\\*\*Propósito:\*\*

La aplicación tiene como objetivo centralizar la gestión de membresías, el control de acceso, la facturación y otros aspectos operativos de un gimnasio de pesas con múltiples sucursales, mejorando la eficiencia y la experiencia del usuario. La aplicación se desarrollará con laravel php + mysql+liveware + tailwincss y lo que consideres necesario

\*\*Características Clave:\*\*

\* \*\*Gestión de Membresías:\*\*

\* Registro de nuevos miembros con datos personales (nombre, apellido, dirección, teléfono, email, fecha de nacimiento), información de contacto, foto (opcional) y selección de tipo de membresía (mensual, trimestral, anual, etc.). Los datos se almacenarán en la base de datos MySQL a través del ORM

\* Gestión de membresías existentes: modificación de datos, renovación (automática o manual), cancelación, suspensión temporal. Se implementarán vistas de Django para facilitar la creación, lectura, actualización y eliminación (CRUD) de membresías.

\* Historial de membresías por miembro: Se utilizarán modelos de Django para representar el historial de cada membresía, incluyendo fechas de inicio y fin, tipo de membresía, y estado.

\* Generación de informes sobre el estado de las membresías (activas, vencidas, por vencer). Se usarán las capacidades de generación de informes o librerías para crear informes PDF o CSV.

\* \*\*Control de Acceso:\*\*

\* Integración con dispositivos de control de acceso: de manera predeterminada seria teclado numérico con la opción de configurar lector biométrico (huella digital. Facial opcional) en cada sucursal. Deberá tener una interfass para configurar y para comunicarse con los dispositivos de control de acceso.

\* Registro de los dispositivos de control de acceso por sucursal: Se almacenará la información de los dispositivos (ID, tipo, sucursal) en la base de datos.

\* Asignación de métodos de acceso (código numérico o huella digital) a cada miembro. Se utilizarán campos en el modelo de Miembro para almacenar el código numérico (hasheado por seguridad) o la plantilla de la huella digital (si es posible, almacenada de forma segura).

\* Validación del acceso: al detectar un por alguno de los métodos de acceso, si el miembro tiene una membresía activa y permitirá o denegará el acceso al molinete/puerta. Se implementará lógica de negocio para validar la membresía y registrar el evento de acceso.

\* Registro de eventos de acceso: fecha, hora, miembro, sucursal, resultado (acceso permitido/denegado). Se creará un modelo para registrar cada evento de acceso, facilitando la auditoría y la generación de informes.

\* Generación de informes de acceso por miembro, sucursal y rango de fechas. Se utilizarán las capacidades de consulta de ORM para generar informes personalizados.

\* \*\*Facturación y Pagos:\*\*

\* Generación automática de facturas por membresía. Se utilizarán tareas asíncronas para generar facturas automáticamente al renovar una membresía.

\* Registro de pagos (efectivo, tarjeta, transferencia, etc.). Se creará un modelo de Pago para registrar cada pago, con campos para el método de pago, la fecha, el monto y la membresía asociada.

\* Gestión de pagos atrasados. Se implementará lógica para identificar y notificar pagos atrasados.

\* Generación de informes de ingresos por sucursal, tipo de membresía y período. Se utilizarán las capacidades de consulta de ORM para generar informes de ingresos.

\* \*\*Gestión de Productos (Opcional):\*\*

\* Control de inventario de productos a la venta (suplementos, ropa, etc.). Se utilizarán modelos de Django para representar productos, categorías y el inventario.

\* Registro de ventas. Se creará un modelo de Venta para registrar cada venta, con detalles de los productos vendidos, el monto y el cliente.

\* Generación de informes de ventas. Se utilizarán las capacidades de consulta de ORM para generar informes de ventas.

\* \*\*Gestión de Usuarios del Sistema:\*\*

\* Creación de usuarios con diferentes roles y permisos (administrador, recepcionista, instructor). Se utilizará el sistema de autenticación y autorización de Django para gestionar usuarios y permisos.

\* Registro de la actividad de los usuarios en el sistema. Se puede utilizar un middleware para registrar la actividad de los usuarios (quién hizo qué y cuándo).

\*\*Interfaz de Usuario:\*\*

\* Debe ser intuitiva y fácil de usar, incluso para usuarios con poca experiencia en software de gestión. con un diseño responsive que se adapte a diferentes dispositivos.

\* Debe tener un diseño visualmente atractivo y profesional. Se puede utilizar un framework CSS para facilitar el diseño de la interfaz.

\* Debe ser adaptable a diferentes tamaños de pantalla. Se utilizarán técnicas de diseño responsive para asegurar que la aplicación se vea bien en dispositivos de escritorio, tablets y teléfonos.

\* Considerar la posibilidad de utilizar un sistema de notificaciones para alertar sobre membresías por vencer, pagos atrasados, etc.

Configurable con logos , formatos, nombre del gym imagnes, títulos temas de colores para el aplicativo reportes

\*\*Control de Acceso Detallado:\*\*

\* El sistema debe permitir configurar diferentes reglas de acceso por sucursal (ej., horario de acceso permitido, días de la semana).

\* Debe haber un panel de control para monitorear el estado de los dispositivos de control de acceso (conectados, desconectados, con problemas). Se implementará un panel de control para mostrar el estado de los dispositivos y permitir la configuración.

\* En caso de fallo del dispositivo biométrico, debe haber un método de acceso alternativo (ej., ingreso manual del código por parte del recepcionista). Se implementará una interfaz en Django para que el recepcionista pueda ingresar manualmente el código del miembro en caso de fallo del dispositivo.

Generación dinámica de códigos QR temporales

\*\*Restricciones/Preferencias Técnicas:\*\*

\*\*Resultado Deseado:\*\*

\* Modelos (diagrama de clases).

pantallas principales (con énfasis en la interacción con la API de control de acceso):

\* Panel de control principal (con resumen de la información clave).

\* Pantalla de gestión de membresías.

\* Pantalla de registro de acceso (interfaz para el recepcionista).

\* Pantalla de gestión de clases.

\* Pantalla de facturación y pagos.

\* Guía de estilo con la paleta de colores, la tipografía y los elementos visuales principales (adaptada Tailwind CSS).

Dashboard

Comunicación omnicanal (app móvil, WhatsApp, web)

Genera todo lo pendiente y pásamelo aquí con el txt de instrucciones que sean necesarios para realizar las pruebas de manera local

El proyecto lo deseo lo implementare el wamp en Windows

Primeramente, de manera local las configuraciones e instalaciones necesarias para el proyecto

Una vez terminada la configuraciones lo subiré a git para que lo controles

Luego la creación de los modelos y migraciones

Pantalla principal menú todo le layout lo probare y alsare al git para verificar

no quiero que lo pruebes, solo todas las configuración y código del sistenma, asi también un instructivo de como crear el proyecto en wamp paso a paso e instalar todo lo necesario para su funcionamiento. Ten encuenta todo de las versiones actuales de laravel y todolo lo demass

https://github.com/edgardodanielmeza/proyectogimnasio voy a subir solamente el proyecto basi creado me gustaría que sea todo en español

composer create-project laravel/laravel gimnasio