



Proyecto Final DAM

VriviYoga

Virginia Ruiz Escaño

CESUR FORMACIÓN

2024-2025

RESUMEN

Esta memoria presenta el desarrollo de la aplicación *VriviYoga*, un sistema diseñado para facilitar la gestión de clases en una academia de yoga. El proyecto abarca el diseño de interfaces, funcionalidades clave como el registro de usuarios y gestión de clases, y la conexión con una base de datos MySQL. Se explican los objetivos, metodología, dificultades y resultados obtenidos durante el proceso de desarrollo.

ABSTRACT

This report presents the development of *VriviYoga*, an application designed to simplify class management for a yoga academy. The project includes interface design, key features such as user registration and class management, and integration with a MySQL database. It describes the objectives, methodology, challenges, and outcomes throughout the development process.

ÍNDICE

1 – INTRODUCCIÓN.....	6
1.1 Contexto y necesidades del sector del yoga.....	7
1.2 Panorama actual y soluciones existentes.....	7
1.3 Motivación y propósito del desarrollo de VriviYoga.....	9
2 – DEFINICIÓN DEL PROYECTO.....	9
2.1 Objetivos estratégicos del desarrollo de VriviYoga.....	9
2.2 Audiencia, impacto y enfoque de la solución propuesta.....	10
3 – ANÁLISIS TÉCNICO DE VRIVIYOGA.....	11
3.1 Herramientas y recursos utilizados en el desarrollo.....	11
3.2 Metodología aplicada en el desarrollo.....	12
3.3 Simulación económica y análisis de recursos.....	13
4 – DISEÑO DE LA APLICACIÓN.....	14
4.1 Necesidades y funcionalidades del sistema.....	14
4.1.1 Funcionalidades previstas por el rol.....	14
4.1.2 Interacción entre usuarios y funcionalidades clave.....	14
4.1.3 Requisitos no funcionales.....	16
4.1.4 Interfaces de usuario.....	16
4.1.5 Gestión de comunicación entre sistema y usuario.....	16
4.2 Condiciones técnicas y límites del desarrollo.....	16
5 – DISEÑO DE LA BASE DE DATOS.....	17
5.1 Descripción general de la solución desarrollada.....	17
5.2 Definición estructural de los datos del sistema.....	18
5.2.1 Modelo entidad-relación.....	18
5.3 Transformación del modelo entidad-relación a modelo físico.....	18
5.4 Representación gráfica del modelo físico de la base de datos.....	19
5.5 Especificación detallada de las estructuras de datos.....	20
5.6 Script de definición e inserción de datos de la base de datos.....	21

6 – ORGANIZACIÓN FUNCIONAL CÓDIGO FUENTE Y ARQUITECTURA.....	21
6.1 Arquitectura del proyecto.....	21
6.2 Esctructura del código fuente.....	22
6.3 Conexión a la base de datos.....	22
6.4 Interacción entre formularios.....	23
6.5 Lógica del funcionamiento.....	23
6.6 Navegación y experiencia visual.....	23
6.7 Capturas del código fuente.....	23
7 – COMPROBACIONES FUNCIONALES INICIALES DE LA APLICACIÓN.....	31
8 – DIFICULTADES DETECTADAS Y POSIBLES MEJORAS.....	32
8.1 Dificultades encontradas.....	32
8.2 Posibles mejoras futuras.....	33
9 – PLANIFICACIÓN TEMPORAL POR FASES Y DESARROLLO.....	33
10 – DESPLIEGUE DE LA APLICACIÓN.....	34
11 – MANUAL DE USUARIO.....	36
11.1 Requisitos para el uso de la aplicación.....	36
11.2 Inicio de sesión.....	36
11.3 Registro de nuevos usuarios	36
11.4 Funcionalidades del rol alumno.....	36
11.5 Funcionalidades del rol instructor.....	37
11.6 Consejos de uso.....	37
12 – MANTENIMIENTO.....	37
12.1 Tipos de mantenimiento.....	37
12.2 Herramientas y buenas prácticas.....	38
12.3 Documentación de soporte.....	39
13 – CONCLUSIONES.....	39
14 – GLOSARIO DE SIGLAS Y ABREVIATURAS.....	39
15 – BIBLIOGRAFÍA.....	40
16 – ANEXOS.....	41

ÍNDICE DE FIGURAS

Figura 1: Secuencia de yoga en la antigüedad.....	6
Figura 2: Software de estudio de yoga Momoyoga.....	8
Figura 3: App de Lalita (academia de yoga).....	9
Figura 4: Diagrama de Gantt.....	13
Figura 5: Diagrama de casos de uso de VriViYoga.....	15
Figura 6: Diagrama de base de datos.....	18
Figura 7: Tabla usuarios.....	19
Figura 8: Tabla clases.....	19
Figura 9: Tabla reservas.....	20
Figura 10: Fragmento Script SQL.....	21
Figura 11: Conexión base de datos (código).....	22
Figura 12: Inicio de sesión (código).....	24
Figura 13: Registro de usuario (código).....	25
Figura 14: Reserva de clase (código).....	26
Figura 15: Visualización de reservas (alumno) (código).....	27
Figura 16: Creación de clase (instructor) (código).....	28
Figura 17: Visualización de clases (instructor) (código).....	30
Figura 18: Inicio de sesión (diseño visual).....	41
Figura 19: Registro de usuario (diseño visual).....	41
Figura 20: Visualización de reservas 1 (alumno) (diseño visual).....	42
Figura 21: Visualización de reservas 2 (alumno) (diseño visual).....	42
Figura 22: Visualización de creación de clase 1 (instructor) (diseño visual)	43
Figura 23: Visualización de creación de clase 2 (instructor) (diseño visual)	43
Figura 24: Visualización de alumnos inscritos a clases (diseño visual)	44

1. INTRODUCCIÓN

1.1. Contexto y necesidades del sector del yoga

En la actualidad, disciplinas como el yoga, la meditación, el Tai Chi o el Pilates han ganado un espacio destacado en la vida de millones de personas. Se estima que más de 300 millones de personas practican yoga de forma regular, consolidándolo como una de las actividades más relevantes para mejorar la salud física, mental y emocional.

En el caso del yoga, es una disciplina originaria de la antigua India, se remonta a al menos 5000 años atrás en la historia. La palabra “yoga” deriva del sánscrito “yuj” que significa unir, simbolizando la unión del cuerpo, mente y espíritu.



