

Práctica: Aplicación web en Laravel con autenticación, CRUD y traducciones

Objetivo general

Desarrollar una aplicación web utilizando **Laravel** que implemente una estructura basada en **componentes Blade**, autenticación de usuarios, gestión de completa de un recurso (CRUD), propongo **proyectos**, pero se puede implementar otro concepto. Traducciones y siempre usando git para control de versiones y ver evolución del proyecto.

Requisitos:

1. Estructura y diseño

- El diseño de la aplicación debe realizarse **usando componentes Blade HTML**.
 - La aplicación debe incluir, los siguientes componentes (según el diseño de las pantallas, se podría plantear de otra forma). El **layout** sí es obligatorio:
 - ✓ *layout principal*
 - ✓ *header*
 - ✓ *nav*
 - ✓ *footer*
 - Según el diseño elegido, se podrán añadir otros componentes adicionales si se considera necesario.
-

2. Página principal

- La página principal será **main.php**(o vista equivalente definida como principal).
 - **Si el usuario NO está autenticado:**
 - Debe mostrarse algún tipo de **hero** o sección de bienvenida.
 - Deben aparecer las opciones de **register** y **login**.
 - **Si el usuario está autenticado:**
 - Debe mostrarse el **nombre del usuario** y la opción de **logout**.
 - Deben mostrarse **tarjetas (cards)** con las distintas opciones disponibles en la aplicación.
-

3. Autenticación

- El sistema debe permitir:
 - ✓ Registro de usuarios

- ✓ Inicio de sesión
 - ✓ Cierre de sesión
 - La interfaz debe adaptarse correctamente según el estado de autenticación del usuario.
-

4. Gestión de proyectos

- Debe existir una sección de **Proyectos** (o recurso que consideres), accesible desde la aplicación.
 - Los proyectos deben cargarse inicialmente mediante **seeders**.
 - El acceso a esta sección debe estar claramente integrado en la navegación.
-

5. Gestión de alumnos

- Debe existir una **tarjeta específica** para la **gestión de alumnos**.
 - La gestión de alumnos debe realizarse mediante una **tabla con operaciones CRUD**:
 - ✓ Crear
 - ✓ Leer
 - ✓ Actualizar
 - ✓ Eliminar
 - La tabla debe cumplir los siguientes requisitos:
 - ✓ Estar **paginada**
 - ✓ Mostrar un **mensaje de confirmación** tras cada acción realizada (crear, editar, eliminar)
 - ✓ La eliminación debe solicitar **confirmación mediante SweetAlert** (o librería similar)
 - En todas las vistas del CRUD debe existir siempre la opción de:
 - ✓ **Cancelar**
 - ✓ **Volver** a la vista anterior
-

6. Traducciones

- La aplicación debe estar traducida a **tres idiomas**:
 - Castellano
 - Inglés
 - Francés
 - Debe existir un **selector de idioma**, que:
 - Muestre el idioma actualmente seleccionado
 - Mantenga el idioma elegido al navegar por la aplicación
-

7. Documentación

- El proyecto debe incluir un archivo **README.md** con:
 - Explicación general del proyecto
 - Requisitos para su ejecución
 - Pasos para la instalación
 - Breve explicación de la estructura y funcionalidades principales
-

8. Control de versiones (Git)

- Nada más crear el proyecto se debe realizar un **git clone** del repositorio.
 - El uso de **Git es obligatorio** durante todo el desarrollo.
 - **Cada día de clase debe realizarse al menos un push**, reflejando el trabajo realizado en esa sesión.
-

Otros Requisitos adicionales

Si se desea ampliar la práctica o mejorar la calificación, se pueden implementar algunas de las siguientes opciones:

- Protección de rutas mediante **middleware** (acceso solo a usuarios autenticados).
 - Validación de formularios con mensajes traducidos.
 - Uso de tailwindcss y opcionalmente librerías de componentes como **daisy** para mejorar el diseño .
 - Inclusión de logs o control de errores básico.
 - Uso correcto de commits con mensajes claros y descriptivos.
-

Entrega

El proyecto se entregará mediante el **repositorio Git**, debidamente actualizado y con el historial de commits que evidencie el trabajo continuo durante las sesiones de clase.