GymYang

Alumno: Jaume Moltó Gallego
CIPFP Batoi
Tutor: Iván Martos Gázquez
Ciclo Formativo de Grado Superior en
Desarrollo de Aplicaciones Multiplataforma

Índice

1. INTRODUCCIÓN

Motivo de la elección del proyecto

Desde pequeño siempre que he entrado a un sitio web o aplicación y he tenido que realizar un formulario, ya sea de inicio de sesión o de rellenar una encuesta, siempre me he preguntado: ¿Cómo lo habrán realizado?, ¿Será muy complicado crear uno por cuenta propia?

Este año nos han enseñado a crear y diseñar nuestros propios formularios y aplicaciones con todo tipo de funciones, aún estoy aprendiendo, y el abanico de funcionalidades que se le puede añadir a un simple formulario o aplicación son infinitas. Por eso el proyecto está enfocado en aprender funcionalidades nuevas y aplicar las vistas en clase.

Descripción y objetivos generales del proyecto

El proyecto desarrolla una aplicación de escritorio para un gimnasio que está a punto de abrir. Está diseñada principalmente para el uso cotidiano de los usuarios del gimnasio, ya sean personas experimentadas o noveles. En ella se pueden encontrar vídeos que explican cómo usar las máquinas disponibles de forma correcta, además de poder reservar plaza para las diferentes clases que pueda ofrecer el gimnasio. Cada usuario podrá así planificar su propia rutina y descargarla en un PDF para una mayor portabilidad.

Posibles aplicaciones prácticas del proyecto

El enfoque principal de la aplicación es ser lo más intuitiva e inclusiva posible para que todos los usuarios que la utilicen estén satisfechos. Esto ayudará al gimnasio a crecer.

Imagina que eres una persona que nunca ha ido al gimnasio y que, además, no tiene a nadie cercano con experiencia en este ámbito. Te apuntas al gimnasio GymYang y decides descargar su aplicación de escritorio, ésta te permite familiarizarte con el entorno del gimnasio desde la comodidad de tu casa. A través de la app, puedes acceder a vídeos explicativos que te enseñan a utilizar correctamente cada máquina y realizar los ejercicios de forma segura, evitando lesiones o malos hábitos desde el principio. También cuentas con un calendario interactivo para apuntar tus rutinas personalizadas, organizar tus entrenamientos semanales y hacer seguimiento de tu progreso. Además, tienes la opción de descargar tus rutinas en formato PDF para consultarlas en cualquier momento, incluso sin conexión, y puedes reservar tu plaza en las clases colectivas del gimnasio de forma rápida y sencilla.

En resumen, la aplicación funciona como una guía personal que te acompaña desde el primer día, dándote la confianza necesaria para comenzar en el mundo del fitness con seguridad y motivación.

2. Fundamentos generales teóricos y prácticos

Este proyecto no abarca todas las asignaturas vistas durante los dos años de DAM pero sí que se ve afectado por las siguientes:

1r Curso

Programación

La programación ha sido fundamental para el desarrollo de este proyecto, ya que me ha proporcionado los conocimientos y las bases necesarias para “pensar en código” y crear los archivos .cs que implementan la funcionalidad de la aplicación. Esta asignatura ha sido uno de los pilares clave del proceso, sin ella no habría sido posible sacar adelante el proyecto, ya que enseña a construir una estructura lógica y a resolver problemas de forma progresiva, paso a paso.

Entornos de desarrollo

Esta asignatura me ha parecido especialmente importante por dos razones fundamentales. En primer lugar, porque me ha enseñado a analizar y comparar diferentes entornos de desarrollo (IDEs), valorando sus ventajas e inconvenientes, lo que me ha permitido elegir el más adecuado para mi proyecto. En segundo lugar y para mí, lo más relevante es que me ha transmitido la importancia de preservar y documentar el trabajo realizado. Gracias a ello, he aprendido a llevar un registro organizado del desarrollo del proyecto utilizando GitHub, donde he podido almacenarlo en la nube y realizar un seguimiento de los avances mediante “commits”. Además, la asignatura me ha enseñado a subir el proyecto tanto desde la terminal como desde la interfaz web, y a crear un archivo “ReadMe” para explicar la finalidad del proyecto y facilitar su comprensión a cualquier persona que lo consulte.