Figura 1: Secuencia de yoga en la antigüedad

Hay más de una treintena de tipos diferentes de yoga, aquí algunos de los más comunes practicados actualmente:

- **Hatha Yoga:** Es la base de muchos estilos y se centra en la práctica de posturas, respiración y meditación.
- **Vinyasa Yoga:** Se caracteriza por la sincronización fluida de movimiento y respiración.
- **Ashtanga Yoga:** Físico y riguroso con secuencia específica de asanas (posturas).
- **Kundalini Yoga:** Posturas, respiración, meditación y mantras para despertar energía kundalini.
- **Yin Yoga:** Enfocado en articulaciones y tejido profundos, busca relajación y flexibilidad.

- **Bikram Yoga:** Practicado en ambiente caliente y húmedo con secuencias de 26 asanas.

El yoga no es únicamente una actividad física, sino una filosofía de vida que promueve el equilibrio entre cuerpo, mente y espíritu. Este auge ha fomentado la creación de academias y espacios especializados, donde comunidades enteras buscan reconectarse consigo mismas y reducir el estrés cotidiano.

Sin embargo, muchas de estas academias – especialmente las de tamaño pequeño o mediano – continúan gestionando sus operaciones de forma manual. La falta de herramientas digitales eficaces provoca errores en la reserva de clases, duplicación de datos, desorganización interna y una experiencia insatisfactoria tanto para alumnos como para instructores. Esta realidad evidencia una necesidad urgente de soluciones tecnológicas adaptadas al contexto real de estas instituciones.

1.2. Panorama actual y soluciones existentes

El número de herramientas digitales orientadas específicamente a academias de yoga es limitado, especialmente si se consideran aquellas accesibles para centros independientes con pocos recursos. Existen plataformas como *Momoyoga* o *Ubindi*, que ofrecen funcionalidades avanzadas, pero que están más enfocadas en academias de gran tamaño, con precios elevados y procesos de configuración poco intuitivos.

En consecuencia, muchas academias optan por soluciones informales como hojas de cálculo, grupos de WhatsApp o agendas en papel. Aunque inicialmente pueden cumplir con su función, estas opciones presentan claras limitaciones para la gestión de clases, comunicación fluida y seguimiento de reservas. Además, dificultan el crecimiento del centro y no ofrecen una experiencia profesional al usuario.

Este vacío en el mercado plantea una clara oportunidad para una herramienta como *VriViYoga*, pensada como una aplicación sencilla, intuitiva, económica y diseñada específicamente para academias que buscan digitalizarse sin necesidad de conocimientos técnicos avanzados.

1.3. Motivación y propósito del desarrollo de VriViYoga

El desarrollo de *VriViYoga* surge como una respuesta a los problemas observados en academias de yoga reales, que enfrentan dificultades a la hora de organizar sus clases, gestionar usuarios y optimizar la experiencia de quienes participan en las sesiones. Digitalizar estas tareas básicas supone una mejora significativa en la eficiencia interna y en la satisfacción del alumnado.

La aplicación desarrollada permite registrar usuarios, reservar clases según el rol (alumno o instructor), visualizar el calendario y gestionar inscripciones. El diseño se ha reforzado con elementos gráficos personalizados, fuentes temáticas y un enfoque amigable al usuario, asentando el vínculo con la identidad del yoga.

En resumen, *VriViYoga* no solo se presenta como una solución real a un problema específico, sino como una herramienta funcional, adaptable y visualmente cuidada, capaz de acompañar a academias de yoga en su proceso de digitalización, con el objetivo de mejorar tanto su organización interna como la experiencia del usuario final.

Ejemplos visuales de las pocas apps usadas para este fin :

Momoyoga, es de las más usadas:



Figura 2: Software de estudio de yoga Momoyoga

Lalita, de una academia local que tiene su propia app:



Figura 3: App de Lalita (academia de yoga)

2. DEFINICIÓN DEL PROYECTO

2.1. Objetivos estratégicos del desarrollo de VriviYoga

El desarrollo de VriviYoga responde a la necesidad de dotar a las academias de yoga de una solución digital eficiente, accesible y orientada a la mejora de su operativa diaria. A continuación, se detallan los objetivos generales y específicos que guían la ejecución del proyecto.

Objetivo general

Diseñar e implantar una plataforma digital multiplataforma que permita a las academias de yoga gestionar clases, reservas y usuarios de forma intuitiva, optimizando los recursos disponibles y mejorando la experiencia del alumno y del instructor

Objetivos específicos

1. Desarrollo de funcionalidades clave:

- Diseñar un sistema de registro e inicio de sesión con validación básica de credenciales.

- Implementar funcionalidades diferenciadas para alumnos, instructores e instructores.

2. Optimización de procesos administrativos:

- Permitir la creación y visualización de clases desde el perfil del instructor.
- Facilitar la reserva de clases por parte del alumnado y su posterior gestión.

3. Integración de tecnológica:

- Desarrollar la solución con Visual Studio y Windows Forms utilizando el lenguaje C#.
- Conectar la aplicación a una base de datos MySQL, diseñada y gestionada mediante MySQL Workbench, para asegurar el almacenamiento estructurado y seguro de la información.

4. Experiencia de usuario centrada en el bienestar

- Aplicar un diseño visual simple y armonioso, alineado con los valores del yoga: calma, sencillez y equilibrio.
- Utilizar una estética cuidada y amigable que mejore la usabilidad y comprensión por parte del usuario.

5. Contribución al entorno educativo y profesional

- Brindar al desarrollador una oportunidad de aplicar conocimientos adquiridos en el ciclo formativo de DAM.
- Servir como modelo funcional y replicable para otros centros o estudiantes con necesidades similares.

2.2. Audiencia, impacto y enfoque de la solución propuesta

El proyecto *VriviYoga* se desarrolla en el contexto de academias de yoga pequeñas y medianas que, a pesar de contar con una creciente demanda de servicios, no disponen de herramientas tecnológicas especializadas que les permitan organizar su actividad de forma eficiente.

Este tipo de centros suele operar mediante sistemas presenciales y procesos manuales, lo que conlleva errores de comunicación, duplicidad de reservas y una pérdida de control sobre la planificación de clases. *VriviYoga* propone una solución digital sencilla, flexible y fácil de usar, adaptada a su realidad.

