Sistema de Gestión de Gimnasio v2

Versión: 2.0

Randsiwim Lopez O

Fecha: 12/18/2024

Decisiones de Diseño

1. Estructura del Proyecto

El proyecto está estructurado utilizando el patrón MVC (Modelo-Vista-Controlador) para separar la lógica de negocio, las interfaces de usuario y el acceso a datos.

2. Almacenamiento y Exportación

Se decidió implementar un sistema dual para la persistencia de datos:

Base de datos SQL Server para almacenamiento principal.

Exportación de datos clave (como facturas) a formato CSV para análisis externo.

3. Roles de Usuarios

El sistema admite tres roles principales:

Administrador: Gestiona usuarios, clases y facturas.

Entrenador: Visualiza y gestiona sus clases asignadas.

Cliente: Consulta su progreso, reservas y facturación.

4. Validaciones

Se implementaron validaciones tanto en cliente como en servidor para garantizar la integridad de los datos.

5. Estilo y Usabilidad

Se emplearon herramientas como Bootstrap para lograr un diseño responsivo y consistente.

Desarrollo

Configuración del Entorno

Requisitos Previos:

.NET Core SDK

SQL Server

Visual Studio

Instalación de Dependencias:

Configura las conexiones en appsettings.json para SQL Server.

Ejecución del Programa

Inicia el servidor :

dotnet run

Accede a la aplicación

CRUD Implementados

Se implementaron los siguientes CRUDs:

Usuarios: Gestión de roles, credenciales y datos personales.

Clases: Creación, edición, eliminación y listado de clases.

Facturas: Gestión de pagos y exportación a CSV.

Progreso de Usuarios: Registro de avances individuales.

Reservas: Manejo de asignaciones a clases.

Resolución de Elementos Específicos

Gestión de Clases:

Se implementó un CRUD completo con validaciones y sincronización en la base de datos.

Exportación de Facturas:

Las facturas se almacenan y exportan a un archivo CSV en tiempo real.

Visualización por Roles:

Cada usuario tiene vistas específicas basadas en su rol, utilizando autorizaciones.

Integridad de Datos:

Validaciones adicionales para evitar datos nulos o inconsistentes.

Testing:

Se obtiene cobertura de en la capa lógica del controlador mediante pruebas unitarias.

Análisis de Resultados

Eficiencia:

La implementación de CSV como respaldo de datos secundarios proporciona una solución ágil para reportes.

Flexibilidad:

El diseño basado en MVC permite una fácil extensión de funcionalidades.

Consistencia:

Las validaciones garantizaron la calidad de los datos ingresados.

Aprendizajes

Fortalecimiento en MVC: Mejor comprensión de separación de responsabilidades.

Validaciones: Importancia de la validación en cliente y servidor.

Sincronización: Manejo de bases de datos y sistemas externos (CSV).

Conclusiones

El proyecto cumple con algunos de los requisitos establecidos, implementando un sistema funcional y escalable.

Se logró integrar soluciones para roles, persistencia, validaciones y exportación de datos.

La estructura permite futuras mejoras y adaptaciones.

Diagrama base de datos