Base de datos

Esta asignatura es fundamental para comprender la diferencia entre dato e información, así como para conocer en profundidad qué es una base de datos, cómo funciona internamente y cómo se crea. Además, se enseña el uso del lenguaje SQL, que permite realizar consultas para visualizar, insertar, modificar o eliminar datos dentro de una base de datos. También resulta muy útil para aprender a estructurar y organizar la información correctamente mediante esquemas y tablas, lo que garantiza un almacenamiento eficiente y bien definido.

2º Curso

Desarrollo de interfaces

Otro pilar fundamental en este proyecto ha sido el desarrollo de interfaces, ya que se centra en aspectos clave como la usabilidad y la accesibilidad, es decir, en ofrecer una buena experiencia de usuario. Esta asignatura nos enseña a distribuir correctamente los elementos dentro de las distintas ventanas o formularios, a integrar la interfaz con otros módulos como bases de datos y programación, y a utilizar controladores para conectar la lógica del programa con el diseño visual. Además, se trabaja en la adaptación de las interfaces a diferentes tamaños de pantalla y dispositivos, lo que resulta esencial para crear aplicaciones multiplataforma funcionales e intuitivas.

Acceso a datos

Gracias a la asignatura de Acceso a Datos he aprendido a conectar una aplicación con una base de datos. En mi caso, he utilizado PostgreSQL, y fue bastante sencillo enlazar el proyecto con una base de datos recién creada. Esta asignatura ha sido especialmente útil en dos aspectos clave del proyecto: por un lado, la posibilidad de realizar operaciones CRUD directamente desde el código, como insertar un nuevo usuario, modificar sus datos o incluso eliminarlo de la base de datos; y por otro, el manejo de ficheros, que desde mediados de curso se trabajó con más profundidad y ha sido fundamental para mi proyecto. Gracias a ello, he podido implementar funcionalidades como guardar las reservas o almacenar el calendario personalizado de cada usuario, con la opción de exportarlo o imprimirlo para facilitar su consulta.

3. Fases de implementación técnica del proyecto

En este apartado voy a explicar el desarrollo del proyecto a lo largo de estos meses.

Fase de preparación

En esta fase me centré en lo necesario para empezar a realizar el proyecto:

1. Hardware necesario
    a. Ordenador o portátil (Windows)
2. Software necesario
    a. PostgreSQL (v17 Windows)
    b. Visual Studio 2022 (Community Version Windows)
    c. PgAdmin 4 (v9.0 Windows)
    d. Paquetes NuGet como:
        i. iTextSharp (v5.5.13.3)(Generación de PDFs)
        ii. Npgsql (v9.0.3)(Conectar con postgres)
        iii. BouncyCastle (v1.8.9)(Seguridad)
        iv. Microsoft.Extensions.DependencyInjection.Abstractions (v8.0.2)(gestionar servicios y flexibilidad)
        v. Microsoft.Extensions.Logging.Abstractions(v8.0.2)(Registrar eventos)

Fase de diseño e ideas

Esta fase empezó con la planificación y realización de una lluvia de ideas sobre qué podría tener la aplicación, es decir, qué funcionalidades tendría y cómo realizar dichas funcionalidades.
Una idea que puede estar muy bien fue que yo mismo realicé los vídeos dónde se explica la utilización de las máquinas. Alguna de las ideas concebidas fueron descartadas por falta de tiempo y desconocimiento. Por ejemplo, que dentro de la aplicación se pudieran realizar dietas personalizadas, pero al no tener conocimientos de nutrición y sólo disponer de un período de 4 meses para la realización del proyecto la descarté.

Fase de implementación inicial

Lo primero para llevar a cabo el proyecto fue la creación del mismo en Visual Studio y la creación de la base de datos en PgAdmin 4.
Decidí crear una base de datos sencilla ya que para un usuario en un gimnasio no se necesitan muchos campos, en mi caso decidí que la base tendría 4 campos:

- Campo id (clave primaria, integer)
    - Este campo se utiliza internamente para hacer validaciones, el usuario no es consciente de su id en ningún momento, más adelante se podría hacer cara al público si la aplicación sigue creciendo.
- Campo nombre (varchar de 255 caracteres)
    - Este campo será rellenado con el nombre que se quiera poner el usuario dentro de unos parámetros establecidos, además se utiliza para validación interna y muy necesario para diferenciar rutinas personalizadas y reservas realizadas.
- Campo apellidos (varchar de 255 caracteres)
    - Este campo será rellenado con el nombre que se quiera poner el usuario dentro de unos parámetros establecidos.
- Campo password (varchar de 255 caracteres)
    - Este campo será rellenado con el nombre que se quiera poner el usuario dentro de unos parámetros establecidos, además se utiliza para validación interna.