Público destinatario

Perfil	Ventajas de BriviYoga	Posibles retos iniciales
Academias de yoga	Optimización de la gestión, ahorro de tiempo, imagen profesional	Adaptación inicial a la herramienta digital
Instructores	Mejora de la organización personal, creación y edición de clases	Preferencia por métodos tradicionales
Alumnos	Reserva online inmediata, acceso a clases organizadas	Dificultades de adaptación tecnológica en público mayor

3. ANÁLISIS TÉCNICO Y TECNOLÓGICO DE VRIVIYOGA**3.1. Herramientas y recursos utilizados en el desarrollo**

El desarrollo de la aplicación *VriviYoga* ha requerido una combinación de recursos tanto de software como de hardware. A continuación, se describen las tecnologías, entornos y herramientas utilizadas, así como el equipo mínimo necesario para su ejecución.

A. Entorno de desarrollo y tecnologías software**1. Lenguaje de programación – C#**

Utilizado para implementar toda la lógica del sistema, incluyendo la navegación entre formularios, validaciones, conexión a base de datos y operaciones CRUD.

2. Framework – .NET (Windows Forms)

Se ha empleado Windows Forms como tecnología principal para construir una aplicación de escritorio en entorno Windows, con interfaz gráfica adaptable y funcional.

3. Base de datos – MySQL

Motor de base de datos relacional que permite almacenar de manera estructurada los datos de usuarios, clases, instructores y reservas.

4. Gestión visual de la base de datos – MySQL Workbench

Herramienta gráfica empleada para el diseño del modelo entidad-relación, creación de tablas y administración de los datos durante el desarrollo.

5. Entorno de programación – Visual Studio 2022

Entorno de desarrollo integrado (IDE) utilizado para programar, compilar, probar y depurar el código de la aplicación.

6. **Gestión de control de versiones – GitHub**

Se ha utilizado para el almacenamiento y respaldo del código, así como para permitir su accesibilidad y mantenimiento futuro.

B. Requisitos hardware mínimos

Para el desarrollo y ejecución de VriViYoga, se ha utilizado el siguiente equipo:

- **Sistema operativo:** Windows 11
- **Procesador:** Intel Core i7
- **Memoria RAM:** 16GB
- **Almacenamiento:** SSD 500 GB
- **Resolución de pantalla:** 1920 x 1080

La base de datos se ejecuta en modo local, utilizando MySQL Server como backend y MySQL Workbench como interfaz gráfica para su gestión..

3.2. Metodología aplicada en el desarrollo

Aunque el proyecto ha sido desarrollado por una sola persona, se ha seguido una estructura inspirada en los principios ágiles, concretamente el modelo **Scrum** adaptado a un entorno individual.

¿Por qué Scrum?

Scrum permite organizar el trabajo por etapas o sprints, facilitando la planificación, seguimiento y revisión continua del proyecto. Aunque no se realizaron reuniones formales, se adaptó una división semanal de tareas que permitió mantener un enfoque estructurado.

Ventajas del enfoque aplicado:

- Claridad en los objetivos de cada fase
- Flexibilidad para reorganizar tareas según las dificultades encontradas.
- Revisión progresiva del trabajo realizado.

Desventajas identificadas:

- Falta de supervisión directa en las fases intermedias.
- Dificultad para mantener un ritmo constante debido a la carga académica externa.

A continuación muestro el cronograma establecido para el desarrollo de la aplicación.

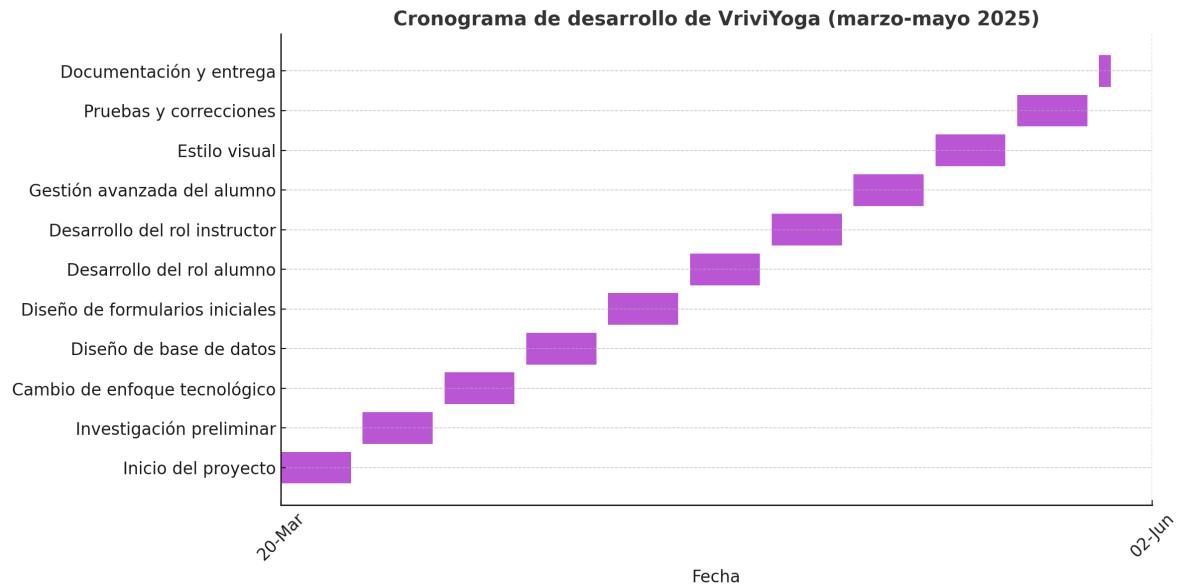


Figura 4: Diagrama de Gantt

3.3. Simulación económica y análisis de recursos

Se ha realizado una simulación de costes estimados para el desarrollo de VriViYoga, incluyendo tanto el tiempo dedicado como los recursos técnicos empleados.

1. Recursos Humanos

Concepto	Estimación
Horas de desarrollo	120 horas
Valor estimado por hora	10 €/hora
Total	1.200 €

2. Recursos técnicos y software

Recurso	Coste estimado
Visual Studio 2022	0 €
MySQL + Workbench	0 €
GitHub	0 €
Ordenador personal	0 €
Servidor local (localhost)	0 €
Total técnico estimado	0 €

Total estimado del proyecto: 1.200 €

4. DISEÑO DE LA APLICACIÓN

Esta sección describe en detalle las funcionalidades, características, restricciones y estructura visual de la aplicación VriViYoga. El diseño responde a los requisitos definidos y busca ofrecer una experiencia funcional, sencilla y coherente con los valores del yoga.

