**Aplicación de Principios SOLID en una Funcionalidad Específica:**

1. **Principio de Responsabilidad Única**:

Cada clase en el diagrama está diseñada para manejar una sola responsabilidad, lo que mejora la claridad y mantenibilidad del código. Por ejemplo, la clase Reserva se encarga exclusivamente de gestionar las reservas de clases, como asignar un Entrenador y establecer el Estado de la reserva. Por otro lado, la clase Finanzas se ocupa únicamente de los aspectos financieros, como registrar ingresos y generar reportes. Esta separación de responsabilidades asegura que cada clase cumpla su función sin interferir con las demás.

2. **Principio Abierto/Cerrado**:

El sistema está diseñado para permitir la extensión sin modificar el código existente. Por ejemplo, si se quiere añadir un nuevo tipo de Clase, se puede crear una nueva clase que herede de la clase base Clase. Esto permite que la nueva clase tenga características adicionales específicas sin alterar la funcionalidad de la clase original. Del mismo modo, en el manejo de Membresía, se puede apreciar este principio con la MembresíaVIP

3. **Principio de Sustitución de Liskov**:

Este principio asegura que los objetos de una subclase puedan reemplazar a los de la clase base sin afectar la funcionalidad del sistema. Por ejemplo, viendo la clase MembresíaVIP que extiende de la clase Membresía, esta subclase es capaz de funcionar en cualquier contexto donde se utilice la clase base, como en el manejo de Pagos o en la verificación de Estado. Esto garantiza que el sistema se mantenga robusto y que las nuevas funcionalidades no introduzcan errores.