Se trata de desarrollar una aplicación que integre los componentes:

Entrega

- Código del proyecto.

- Video explicando los detalles del código y demostrando los conocimientos adquiridos. Máximo 12 minutos.

Backend

1. **API REST** autenticada con JWT.
2. **Persistencia.** Los datos se almacenarán en 2 sistemas de persistencia, uno en memoria, y otro en una base de datos MySQL, se deberá desacoplar el sistema de persistencia aplicando el patrón open/close.
    a. Deberéis descargar de kaggel, un fichero dataset del tema que más os llame la atención.
    b. Se cargarán los datos del fichero descargado de kaggel, en una BBDD MySQL, y almacenando los datos en memoria usando LINQ.
    c. La configuración de la aplicación se almacenará en un fichero App.config, y cuando arranque la aplicación se leerá de aquí la configuración (cadena de conexión, etc). En este fichero de configuración seleccionaremos el sistema de persistencia inicial, datos en BBDD o datos en memoria.
3. **Protocolo MCP e integración con la IA**. Se deberá desarrollar el cliente MCP y el servidor MCP, para realizar consultas sobre la base de datos utilizando el lenguaje natural.
    a. Deberá tener 2 enrutadores,
        i. Otro enrutador manual con reglas (se ejecuta primero)
        ii. Otro con LLM (si no encuentra reglas enruta el LLM)

Frontend

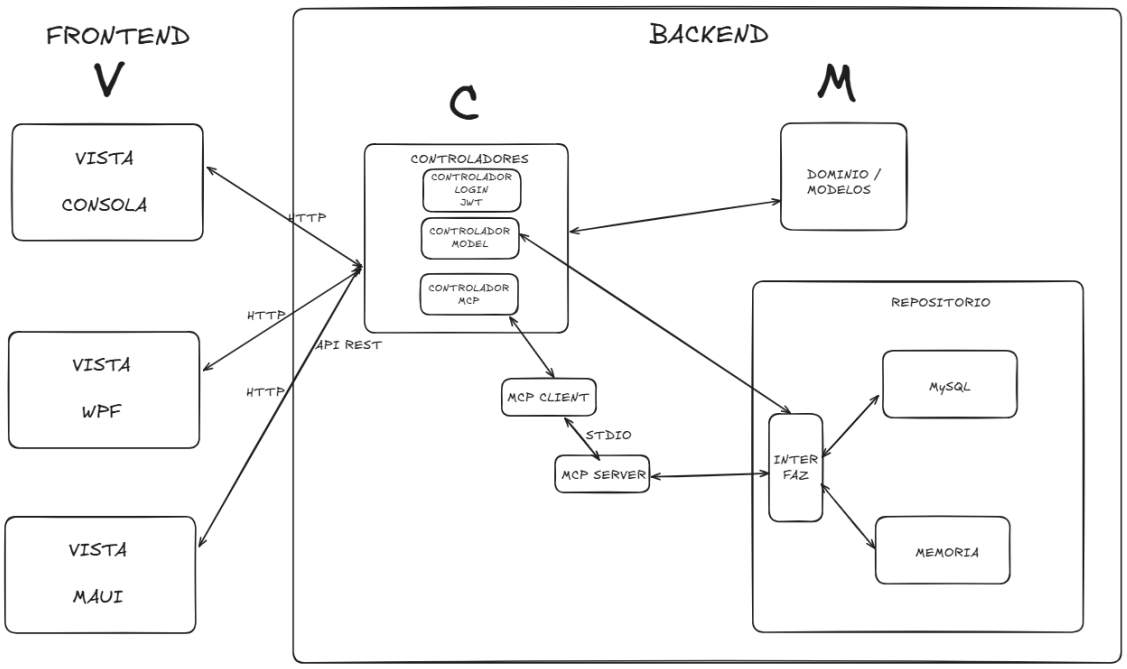
4. **Interfaces graficas**: Crearan 3 vistas, cada una en un proyecto.
    a. **Consola**.
        i. Autenticación con la API y obtención JWT
        ii. Selección del sistema de persistencia.
        iii. Carga de datos de kaggel al sistema de persistencia
        iv. Operaciones CRUD
    b. **WPF**.
        i. Interfaz MDI
        ii. Pantalla de login, que autentica con la API y obtiene JWT
        iii. Pantalla de configuración, con un botón para importar los datos del fichero de kaggel a memoria y a la BBDD MySQL

iv. Pantalla de Master (muestra todos los datos) y podemos eliminar un dato.
v. Pantalla Detail (permite insertar y editar datos)
vi. Pantalla de MCP, donde podemos interactuar con nuestro sistema mediante lenguaje natural

c. .NET MAUI

i. Menú de navegación
ii. Pantalla de login, que autentica con la API y obtiene JWT
iii. Pantalla de Master (muestra todos los datos) y podemos eliminar un dato.
iv. Pantalla Detail (permite insertar y editar datos)
v. Pantalla de MCP, donde podemos interactuar con nuestro sistema mediante lenguaje natural

5. **Rendimiento**. Programación asíncrona, se deberá optimizar el rendimiento del sistema, evitando que la interfaz quede bloqueada, haciendo uso de async y await. También se puede optimizar el rendimiento usando Lazy Loading en NET MAUI.



Rúbrica

	Puntos
1. API REST autenticada con JWT	1
2. Persistencia - Carga fichero en Memoria (LINQ) - Carga fichero en BBDD - Lee fichero config - Aplica Open/Close	2
3. Protocolo MCP - Enrutador reglas - Enrutador LLM	2
4. Interfaces gráficas - Consola - WPF - MAUI - Validación de campos	4
5. Rendimiento	1