	id [PK] integer	nombre character varying (255)	apellidos character varying (255)	password character varying (255)
1	4	Jaume	Moltó Gallego	1234
2	5	Vicent	Moltó	1234
3	6	Popi	Popi	12345
4	8	Iván	Martos	12345
5	9	Gabriel	Vilaplana García	12345
6	11	Ripop	Ripop	123

Para la creación del proyecto en Visual Studio elegí la opción de Aplicación de Windows Forms ya que es ideal para una aplicación de escritorio y para Windows. En ella lo primero que hice fue intentar enlazar con la base de datos, para ello tuve que crear una clase llamada **Conexion.cs** y en esta se realizan los pasos necesarios para poder enlazar base de datos y proyecto.

Fase del desarrollo

Formulario Home

Es un formulario simple y limpio donde se puede visualizar el slogan del gimnasio y el logo de éste. A simple vista se puede ver que los colores distintivos de la compañía son el naranja y el negro con toques blancos en algunos apartados. El logo hace referencia al yin y al yang, ya que debe haber un equilibrio entre mente y cuerpo. A los lados tiene algo similar a los extremos de una mancuerna, en referencia a un objeto típico de un gimnasio y debajo está el nombre del mismo.
El formulario cuenta con dos botones principales, situados de forma visible para facilitar la navegación del usuario. Estos botones permiten al usuario crear una cuenta nueva o bien iniciar sesión en caso de que ya tenga una. Cada uno de estos botones se dirige a formularios específicos: uno dedicado al registro de nuevos usuarios, donde se solicita la información necesaria para la creación de una cuenta, y otro para el inicio de sesión de usuarios ya registrados, garantizando así un acceso sencillo y personalizado a la aplicación.
El fondo es una colección de fotos que se van intercalando cada dos segundos, mostrando parte de las máquinas e instalaciones del gimnasio, para poder hacer que vayan intercalando era necesario añadir un “evento” llamado *timer* que es el encargado de intercambiar las fotos cada dos segundos.
El formulario está anclado, de forma que si quieres minimizar o agrandarlo los elementos del diseño escalan según el tamaño que el usuario/cliente quiera tener.

Formulario de Inicio de sesión

He diseñado un formulario sencillo donde el usuario puede interactuar mediante el teclado. Cuenta con dos campos de texto (TextBox) para introducir el nombre de usuario y la contraseña, junto con un botón que, al ser pulsado, inicia sesión si las credenciales son correctas.

Para verificar esta información, se realiza una consulta SQL del tipo “SELECT” a la base de datos, recuperando el nombre y la contraseña del usuario y comparándolos con los datos introducidos por teclado. Además, pensando en futuras funcionalidades, decidí crear una clase llamada **Usuario.cs**, que almacena los datos introducidos tanto en el formulario de inicio de sesión como en el de registro. Esta clase permite conservar la información del usuario activo y reutilizarla en otros formularios, facilitando así la identificación y personalización por usuario dentro de la aplicación.

Formulario de Creación de usuario

Este formulario es muy parecido al de inicio de sesión, ya que el diseño visual es prácticamente el mismo, aunque incorpora dos campos adicionales (TextBox) para que el usuario introduzca y confirme su contraseña. La principal diferencia radica en su lógica interna, ya que en este caso es fundamental realizar correctamente todas las validaciones antes de enviar los datos del nuevo usuario a la base de datos. Para ello, el formulario realiza dos consultas SQL. La primera es una “SELECT”, que permite comprobar si el nombre de usuario ya existe en la base de datos. En caso de que el nombre esté repetido o las contraseñas introducidas no coinc3idan, se lanza un mensaje de error informando al usuario de que no puede crear su perfil. Sólo si todas las validaciones se superan correctamente, se ejecuta una segunda consulta del tipo “INSERT”, que registra el nuevo usuario en la base de datos. Al igual que en el formulario de inicio de sesión, también se guarda la información en una instancia de la clase **Usuario.cs**, permitiendo así conservar los datos del usuario registrado para futuras operaciones dentro de la aplicación.

