Práctica Flutter – Aplicación con API REST, Navegación y Control de Estado (CRUD)

Objetivo

Desarrollar una aplicación móvil en **Flutter** que consuma datos desde una API REST pública (o propia). La aplicación debe implementar navegación entre pantallas y un gestor de estado que permita cubrir el ciclo completo de un CRUD: crear, leer, actualizar y eliminar registros.

Descripción

Debes diseñar una aplicación Flutter conectada a una API REST.
La aplicación debe incluir, como mínimo, las siguientes funcionalidades:

- **Pantalla de lista:** muestra los registros obtenidos de la API.
- **Pantalla de detalle:** al seleccionar un registro, se despliega su información completa.
- **Pantalla de formulario:** permite crear o editar un registro.
- **Acción de eliminación:** desde la lista o el detalle, se debe poder eliminar un registro.

El tipo de datos queda a tu elección (usuarios, productos, publicaciones, tareas, etc.).

Requisitos funcionales

Navegación

- La aplicación debe contar con al menos **tres pantallas** (lista, detalle y formulario).

Consumo de API REST

- Obtener registros (**GET**).
- Crear registros (**POST**).
- Editar registros (**PUT/PATCH**).
- Eliminar registros (**DELETE**).

Control de estado

- Manejo de **estados de carga, éxito y error** en todas las operaciones.
- Actualización automática de la interfaz cuando se agreguen, editen o eliminen registros.

Validación de formularios

- El formulario debe incluir validaciones básicas (ejemplo: que un campo obligatorio no esté vacío).

Organización del código en capas

- **models** → clases de datos.
- **services** → comunicación con la API.
- **providers/controllers** → gestión del estado.
- **screens** → pantallas y widgets principales.

Interfaz de usuario

- La aplicación debe contar con una **interfaz atractiva, clara y fácil de usar**.
- La interfaz debe ser **responsive**, adaptándose correctamente a diferentes tamaños de pantalla y orientaciones.

Funcionalidades requeridas

- Incorporar un **buscador o filtro** en la lista de registros.
- Implementar **paginación** o **scroll infinito** para optimizar la carga de datos.

APIs sugeridas

Puedes usar cualquiera de las siguientes **APIs públicas gratuitas**, elegir otra API pública de tu preferencia o incluso desarrollar una propia:

- JSONPlaceholder → Posts, comentarios, usuarios, tareas.
- Reqres API → Información de usuarios.
- DummyJSON → Productos de tienda en línea.
- Fake Store API → Productos de tienda en línea.

Entregables

- Código fuente en un repositorio (GitHub, GitLab, u otro).
- Archivo **README.md** que incluya:
    - Breve explicación de la API seleccionada.
    - Link de video corto que muestre el funcionamiento de la aplicación.