4.1 Necesidades y funcionalidades del sistema

La especificación de requisitos define qué debe cumplir la aplicación para satisfacer las necesidades reales de los alumnos e instructores en una academia de yoga. También se consideran aspectos no funcionales, como usabilidad, rendimiento y compatibilidad.

4.1.1 Funcionalidades previstas por rol

Funcionalidades generales:

1. Sistema de autenticación y registro

- Registro de usuarios.
- Inicio de sesión con validación básica de credenciales.
- Acceso personalizado según el rol seleccionado (alumno o instructor).

2. Gestión de clases (Instructor y Alumno)

- Instructor: puede crear, editar y eliminar clases.
- Alumno: puede visualizar las clases disponibles, reservar y cancelar reservas.

3. Gestión de reservas (Alumno)

- Acceso a listado de clases reservadas.
- Cancelación de reservas realizadas.

4. Visualización de clases creadas (Instructor)

- Listado general de clases propias.
- Visualización de alumnos inscritos por clase.

4.1.2 Interacción entre usuarios y funcionalidades clave

A continuación se muestra el diagrama de casos de uso generado con draw.io que representa la relación entre **Alumno**, **Instructor** y las funcionalidades principales:

A continuación se incluye un escenario detallado del caso de uso:

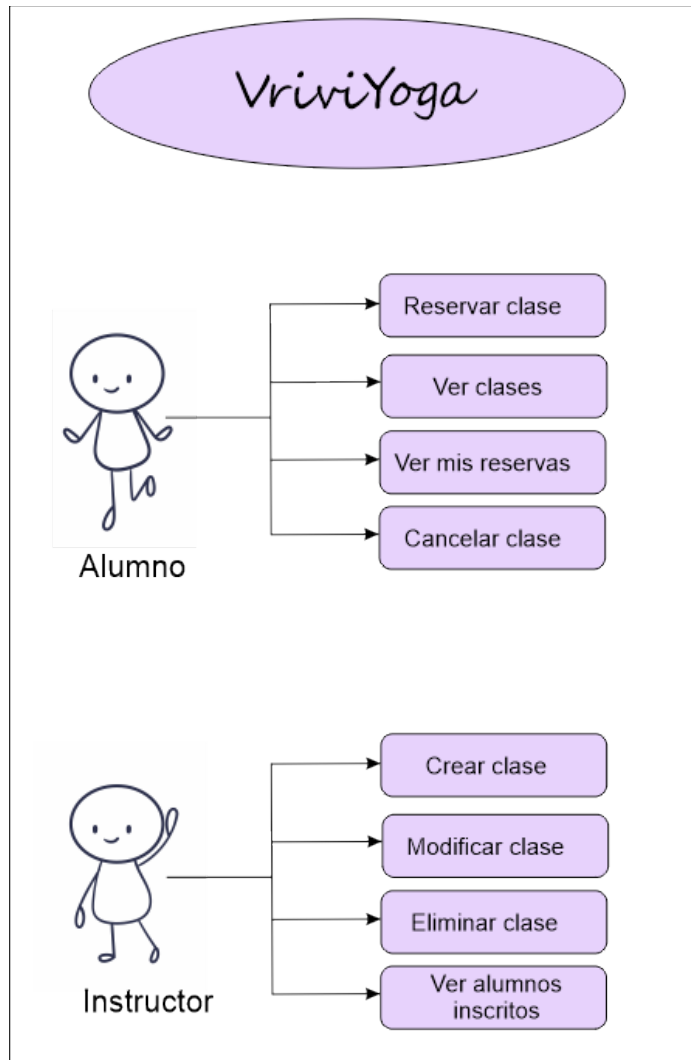


Figura 5. Diagrama de casos de uso de VriviYoga

Tabla de pasos del caso de uso: Alta de clase (Instructor)

Nº	Acción del actor (Instructor)	Respuesta del sistema
1	Accede a la aplicación	Muestra pantalla de inicio de sesión
2	Introduce credenciales	Accede al panel del instructor
3	Pulsa el botón "Crear clase"	Muestra el formulario de nueva clase
4	Rellena descripción, fecha, hora e instructor	El sistema valida los campos
5	Pulsa "Guardar"	Se almacena la clase y aparece mensaje de éxito

4.1.3 Requisitos no funcionales

Categoría	Descripción
Rendimiento	Carga rápida de formularios y reservas (menos de 3 segundos),
Compatibilidad	Solo para escritorio Windows. No compatible con Android/iOS,
Seguridad	Uso de parámetros en SQL para prevenir inyecciones.
Usabilidad	Diseño visual simple, enfocado en accesibilidad y estética relajante,

4.1.4 Interfaces de usuario

La aplicación VriviYoga se diseñó siguiendo principios de simplicidad, organización y coherencia visual. Se utilizaron fondos claros, fuentes suaves y botones bien distribuidos para mantener una interfaz atractiva y funcional.

- Pantalla de inicio de sesión (ver figura 11 en el apartado Anexos)
- Formulario de registro de usuarios (ver figura 12 en el apartado Anexos)
- Panel del alumno (ver figuras 13 y 14 en el apartado Anexos)
- Panel del instructor (ver figuras 15 y 16 en el apartado Anexos)
- Ventana para ver alumnos por clase (instructor) (ver figura 17 en el apartado Anexos)

4.1.5 Gestión de comunicación entre sistema y usuarios

La aplicación VriviYoga no depende de servicios web externos ni APIs. Toda la lógica del sistema se gestiona de manera local mediante formularios programados con C# y conectados a una base de datos MySQL local. Esto facilita su uso sin necesidad de conexión permanente a Internet ni servidores remotos.

4.2 Condicionantes técnicos y límites del desarrollo

Durante el desarrollo se han definido una serie de restricciones necesarias para garantizar la viabilidad del proyecto dentro del entorno académico.

Condicionante	Detalles
Lenguaje de programación	C# como lenguaje principal y compatible con Windows Forms
Metología	Enfoque ágil inspirado en Scrum, con planificación semanal adaptada
Entorno de desarrollo	Visual Studio 2022 y MySQL Workbench en modo local
Compatibilidad	Aplicación para Windows de escritorio; no multiplataforma en esta fase
Librerías externas	Solo librerías estándar de .NET. No se usaron paquetes externos

5. DISEÑO DE LA BASE DE DATOS

5.1 Descripción general de la solución desarrollada

La aplicación *VrivYoga* es un sistema de gestión de reservas de clases en una academia de yoga. Su objetivo principal es permitir a los alumnos visualizar clases disponibles e inscribirse, y a los instructores crear y gestionar dichas clases.