Formulario de Página principal

Este formulario representa el corazón de la aplicación de escritorio, no por su complejidad técnica, sino porque desde él se gestionan y visualizan todas las funcionalidades relacionadas con ejercicios, clases y rutinas. Su diseño se compone de dos partes principales: un gran panel a la izquierda, que actúa como “pantalla” principal donde se muestran las distintas funcionalidades, y una barra lateral a la derecha, que funciona como menú de navegación.

En el panel izquierdo se pueden realizar acciones como personalizar la rutina, reservar plaza en una clase o visualizar vídeos explicativos sobre el uso correcto de las máquinas del gimnasio. La barra lateral, por su parte, está organizada mediante varios paneles y subpaneles para lograr una distribución clara e intuitiva. En la parte superior de este menú se encuentra un botón que dirige al perfil del usuario, abriendo un formulario específico que explicaré posteriormente.

Justo debajo, hay cinco botones correspondientes a distintos grupos musculares (brazo, pecho, espalda, piernas y abdomen). Al pulsar uno de estos botones, se despliega un subpanel con tres botones adicionales, cada uno asociado a una máquina distinta que trabaja músculos dentro del grupo seleccionado. Por ejemplo, si
se elige el botón de “Pecho” y luego uno de los botones del subpanel como “Jalón de pecho”, se mostrará en el panel principal un vídeo explicativo sobre el uso correcto de esa máquina, con el fin de evitar lesiones y fomentar el uso responsable del material. Para que estos subpaneles funcionen correctamente, inicialmente se configuran con el valor de visibilidad en false y luego, mediante un método específico, se activa su visualización a true.

Más abajo, se encuentra un botón para acceder al apartado de reserva de clases. Al pulsarlo, se despliega otro subpanel con dos botones, cada uno enlazado a un nuevo formulario (explicado más adelante) que permite reservar una plaza en una de las clases disponibles. Finalmente, en la parte inferior del menú lateral, hay un botón que lleva a una nueva vista con un calendario. Este calendario permite seleccionar un día de la semana y escribir anotaciones, ideal para planificar una rutina semanal personalizada.

En cuanto al código, a pesar de ser uno de los pilares fundamentales de la aplicación, su estructura es relativamente simple. La mayor parte del funcionamiento se basa en que, al pulsar un botón, se muestre un formulario dentro del panel izquierdo del formulario principal. Para lograr esto, he creado un método al que se le pasa como parámetro un formulario hijo, el cual se mete dentro del formulario padre. Cabe destacar que el botón “Mi perfil” funciona de manera distinta al resto, ya que en lugar de mostrar un nuevo formulario en el panel, redirige directamente a una nueva vista, es decir, cambia por completo la interfaz visible.

Dentro de este método, lo primero que se hace es comprobar si ya hay un formulario abierto; si es así, se cierra. Luego, se configuran las propiedades del formulario hijo de manera explícita en el código: se ajusta su tamaño, se desactiva su borde, y se le posiciona correctamente dentro del contenedor. Posteriormente, se lleva al frente con formHijo.BringToFront() y finalmente se muestra con formHijo.Show().

Esto es lo explicado anteriormente de una manera más visual y clara:
(Los demás botones están sin desplegar)

Formulario de Mi perfil

Este formulario es sencillo y está diseñado para mostrar los datos del usuario, como el nombre, los apellidos y la contraseña. Al iniciar, la contraseña aparece cifrada pero incluye un botón que permite hacerla visible. Para ello, programé un método que simplemente cambia los caracteres mostrados en el TextBox.

Para que el formulario pueda mostrar los datos del usuario que ha iniciado sesión, es necesario pasarle un objeto Usuario con dicha información. Por eso, también necesité pasar ese objeto al formulario de la página principal, desde donde se accede a éste.

Además, el formulario cuenta con cuatro botones con distintas funcionalidades. El primero es el botón de modificar, que permite al usuario actualizar sus datos. Al pulsarlo, se realiza una consulta “UPDATE” que reemplaza todos los campos en la base de datos con los valores que estén escritos en ese momento, incluso si sólo se ha modificado uno (por ejemplo, el nombre).

El segundo botón sirve para volver atrás, es decir, salir del perfil y regresar a la vista principal. Durante el desarrollo, me encontré con un problema: si cerraba este formulario directamente, el anterior no se recargaba de forma correcta. Por eso, también tuve que pasarle el formulario principal como parámetro.

