## Objetivos (punto 4)

1. Desarrollar una aplicación web funcional que permita a los usuarios registrarse, iniciar sesión y gestionar su cuenta de manera segura y sencilla, utilizando Laravel como framework backend y Astro como tecnología frontend.
2. Implementar un sistema para la publicación de recetas de cocina, en el que los usuarios puedan crear, editar y eliminar sus propias recetas incluyendo título, descripción, ingredientes, pasos de preparación y una imagen ilustrativa.
3. Diseñar una interfaz web intuitiva, atractiva y responsive con Astro, garantizando una experiencia de usuario óptima en dispositivos móviles y de escritorio.
4. Definir y gestionar una base de datos relacional en MySQL, destinada a almacenar de forma eficiente y segura la información de usuarios, recetas y demás entidades necesarias para el funcionamiento de la aplicación.
5. Incorporar un sistema de búsqueda y filtrado de recetas, que permita a los usuarios encontrar contenidos relevantes según nombre, ingredientes o categoría.
6. Aplicar mecanismos de seguridad en el desarrollo de la aplicación, tales como validación de formularios, protección contra ataques comunes (como inyecciones SQL) y almacenamiento seguro de contraseñas.
7. Aplicar buenas prácticas de desarrollo web tanto en el frontend como en el backend, incluyendo el uso de control de versiones con Git, documentación del código y separación de responsabilidades entre componentes.
8. Realizar pruebas funcionales y de usuario, con el objetivo de verificar el correcto funcionamiento de todas las funcionalidades implementadas y asegurar una buena experiencia de uso.

## Requisitos funcionales y no funcionales (punto 5.1.)

**Requisitos funcionales**

Los requisitos funcionales describen las funcionalidades que el sistema debe ofrecer al usuario final. Para esta aplicación web de recetas de cocina, se han identificado los siguientes:

Registro de usuarios: La aplicación debe permitir a nuevos usuarios registrarse mediante un formulario con nombre, correo electrónico y contraseña.

Inicio de sesión y cierre de sesión: Los usuarios registrados deben poder autenticarse para acceder a funcionalidades personalizadas, así como cerrar sesión de forma segura.

Gestión de perfil de usuario: Cada usuario debe poder visualizar y editar la información de su perfil, incluyendo su nombre, foto (opcional) y contraseña.

Publicación de recetas: Los usuarios autenticados podrán crear recetas introduciendo título, descripción, ingredientes, pasos de preparación e imagen.

Edición y eliminación de recetas: Los usuarios podrán editar o eliminar únicamente las recetas que hayan creado.

Visualización de recetas: Cualquier visitante podrá visualizar todas las recetas disponibles, incluyendo su contenido completo y la información del autor.

Buscador de recetas: La aplicación debe ofrecer un sistema de búsqueda por palabras clave (título, ingredientes, etc.).

Filtrado de recetas: El usuario podrá filtrar recetas por categorías predefinidas (por ejemplo: entrantes, platos principales, postres...).

Sistema de comentarios: Los usuarios autenticados podrán comentar en las recetas de otros usuarios.

Marcado de recetas favoritas: Los usuarios podrán marcar recetas como favoritas para consultarlas posteriormente.

**Requisitos no funcionales**

Los requisitos no funcionales definen características de calidad y restricciones técnicas que debe cumplir el sistema:

Seguridad: Las contraseñas de los usuarios deben almacenarse cifradas. Se deben validar las entradas del usuario para evitar ataques como inyección SQL. Se debe implementar protección contra ataques CSRF en los formularios.

Usabilidad: La interfaz debe ser intuitiva, fácil de usar y accesible para usuarios sin conocimientos técnicos. Se debe garantizar una experiencia fluida tanto en dispositivos móviles como en ordenadores.

Rendimiento: Las páginas deben cargarse en un tiempo razonable (<2 segundos en condiciones normales). El sistema debe ser capaz de manejar múltiples usuarios y recetas sin disminuir significativamente su rendimiento.

Compatibilidad: La aplicación debe ser compatible con los navegadores web más utilizados (Chrome, Firefox, Edge, Safari).

Mantenibilidad: El código debe estar estructurado de forma clara y modular, utilizando buenas prácticas de programación para facilitar su mantenimiento y ampliación.

Escalabilidad: La arquitectura basada en frontend y backend separados (Astro + Laravel) debe permitir futuras ampliaciones (por ejemplo, integración con APIs, app móvil, etc.).

Portabilidad: El proyecto debe poder instalarse y desplegarse fácilmente en otros entornos utilizando herramientas comunes como Docker o entornos LAMP/WAMP.

Disponibilidad: La aplicación debe estar disponible las 24 horas del día, en caso de desplegarse en un entorno de producción.

## Diagrama de Entidad-Relación (punto 5.4).

En el caso de que el proyecto utilice una base de datos relacional, incluir el diagrama de entidad-relación utilizando algunas de las herramientas disponibles en el mercado.

## Diagrama de Casos de Usos (punto 5.5).

Incluir el diagrama para representar el funcionamiento del software desarrollado.

Diagrama pocho pero da ideas

Diagrama

El contenido generado por IA puede ser incorrecto.