El sistema está desarrollado en C# utilizando Visual Studio 2022 y Windows Forms, y se conecta a una base de datos MySQL diseñada en MySQL Workbench.

Durante el desarrollo, se diseñaron interfaces visuales desde el entorno de desarrollo directamente, sin utilizar herramientas como Figma. Las pantallas principales de la aplicación son:

- Inicio de sesión
- Registro de usuarios
- Vista de clases disponibles
- Gestión de reservas para alumnos
- Gestión de clases para instructores (crear, ver, eliminar)

El sistema se encuentra implementado completamente, con navegación funcional y conexión activa con la base de datos local.

5.2 Definición estructural de los datos del sistema

La base de datos de VriviYoga se compone de tres tablas principales: Usuarios, que contiene la información de todos los usuarios (alumnos e instructores); Clases, que define las clases ofrecidas por la academia y Reservas, que registra las reservas que hace cada alumno para una clase.

Relaciones clave:

- Un usuario puede realizar muchas reservas.
- Una clase puede tener muchas reservas.
- Una reserva vincula un usuario con una clase.

5.2.1 Modelo entidad – relación

El siguiente modelo representa gráficamente las relaciones del sistema:

- usuarios → contiene email, que actúa como Primary Key
- clases → contiene id (clave primaria) y campos como descripcion, fecha, hora, instructor
- reservas → contiene id, alumno_email (FK hacia usuarios.email) y clase_id (FK hacia clases.id)

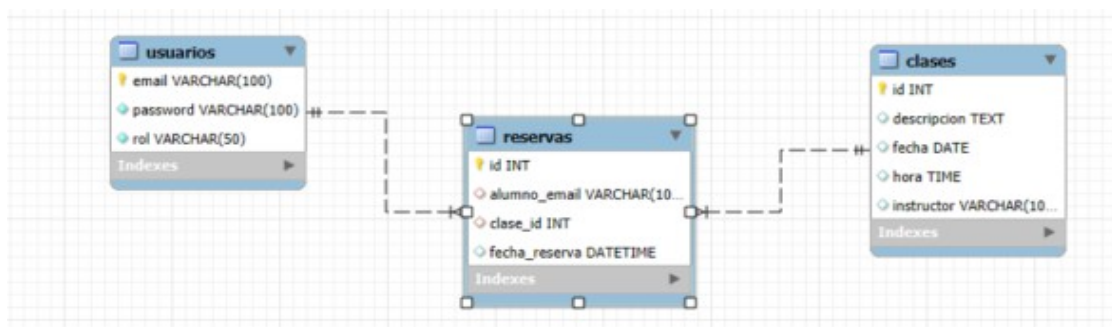


Figura 6: Diagrama de base de datos

5.3 Transformación del modelo entidad-relación a modelo físico

Cada entidad fue transformada en una tabla dentro de MySQL, con claves primarias (PK) y foráneas (FK) para garantizar la integridad.

Tabla	Campos	Relaciones
usuarios	Email (PK), contraseña, rol	-
clases	Id (PK), descripcion, fecha, hora, instructor	Instructor es texto (sin FK por diseño actual)
reservas	Id (PK), alumno_email (FK), clase_id (FK), fecha_reserva	alumno_email → usuarios, clase_id → clases

5.4 Representación gráfica del modelo físico de la base de datos

A continuación se presentan las tres tablas extraídas del MySQL Workbench en su modelo físico:

- Tabla usuarios: email como clave primaria, junto con password y rol.
- Tabla clases: id como clave primaria, junto con fecha, hora, descripción e instructor.
- Tabla reservas: id como PK y alumno_email y clase_id como claves foráneas.

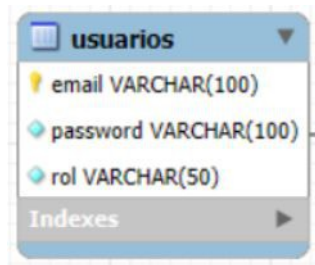


Figura 7: tabla usuarios

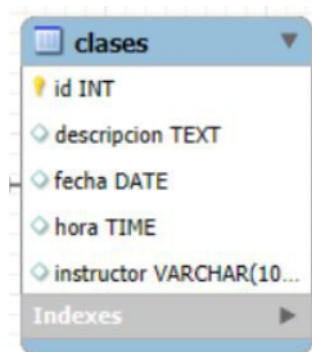


Figura 8: tabla clases



Figura 9: tabla reservas

5.5 Especificación detallada de las estructuras de datos

A continuación se presenta el diccionario de datos de las principales tablas del sistema, que incluye información técnica relevante para el desarrollo e integración del proyecto. Se detallan los campos, tipo de dato, claves, restricciones y descripción funcional de cada columna.

Tabla: usuarios

Columna	Tipo	Clave	Nulo	Descripción
email	VARCHAR(100)	PK	No	Identificador único del usuario
password	VARCHAR(100)		No	Contraseña
rol	VARCHAR(50)		No	Rol (alumno o instructor)

Tabla: clases

Columna	Tipo	Clave	Nulo	Descripción
id	INT	PK	No	Identificador único de la clase
descripcion	TEXT		No	Descripción de la clase
fecha	DATE		No	Fecha de la clase
hora	TIME		No	Hora de la clase
instructor	VARCHAR (100)		No	Instructor responsable (nombre)

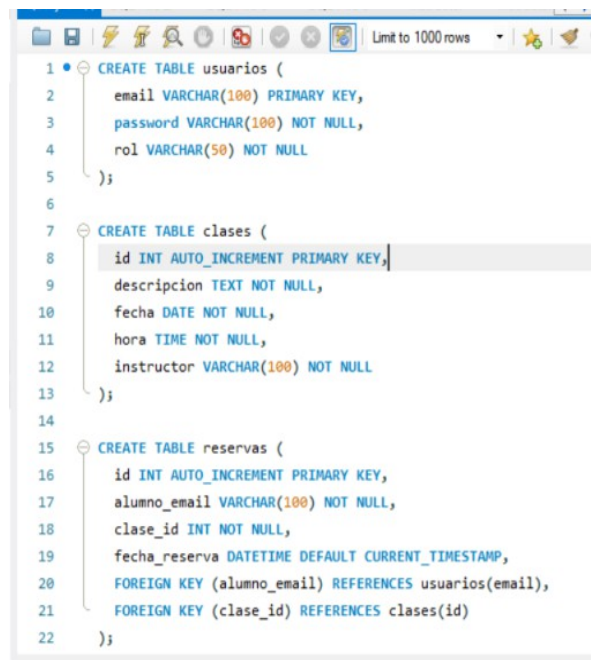
Tabla: reservas

Columna	Tipo	Clave	Nulo	Descripción
id	INT	PK	No	Identificador único de la reserva
alumno_email	VARCHAR(100)	FK	No	Email del alumno que reserva
clase_id	INT	FK	No	Referencia a la clase reservada

fecha_reserva	DATETIME	No	Fecha y hora en la que se realizó la reserva
---------------	----------	----	--