El tercer botón es para eliminar la cuenta, pensado para usuarios que desean darse de baja del gimnasio. Al pulsarlo, aparece una advertencia preguntando si se desea continuar con la eliminación. Si se confirma, se ejecuta una consulta “DELETE” que elimina el registro correspondiente en la base de datos. Después, se cierran todos los formularios abiertos y se redirige al formulario de inicio, donde se puede crear una nueva cuenta o iniciar sesión.

Por último, el cuarto botón permite cerrar sesión y también redirige directamente al formulario de inicio.

Formulario de Videos explicativos

La función principal de este formulario es reproducir un vídeo, y aunque pueda parecer muy simple, tuve que desarrollar dos versiones diferentes debido a errores de planteamiento inicial. Para lograr que el formulario pudiera reproducir vídeos, investigué distintas librerías hasta que finalmente opté por **AxWMPLib** junto con **WMPLib**, ya que ambas trabajan conjuntamente y permiten integrar un reproductor de vídeo funcional en la aplicación.

Para los vídeos, tuve la idea de grabarme en el gimnasio usando las máquinas disponibles y añadir esas grabaciones al proyecto. Como comenté anteriormente, desarrollé dos versiones del formulario. La primera simplemente reproducía un vídeo específico al pulsar un botón concreto, utilizando una ruta absoluta escrita directamente en el código. Sin embargo, pronto me di cuenta que esa solución no era escalable.

Así que creé una segunda versión, más flexible, donde el formulario recibe un parámetro con la ruta del vídeo (el path), permitiendo que funcione con cualquier archivo sin necesidad de modificar el código cada vez. Al principio, tuve problemas al intentar usar rutas absolutas directamente. Para solucionarlo, dividí la ruta en dos partes y las uní antes de pasarlas al formulario del vídeo. Por eso, en el formulario principal declaré dos atributos: uno con la primera parte del path, otro con la segunda más la primera, este atributo es el que finalmente se envía al formulario del vídeo.

En cuanto al diseño, es muy simple, el reproductor ocupa todo el espacio del formulario y permite al usuario pausar, adelantar o retroceder el vídeo según prefiera.

Formulario de Reservar clases

Este formulario tiene como función principal permitir la reserva de una plaza para una clase, ya sea de zumba o yoga. Está compuesto por varios botones: uno de ellos actúa como etiqueta (label), aunque decidí implementarlo como botón para poder personalizarlo mejor y mantener la estética general del proyecto. El componente principal es un calendario, ubicado en el centro del formulario.

La distribución es sencilla, el calendario se encuentra en el centro, rodeado de los botones que simulan las plazas disponibles, y en la parte superior está el botón-etiqueta que indica el tipo de clase (zumba o yoga).

En cuanto a la lógica del formulario, fui construyéndola paso a paso. Primero, implementé el que al pulsar una plaza ésta cambiara de color a negro, indicando que ya ha sido reservada por un usuario. El segundo paso fue asegurarme de que un usuario no pudiera cancelar la reserva de otro, es decir, por ejemplo, que Pepe no pudiera anular la reserva hecha por Jaume ni por ningún otro usuario, sólo la suya.

Finalmente, el paso más complejo fue lograr que cada día del calendario fuera independiente y que, además, el estado de las reservas se mantuviera guardado incluso después de cerrar la aplicación. A continuación, mostraré una breve traza del código:

```
private void BotonPlaza_Click(object sender, EventArgs e)
{
    Button btn = sender as Button;
    int plazaId = (int)btn.Tag;
    string fechaKey = monthCalendar.SelectionStart.Date.ToString("yyyy-MM-dd");

    if (!reservasPorFecha.ContainsKey(fechaKey))
        reservasPorFecha[fechaKey] = new Dictionary<int, string>();

    if (reservasPorFecha[fechaKey].ContainsKey(plazaId))
    {
        if (reservasPorFecha[fechaKey][plazaId] == usuarioActual)
        {
            //Solo puedes liberar tu propia plaza
            reservasPorFecha[fechaKey].Remove(plazaId);
            btn.BackColor = Color.SandyBrown;
            btn.Text = plazaId.ToString();
        }
        else
        {
            MessageBox.Show($"Plaza reservada por otro usuario: {reservasPorFecha[fechaKey][plazaId]}",
                            "No puedes liberar esta plaza", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
    else
    {
        //Reservar plaza
        reservasPorFecha[fechaKey][plazaId] = usuarioActual;
        btn.BackColor = Color.Black;
        btn.Text = usuarioActual.Substring(0, 1).ToUpper();
    }

    GuardarReservas();
}
```

