. El **número de columnas** del tablero.
2. Para cada columna:
 - El **nombre personalizado** de la columna.
 - El **límite máximo de tareas** permitidas en dicha columna.

Requisitos del formulario:

- El formulario debe ser **dinámico**:
 - Al indicar el número de columnas, deben aparecer automáticamente los campos necesarios para configurar cada columna.
- No debe existir HTML fijo para las columnas del tablero.
- Los campos del formulario deben validarse (por ejemplo, no permitir nombres vacíos o límites inválidos).
- Al enviar el formulario:



- Se debe crear la estructura del tablero utilizando el DOM.
- La configuración introducida debe guardarse en localStorage.
-

Comportamiento al recargar la página:

- Si existe una configuración guardada en localStorage:
 - El formulario **no debe mostrarse**.
 - El tablero debe generarse automáticamente a partir de los datos almacenados.
- El formulario solo debe mostrarse si **no existe configuración previa** o si el usuario decide reiniciar el tablero.

Gestión de tareas en el tablero

Añadir tareas

Cada columna deberá permitir añadir tareas mediante un campo de texto y un botón.

Requisitos:

- No se permitirán tareas vacías.
- No se permitirá añadir tareas si se ha alcanzado el límite de la columna.
- Las tareas deberán guardarse en localStorage.

Mostrar tareas

Las tareas deberán mostrarse dinámicamente en la columna correspondiente utilizando el DOM.

Cada tarea deberá:

- Mostrar su contenido
- Poder interactuar con el usuario (arrastrar, eliminar, etc.)

Eliminar tareas

Cada tarea deberá poder eliminarse mediante una acción del usuario (por ejemplo, un botón o doble clic).

Al eliminar una tarea:

- Se eliminará del DOM
- Se eliminará de la estructura de datos
- Se actualizará localStorage

Drag & drop de tareas

Las tareas deberán poder moverse entre columnas mediante **drag & drop**, reutilizando los conocimientos vistos en el tema anterior.

Requisitos:

- Solo se podrán arrastrar tareas (no columnas).
- Al mover una tarea:
 - Se deberá actualizar la estructura de datos



- Se deberá actualizar el DOM
 - Se deberán guardar los cambios en localStorage
- No se permitirá mover tareas a una columna que haya alcanzado su límite máximo.

Persistencia de datos

La aplicación deberá utilizar localStorage para almacenar:

- La configuración del tablero
- Las columnas
- Las tareas de cada columna

Al cargar la página:

- Si existen datos almacenados, el tablero deberá reconstruirse automáticamente.
- El usuario deberá ver el tablero exactamente en el mismo estado en el que lo dejó.