5.6 Script de definición e inserción de datos de la base de datos

A continuación se muestra un fragmento del script SQL utilizado para crear las tablas y las relaciones:



```

1 CREATE TABLE usuarios (
2   email VARCHAR(100) PRIMARY KEY,
3   password VARCHAR(100) NOT NULL,
4   rol VARCHAR(50) NOT NULL
5 );
6
7 CREATE TABLE clases (
8   id INT AUTO_INCREMENT PRIMARY KEY,
9   descripcion TEXT NOT NULL,
10  fecha DATE NOT NULL,
11  hora TIME NOT NULL,
12  instructor VARCHAR(100) NOT NULL
13 );
14
15 CREATE TABLE reservas (
16   id INT AUTO_INCREMENT PRIMARY KEY,
17   alumno_email VARCHAR(100) NOT NULL,
18   clase_id INT NOT NULL,
19   fecha_reserva DATETIME DEFAULT CURRENT_TIMESTAMP,
20   FOREIGN KEY (alumno_email) REFERENCES usuarios(email),
21   FOREIGN KEY (clase_id) REFERENCES clases(id)
22 );

```

Figura 10: Fragmento Script SQL

6. ORGANIZACIÓN FUNCIONAL CÓDIGO FUENTE Y ARQUITECTURA

La aplicación *VviviYoga* ha sido desarrollada como una solución de escritorio utilizando el lenguaje C# bajo la plataforma Windows Forms (WinForms) en Visual Studio 2022, conectada a una base de datos MySQL gestionada con MySQL Workbench.

6.1 Arquitectura del proyecto

El proyecto sigue una estructura modular basada en la separación por formularios (Forms) y responsabilidad por rol del usuario:

Form1.cs

Ventana principal de inicio de sesión. Permite que los usuarios se identifiquen como alumno o instructor.

RegistroForm.cs

Formulario de registro de nuevos usuarios. Gestiona la creación de cuentas con rol y contraseña.

AlumnoForm.cs

- Interfaz específica para el alumno. Permite:
- Ver clases disponibles.
- Reservar clases.
- Consultar clases reservadas.
- Cancelar reservas.

InstructorForm.cs

- Interfaz para el instructor. Desde aquí se puede:
- Crear nuevas clases.
- Ver las clases creadas.
- Modificar o eliminar clases.
- Ver los alumnos inscritos a una clase.

6.2 Estructura del código fuente

El código está organizado en archivos independientes según su funcionalidad. Cada formulario posee:

- Un archivo .cs que contiene la lógica (eventos, conexión a base de datos, control de botones, etc.)
- Un archivo .Designer .cs que contiene los elementos visuales del formulario.

Un archivo .resx de recursos asociados (si aplica).

6.3 Conexión a la base de datos

La conexión con MySQL se realiza mediante `MySQL.Data.MySqlClient`. Cada formulario establece su propia conexión local usando una cadena de conexión como:

```
string connectionString =  
"server=localhost;user=root;password=admin1234;database=vriviyoga;"
```

Figura 11: Conexión base de datos (código)

Esta cadena es reutilizada para consultas SELECT, inserciones INSERT INTO, eliminaciones DELETE y actualizaciones UPDATE según corresponda.

6.4 Interacción entre formularios

- Desde Form1.cs, tras validar credenciales, se redirige al formulario AlumnoForm o InstructorForm según el campo rol de la tabla de usuarios.
- El formulario de registro se abre desde el link Crear cuenta en Form1.

6.5 Lógica de funcionamiento

- La lógica está encapsulada en los eventos Click de los botones. Ejemplos:
- btnLogin_Click: verifica credenciales e inicia sesión.
- btnRegistrar_Click: crea un nuevo usuario en la base de datos.
- btnActualizar_Click (Alumno): carga las clases disponibles.
- btnCrear_Click (Instructor): inserta una nueva clase.
- Cada consulta está parametrizada para evitar inyecciones y errores.

6.6 Navegación y experiencia visual

- La aplicación usa formularios independientes, no navegación embebida.
- Cada rol tiene su interfaz aislada.
- Se han personalizado los formularios con:
- Fuentes tipo manuscrita para el título.
- Imágenes de fono relacionadas con el yoga.
- Botones estilizados.
- Colores suaves (rosados, lilas) para transmitir calma y conexión con la temática del bienestar.

6.7 Capturas del código fuente

A continuación, se insertan capturas de código representativo desde Visual Studio:

- Validación de login.
- Registro de usuario.
- Reserva de clase.
- Creación o eliminación de clases.

1. Inicio de sesión → Form1.cs (btnLogin_Click) → Valida usuarios, conecta con base de datos y abre formulario según rol.

```
private void btnLogin_Click(object sender, EventArgs e)
{
    string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";
    string email = textEmail.Text;
    string password = textPassword.Text;

    using (MySqlConnection conn = new MySqlConnection(connectionString))
    {
        try
        {
            conn.Open();
            string query = "SELECT * FROM usuarios WHERE email = @correo AND
password = @clave";
            MySqlCommand cmd = new MySqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@correo", email);
            cmd.Parameters.AddWithValue("@clave", password);

            // Opcional para depurar
            // MessageBox.Show("Email: " + email + "\nPassword: " + password);

            MySqlDataReader reader = cmd.ExecuteReader();
            if (reader.Read())
            {
                string rol = reader["rol"].ToString();

                // ✓ Guardar email para usarlo en AlumnoForm
                EmailActivo = email;

                if (rol == "alumno")
                {
                    AlumnoForm alumno = new AlumnoForm();
                    alumno.Show();
                    this.Hide();
                }
                else if (rol == "instructor")
                {
                    InstructorForm instructor = new InstructorForm();
                    instructor.Show();
                    this.Hide();
                }
            }
        }
        else
        {