Las tres primeras líneas del método corresponden a variables necesarias, siendo la más importante la última, ya que permite determinar la fecha exacta de la reserva.
La primera comprobación que se realiza es para verificar si ya existe alguna reserva registrada para esa fecha. Si no es así, se crea una nueva entrada en reservas por fecha, que es un atributo de la clase.

A continuación, se comprueba si la plaza en cuestión ya está reservada. Si lo está, se revisa qué usuario la reservó para determinar si el usuario actual tiene permiso para cancelarla. Por el contrario, si la plaza no está reservada, se asigna automáticamente al usuario actual, y se mostrará la inicial de su nombre sobre dicha plaza.

Lo más complejo de este proceso fue plasmar la idea inicial en código y definir claramente los pasos a seguir y juntarlo con los métodos ya creados anteriormente, lo más difícil fue la lógica que debía de implementar para este formulario.

Formulario del Calendario para rutinas

El objetivo principal de este formulario es permitir al usuario planificar su rutina semanal de ejercicios de forma libre y personalizada. La idea es crear un calendario interactivo en el que se puedan anotar los ejercicios que se realizan cada día de la semana, permitiendo así organizar el ritmo del usuario sin la presión de seguir un plan impuesto por un entrenador personal. Esta funcionalidad busca fomentar la autonomía y adaptar la experiencia a las necesidades de cada usuario.

Además, para mejorar la utilidad y la portabilidad, decidí implementar una función que permite exportar la rutina en formato PDF, de forma que el usuario pueda imprimir su planificación y llevarla consigo al gimnasio o consultarla sin necesidad de tener la aplicación abierta.

En cuanto a la lógica, guarda cierta similitud con el formulario anterior en el sentido de que también se guarda un estado, aunque en este caso se trata de contenido escrito por el usuario en lugar del estado de un botón. Para conservar esta información de forma persistente, decidí utilizar archivos .json, lo cual me pareció una solución eficiente y sencilla. Cada vez que el usuario abre la aplicación, estos archivos se cargan automáticamente, permitiendo visualizar los ejercicios anotados previamente para cada día.

4. Problemas

Diseños

Uno de los principales problemas que encontré fue que algunos componentes, como el calendario, no permitían modificar su diseño, ni desde el propio editor ni escribiendo el código manualmente. Viendo esta limitación, me puse a investigar y probé con varias librerías externas que, en teoría, ofrecían más opciones de personalización. Sin embargo, ninguna me permitió hacer los cambios que buscaba. Además, al cambiar de librería, el estilo visual de otros elementos como los botones o formularios se veían afectados, rompiendo la estética general del proyecto. Por todo esto, decidí mantener la librería original, aunque tuviera sus limitaciones, para no complicar más el desarrollo ni perder la coherencia en el diseño de la aplicación.

Repositorio en GitHub

Antes de empezar con el desarrollo del proyecto, creé un repositorio en GitHub con el objetivo de llevar un registro del trabajo realizado a lo largo del tiempo. Sin embargo, unas dos o tres semanas antes de la fecha de entrega, se agotó la batería de mi portátil mientras realizaba un “commit” o un “push”, y al revisar el repositorio noté que faltaban archivos y que algunos no estaban correctamente sincronizados. Tras investigar en diversas fuentes, llegué a la conclusión de que el “commit” o “push” se había corrompido. Por esta razón, decidí eliminar el repositorio anterior, crear uno nuevo y volver a subir el trabajo correspondiente a esas últimas dos semanas.

.json

El principal problema que experimenté con los archivos **.json** fue que la información no se cargaba correctamente al reiniciar la aplicación. Por ejemplo, si reservaba una plaza en una clase y luego cambiaba de vista para después volver a la sección de reservas, la aplicación mostraba que la reserva se había realizado con éxito, lo cual indicaba que el sistema estaba funcionando correctamente. Sin embargo, al cerrar y volver a abrir la aplicación, la reserva desaparecía por completo. Tras revisar el código, descubrí que esto ocurría porque el método encargado de cargar los datos desde el archivo **.json** no se estaba ejecutando en el momento adecuado dentro del ciclo de vida de la aplicación. El orden de ejecución era incorrecto, y como consecuencia, la información no se cargaba desde el archivo al iniciar la aplicación, lo que provocaba la pérdida aparente de los datos almacenados.

