# Task Manager Pro

## Descripción General

**Task Manager Pro** es una aplicación web sencilla que permite a los usuarios **crear, listar, filtrar y gestionar tareas**, aplicando **patrones de diseño en JavaScript**.

El objetivo principal es aprender a **estructurar un proyecto realista**, aplicando al menos **un patrón de cada tipo**:

- **Creacional**
- **Estructural**
- **De Comportamiento**

## Objetivos del Proyecto

1. Aplicar los conocimientos de **HTML, CSS y JavaScript** en una aplicación funcional.
2. Implementar **al menos un patrón de diseño de cada categoría**.
3. Diseñar una estructura de código modular y mantenible.
4. Practicar el pensamiento de diseño y la elección de patrones adecuados a un problema real.
5. Mejorar la presentación del proyecto con un diseño limpio y responsive.

## Requisitos Técnicos

- El proyecto debe funcionar **sin frameworks** (solo JS, HTML y CSS puro).
- El código debe estar organizado en carpetas (`patterns/`, `ui/`, `models/`, etc.).

- Los patrones deben estar **comentados y justificados** en el código o en el README.
- El diseño debe usar **Flexbox o GridCSS** para organizar la interfaz.
- No se permite usar librerías externas salvo para íconos o fuentes (ej. FontAwesome, Google Fonts).

## Funcionalidades Mínimas

El usuario debe poder:

1. **Añadir una tarea** con título, descripción y prioridad.
2. **Marcar tareas como completadas.**
3. **Eliminar tareas.**
4. **Filtrar tareas** (por estado o prioridad).
5. **Guardar tareas** en `localStorage` para mantener los datos entre recargas.

## Modelo de Datos: Task

Cada tarea debe representarse con un objeto que tenga **al menos las siguientes propiedades**:

| Propiedad | Tipo | Descripción |
|---|---|---|
| id | number | Identificador único (por ejemplo Date.now()). |
| title | string | Título corto de la tarea. |
| description | string | Descripción o detalles de la tarea. |
| priority | string | Puede ser 'low', 'normal' o 'high'. |
| done | boolean | Indica si la tarea está completada. |
| createdAt | Date | Fecha de creación de la tarea. |

# Estructura de Carpetas Recomendada

```
task-manager-pro/
│
├── index.html
├── style.css
├── README.md
└── js/
    ├── main.js
    ├── models/
    │   └── task.js
    ├── patterns/
    │   ├── factory.js
    │   ├── decorator.js
    │   └── observer.js
    └── ui/
        ├── domFacade.js
        └── taskListView.js
```

# Patrones de Diseño a Usar

Mínimo un patrón por tipo.

## Creacional — *Factory Pattern o Singleton*

- **Factory Pattern:** para crear tareas de forma centralizada.
- **Singleton:** para mantener una única instancia de configuración o almacenamiento (por ejemplo, un gestor de tareas).

*Ejemplo de uso:*
Una clase `TaskFactory` que reciba los datos y devuelva un objeto `Task` con propiedades validadas.

## Estructural — *Decorator o Facade*

- **Decorator:** para añadir comportamientos o propiedades adicionales a las tareas (por ejemplo, convertir una tarea en “Urgente”).
- **Facade:** para simplificar la manipulación del DOM con funciones como `addTaskToList()` o `renderTasks()`.

*Ejemplo de uso:*
Una clase o módulo `DOMFacade` que oculte los detalles del DOM y exponga métodos simples como:

```
DOMFacade.renderTasks(taskArray);
DOMFacade.clearInputs();
```

### De Comportamiento — *Observer, Command o Strategy*

- **Observer:** para actualizar la interfaz cada vez que cambie la lista de tareas.
- **Command:** para implementar acciones (añadir, eliminar, deshacer).
- **Strategy:** para aplicar distintos filtros (por prioridad, estado, etc.).

## Requisitos de Estilo (CSS)

Tu diseño debe ser **limpio, responsive y claro**:

- Usa **Flexbox** o **GridCSS** para dividir la interfaz.
- Mínimo, debe tener **dos secciones**:
  - `#task-form`: formulario de creación de tareas.
  - `#task-list`: listado de tareas.

# README.md

Debe incluir:

1. **Descripción breve** del proyecto.
2. **Explicación de los patrones utilizados** (qué patrón, por qué, y dónde se usa).
3. **Instrucciones** para ejecutar el proyecto.
4. (Opcional) Capturas o GIF de la app en funcionamiento.

# Evaluación

| Criterio | Puntos |
|---|---|
| Aplicación funcional y sin errores | 4 |
| Uso correcto de 1 patrón **creacional** | 2 |
| Uso correcto de 1 patrón **estructural** | 2 |
| Uso correcto de 1 patrón **de comportamiento** | 2 |
| Documentación clara (README) y código legible | +1 extra |