```



```

        lblMensaje.Text = "Correo o contraseña incorrectos.";
        lblMensaje.ForeColor = System.Drawing.Color.Red;
    }
}
catch (Exception ex)
{
    MessageBox.Show("Error de conexión: " + ex.Message);
}
}
}

```

Figura 12 : Inicio de sesión (dódigo)

2. Registro de usuario → RegistroForm.cs (btnRegistrar_Click) → Guarda nuevo usuario con rol, correo y contraseña.

```

private void btnRegistrar_Click(object sender, EventArgs e)
{
    string email = txtEmail.Text;
    string password = txtPassword.Text;
    string rol = cmbRol.Text;

    string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";

    using (MySqlConnection conn = new MySqlConnection(connectionString))
    {
        try
        {
            conn.Open();
            string query = "INSERT INTO usuarios (email, password, rol)
VALUES (@correo, @clave, @rol)";
            MySqlCommand cmd = new MySqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@correo", email);
            cmd.Parameters.AddWithValue("@clave", password);
            cmd.Parameters.AddWithValue("@rol", rol);

            int filasAfectadas = cmd.ExecuteNonQuery();

            if (filasAfectadas > 0)
            {
                this.lblMensaje.Text = "Usuario registrado correctamente.";
                this.lblMensaje.ForeColor = System.Drawing.Color.Green;
            }
            else
            {
                lblMensaje.Text = "No se pudo registrar.";
                lblMensaje.ForeColor = System.Drawing.Color.Red;
            }
        }
    }
}

```

```

        catch (Exception ex)
        {
            this.lblMensaje.Text = "Error: " + ex.Message;
            this.lblMensaje.ForeColor = System.Drawing.Color.Red;
        }
    }
}

```

Figura 13: Registro de usuario (código)

3. Reserva clase → AlumnoForm.cs (dvgClases_CellContentClick) → Botón dentro de DataGridView para insertar reserva.

```

private void dvgClases_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    if (dvgClases.Columns[e.ColumnIndex].Name == "Reservar" && e.RowIndex >= 0)
    {
        int claseId =
Convert.ToInt32(dvgClases.Rows[e.RowIndex].Cells["id"].Value);
        string emailAlumno = Form1.EmailActivo;

        string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";
        using (MySqlConnection conn = new MySqlConnection(connectionString))
        {
            try
            {
                conn.Open();
                string query = "INSERT INTO reservas (alumno_email, clase_id)
VALUES (@alumno, @clase)";
                MySqlCommand cmd = new MySqlCommand(query, conn);
                cmd.Parameters.AddWithValue("@alumno", emailAlumno);
                cmd.Parameters.AddWithValue("@clase", claseId);
                cmd.ExecuteNonQuery();
                MessageBox.Show("¡Clase reservada con éxito!");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error al reservar: " + ex.Message);
            }
        }
    }

    if (dvgClases.Columns[e.ColumnIndex].Name == "Cancelar" && e.RowIndex >= 0)
    {
        string emailAlumno = Form1.EmailActivo;
    }
}

```

```

        string tituloClase =
dvgClases.Rows[e.RowIndex].Cells["Título"].Value.ToString();

        int claseId = ObtenerClaseIdPorTitulo(tituloClase);
        if (claseId == -1)
        {
            MessageBox.Show("Error: no se encontró la clase.");
            return;
        }

        string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";
        using (MySqlConnection conn = new MySqlConnection(connectionString))
        {
            try
            {
                conn.Open();
                string query = "DELETE FROM reservas WHERE alumno_email = @alumno
AND clase_id = @clase";
                MySqlCommand cmd = new MySqlCommand(query, conn);
                cmd.Parameters.AddWithValue("@alumno", emailAlumno);
                cmd.Parameters.AddWithValue("@clase", claseId);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Reserva cancelada.");
                btnReservasAlumnoTest_Click(null, null); // recarga la vista de
reservas
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error al cancelar: " + ex.Message);
            }
        }
    }
}

```

Figura 14: Reserva de clase (código)

4. Visualización de reservas (alumnos) → AlumnoForm.cs

(btnREservasAlumnoTest_Click) → Consulta SQL que muestra clases reservadas por el alumno.

```

private void btnReservasAlumnoTest_Click(object sender, EventArgs e)
{
    string emailAlumno = Form1.EmailActivo;
    string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";

    using (MySqlConnection conn = new MySqlConnection(connectionString))
    {

```

```

try
{
    conn.Open();
    string query = @"
        SELECT c.descripcion AS 'Descripción', c.fecha AS 'Fecha', c.hora
        AS 'Hora', c.instructor AS 'Instructor'
        FROM reservas r
        JOIN clases c ON r.clase_id = c.id
        WHERE r.alumno_email = @correo";

    MySqlCommand cmd = new MySqlCommand(query, conn);
    cmd.Parameters.AddWithValue("@correo", emailAlumno);

    MySqlDataAdapter adapter = new MySqlDataAdapter(cmd);
    DataTable tabla = new DataTable();
    adapter.Fill(tabla);
    dgvClases.DataSource = tabla;

    if (!dvgClases.Columns.Contains("Cancelar"))
    {
        DataGridViewButtonColumn btnCancelar = new
DataGridViewButtonColumn();
        btnCancelar.Name = "Cancelar";
        btnCancelar.Text = "Cancelar";
        btnCancelar.UseColumnTextForButtonValue = true;
        dvgClases.Columns.Add(btnCancelar);
    }
}
catch (Exception ex)
{
    MessageBox.Show("Error al cargar tus reservas: " + ex.Message);
}
}
}

```

Figura 15. Visualización de reservas (alumnos) (código)

5. Creación de clase (instructor) → InstructorForm.cs (btnCrear_Click) → Inserta clase con descripción, fecha, hora e instructor.

```

private void btnCrear_Click(object sender, EventArgs e)
{
    // Validación
    if (string.IsNullOrEmpty(txtDescripcion.Text) ||
string.IsNullOrEmpty(txtInstructor.Text))
    {
        MessageBox.Show("Por favor, rellena todos los campos requeridos.");
        return;
    }
}