5. Coste económico de implementación real

Si se quisiera implementar este proyecto para su uso, necesitamos publicar la aplicación en una tienda como por ejemplo en Microsoft Store. Además, hacer una campaña de marketing en la que se anuncie la aplicación y ser una condición importante para elegir el gimnasio. En cuanto a los costes legales habría que contratar los servicios de un abogado para que nos ayude con este tipo de temas legales y además contratar también un tipo de licencia específica. Si en un futuro pasase a ser una aplicación de móvil sería necesario consultar los precios de la Play Store y la App Store.
Todo esto sabiendo que como el que ha desarrollado la aplicación y la ha probado he sido yo, no cuesta nada de dinero.

Concepto	**Precio**
Publicación Microsoft Store	90€ (pago único)
Marketing	100€ (depende de la campaña)
Abogado especializado	400€
Firma digital	505€
TOTAL	**1095€**

6. Comparación situación actual con soluciones alternativas

Hoy en día, el mundo del fitness y el bienestar físico está en pleno auge, y cada vez existen más gimnasios con propuestas variadas. La mayoría de estos centros ofrecen aplicaciones móviles para gestionar reservas o consultar información, pero muy pocos cuentan con una versión de escritorio. Comparando mi aplicación con algunas opciones tanto móviles como de escritorio, he llegado a la conclusión que, aunque aún le faltan ciertas mejoras que comentaré más adelante, también aporta funcionalidades que otras no incluyen. Una de ellas, y que considero un punto diferenciador, es la posibilidad de escribir tu propia rutina personalizada dentro de un calendario integrado en la aplicación. Además, esta rutina puede imprimirse fácilmente en formato PDF, lo cual resulta muy útil tanto para uso personal como para compartirla con un entrenador o dietista si se desea un seguimiento más detallado.

7. Conclusiones

Resultados obtenidos

Los resultados obtenidos han sido los esperados en cuanto al funcionamiento general de la aplicación, aunque no exactamente los deseados. Como he comentado anteriormente, algunas ideas iniciales tuvieron que ser descartadas, como la incorporación de un apartado dedicado a dietas personalizadas, que habría complementado muy bien la propuesta. Aun así, estoy satisfecho con el resultado final del proyecto, ya que ha cumplido con creces mi principal objetivo, aprender y profundizar en el uso de Windows Forms y la gestión de formularios, aplicándolo de forma práctica en el desarrollo de una aplicación de escritorio funcional y personalizada además de tocar un tema de actualidad.

Puntos pendientes o mejoras

Como he mencionado anteriormente, haber añadido un apartado de dietas personalizadas habría sido un punto clave que le habría dado un valor añadido muy interesante a la aplicación de escritorio. Otra posible mejora sería en la forma de almacenar los datos, tanto de las reservas de plazas como del calendario donde se apunta la rutina personalizada. En lugar de utilizar archivos **.json**, habría sido más eficiente y profesional hacerlo mediante una base de datos, y si esta fuera remota mucho mejor, ya que permitiría mayor escalabilidad y acceso desde distintos dispositivos. Sin embargo, debido al tiempo limitado del proyecto (40 horas estimadas), no fue posible implementar todas estas ideas. Aun así, en mi caso personal, he dedicado entre 60-70 horas para poder alcanzar un resultado con el que sentirme satisfecho.

Consejos destacables

Normalmente, cuando uno tiene un margen de 3-4 meses para desarrollar un proyecto, lo habitual es relajarse al principio. Sin embargo, personalmente recomiendo empezar desde los primeros días, aunque sea con una simple lluvia de ideas o planificando cómo organizar el trabajo a lo largo de las prácticas.

En nuestro campo es muy común frustrarse por errores que no logramos identificar ni siquiera tras depurar varias veces. En mi caso, algo que me funcionó fue pedir ayuda a personas externas al proyecto para que revisaran el código o, directamente, dejar el error aparcado un día y retomarlo al día siguiente con la mente más despejada. Parece una tontería, pero realmente ayuda a ver cosas que antes pasaban desapercibidas. Por eso creo que es fundamental ser paciente, constante y evitar dejarlo todo para el final.

Personalmente, lo más tedioso ha sido redactar la memoria del proyecto, pero si se lleva al día y se va escribiendo un punto cada día, se hace mucho más llevadera y no se acumula trabajo innecesariamente.

8. Necesidades y sugerencias de formación

A lo largo del ciclo formativo, se nos enseñan a implementar pequeñas funcionalidades a través de distintas mini actividades, lo cual está bien para entender conceptos concretos. Sin embargo, considero que sería más útil si, después de aprender esas funcionalidades, se propusiera un proyecto más grande donde se pudieran integrar todas o al menos la mayoría de ellas. Esto permitiría ver cómo encajan entre sí en un entorno más realista.

Por otro lado, creo que para algunas personas puede resultar complicado adaptarse a **C#** si no tienen una base previa. En primero, la base que se nos da es **Java**, un lenguaje muy accesible y con una estructura sólida que sienta buenas bases para otros lenguajes. Aun así, habría estado bien profundizar más en cada lenguaje por separado, para poder dominarlos mejor en lugar de pasar rápidamente de uno a otro.

9. Bibliografía

Visual Studio 2022
https://visualstudio.microsoft.com/es/vs/community/

PostgreSQL
https://www.postgresql.org/download/

PgAdmin
https://www.pgadmin.org/download/

Información sobre **AxWMPLib** y **WMPLib**
https://learn.microsoft.com/es-es/previous-versions/windows/desktop/wmp/using-the-windows-media-player-control-with-microsoft-visual-studio?utm_source=chatgpt.com

Problemas GitHub
https://git-scm.com/book/es/v2/Los-entresijos-internos-de-Git-Mantenimiento-y-recuperaci%C3%B3n-de-datos.html

https://www.analyticslane.com/2024/03/13/guia-para-recuperar-commits-perdidos-en-git/

Desarrollo de la aplicación (temario en clase)
SA3. Generación de interfaces gráficas de usuario
SA4. Creación de informes

10. Recursos utilizados

Hardware

PC

Se requiere un ordenador de gama media-baja para poder ejecutar la aplicación de Windows Forms sin problema, en mi caso decidí utilizar un portátil ya que para el día de la presentación me puede ser muy útil y además he hecho todas las pruebas en él.

Software

Visual Studio 2022

Visual Studio es el entorno de desarrollo integrado (IDE) perfecto para este proyecto, orientado a la creación de aplicaciones de escritorio para **Windows** que además ofrece soporte para varios lenguajes pero en mi caso he usado **C#**, otro aspecto positivo es que tiene soporte para **GitHub** y hace que los commits resulten más fáciles.

PostgreSQL

Es un sistema de gestión de base de datos relacional, reconocido por su potencia, estabilidad y cumplimiento de estándares. Es ideal para manejar aplicaciones aplicadas a Windows o sistemas empresariales como es mi caso, soporta extensiones **.json** y permite la personalización directa por el usuario, otro punto a favor es que es fácil de implementar en lenguajes como **C#**.

PgAdmin 4

Es una herramienta gráfica de administración y desarrollo para bases de datos **PostgreSQL**. Permite la administración de base de datos, tablas, funciones y otros objetos de forma visual, intuitiva y fácil. También ofrece un editor de consultas **SQL** para que puedas ver resultados o datos concretos dentro de tu base de datos.

C#

C# es un lenguaje de programación moderno, orientado a objetos y desarrollado por Microsoft como parte de su plataforma **.NET**. Está diseñado para ser simple, seguro y versátil, permitiendo desarrollar una amplia variedad de aplicaciones, desde software de escritorio (como en mi caso) y servicios web hasta aplicaciones móviles y videojuegos. **C#** combina lo mejor de **Java** y **C++**, como por ejemplo la gestión automática de memoria, lenguaje fácil y legible y programación asíncrona.

Paquetes NuGet

Estos son componentes reutilizables que puedes descargar para tu proyecto, estos contienen código, herramientas, bibliotecas u otros recursos útiles para los desarrolladores.
Es el sistema oficial de gestión de paquetes para la plataforma .NET, y permite instalar, actualizar y administrar dependencias directamente desde Visual Studio o mediante línea de comandos.