```

```

    }

    string descripcion = txtDescripcion.Text;
    string fecha = dtpFecha.Value.ToString("yyyy-MM-dd");
    string hora = dtpHora.Value.ToString("HH:mm:ss");
    string instructor = txtInstructor.Text;

    string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";

    using (MySqlConnection conn = new MySqlConnection(connectionString))
    {
        try
        {
            conn.Open();
            string query = "INSERT INTO clases (descripcion, fecha, hora,
instructor) VALUES (@descripcion, @fecha, @hora, @instructor)";
            MySqlCommand cmd = new MySqlCommand(query, conn);
            cmd.Parameters.AddWithValue("@descripcion", descripcion);
            cmd.Parameters.AddWithValue("@fecha", fecha);
            cmd.Parameters.AddWithValue("@hora", hora);
            cmd.Parameters.AddWithValue("@instructor", instructor);

            int resultado = cmd.ExecuteNonQuery();

            if (resultado > 0)
            {
                lblMensaje.Text = "Clase creada correctamente.";
                lblMensaje.ForeColor = System.Drawing.Color.Green;
                CargarClases();
            }
            else
            {
                lblMensaje.Text = "No se pudo crear la clase.";
                lblMensaje.ForeColor = System.Drawing.Color.Red;
            }
        }
        catch (Exception ex)
        {
            lblMensaje.Text = "Error: " + ex.Message;
            lblMensaje.ForeColor = System.Drawing.Color.Red;
        }
    }
}

```

Figura 16: Creación de clase (instructor) (código)

6. Visualización de clases (instructor) → InstructorForm.cs (CargarClases()) → Muestra todas las clases creadas por el instructor.

```

private void btnVerClasesInstructor_Click(object sender, EventArgs e)
{
    CargarClases();
}

private void btnModificarClase_Click(object sender, EventArgs e)
{
    if (dvgInstructorClases.SelectedRows.Count > 0)
    {
        int id =
Convert.ToInt32(dvgInstructorClases.SelectedRows[0].Cells["id"].Value);
        string titulo = txtDescripcion.Text;
        string descripcion = txtDescripcion.Text;
        string fecha = dtpFecha.Value.ToString("yyyy-MM-dd");
        string hora = dtpHora.Value.ToString("HH:mm:ss");

        string connectionString =
"server=localhost;user=root;password=admin1234;database=vriviyoga;";
        using (MySqlConnection conn = new MySqlConnection(connectionString))
        {
            try
            {
                conn.Open();
                string query = "UPDATE clases SET titulo = @titulo, descripcion =
@descripcion, fecha = @fecha, hora = @hora WHERE id = @id";
                MySqlCommand cmd = new MySqlCommand(query, conn);
                cmd.Parameters.AddWithValue("@titulo", titulo);
                cmd.Parameters.AddWithValue("@descripcion", descripcion);
                cmd.Parameters.AddWithValue("@fecha", fecha);
                cmd.Parameters.AddWithValue("@hora", hora);
                cmd.Parameters.AddWithValue("@id", id);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Clase modificada correctamente.");
                CargarClases();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error al modificar: " + ex.Message);
            }
        }
    }
    else
    {
        MessageBox.Show("Selecciona una clase para modificar.");
    }
}

```

Figura 17: Visualización de clases (instructor) (código)

7. COMPROBACIONES FUNCIONALES INICIALES DE LA APLICACIÓN

Durante el desarrollo de la aplicación *VriviYoga*, se han realizado pruebas funcionales manuales para validar el correcto funcionamiento de las funcionalidades implementadas. Estas comprobaciones se llevaron a cabo directamente en el entorno de ejecución de Visual Studio 2022, conectado a una base de datos local en MySQL Workbench.

Las pruebas se centraron en los siguientes aspectos clave:

- Inicio de sesión: Se comprobó que el sistema permite acceder correctamente con credenciales válidas. Se valida el acceso diferenciado entre usuarios con rol de alumno o instructor, redirigiéndolos a la interfaz correspondiente.
- Registro de usuarios: Se probó el formulario de registro para asegurar la creación correcta de nuevos usuarios, almacenando los datos en la tabla usuarios con su rol asignado.
- Reserva de clases (alumno): Los alumnos pueden visualizar clases disponibles y realizar reservas mediante un botón específico en la tabla. Se validó que estas reservas se almacenan correctamente en la tabla reservas.
- Visualización de reservas (alumno): El usuario puede consultar exclusivamente sus clases reservadas, filtradas por su correo electrónico.
- Creación de clases (instructor): Los instructores pueden añadir nuevas clases con sus respectivos campos (descripción, fecha, hora e instructor), las cuales se guardan correctamente en la base de datos.
- Modificación y eliminación de clases: Se implementaron opciones para que el instructor pueda modificar la descripción de una clase o eliminarla, comprobando que ambas acciones actualizan la base de datos sin errores.
- Visualización de alumnos inscritos (instructor): Se permite consultar qué alumnos han reservado clases específicas, mediante una consulta relacional entre las tablas usuarios, clases y reservas.
- Navegación entre formularios: La aplicación permite moverse fluidamente entre las pantallas de inicio, registro, panel del alumno y panel del instructor, sin errores de navegación.

Errores encontrados y solución

Durante el proceso de pruebas, se identificaron y resolvieron diversos errores comunes:

- Errores CS0103 / CS1061: Relacionados con elementos visuales no declarados o mal nombrados en el formulario.

- Errores de integridad referencial en MySQL: Problemas con claves foráneas en la tabla reservas, que fueron corregidos con relaciones adecuadas entre usuarios y clases.
- Problemas de carga en formularios: Errores relacionados con el DataGridView no vinculado correctamente al nombre declarado, solucionado renombrando y asociando correctamente en el Designer.

8. DIFICULTADES DETECTADAS Y POSIBLES MEJORAS

Durante el desarrollo del proyecto *VriviYoga*, se presentaron diversas dificultades tanto a nivel técnico como organizativo, especialmente considerando que fue realizado por una sola persona con conocimientos limitados en programación al inicio del trabajo.

8.1 Dificultades encontradas

- **Errores en Visual Studio:** A lo largo de desarrollo se detectaron errores frecuentes como CS5001 (“el programa no contiene un método Main”) y CS1061 (“no contiene una definición para InitializeComponent”), que dificultaron el arranque y la compilación del proyecto. Estos errores se solucionaron con la ayuda de depuración línea por línea y revisión de los formularios de diseño.
- **Integración con MySQL Workbench:** Se presentaron dificultades al establecer relaciones foráneas, especialmente el Error 1452, derivado de discrepancias entre los identificadores reales y los requeridos. Fue necesario realizar ajustes manuales en los datos y revisar la estructura del modelo entidad-relación para garantizar su integridad:
- **Visualización del diagrama E-R:** Inicialmente no se visualizaban correctamente las relaciones entre tablas. Esto obligó a revisar las claves primarias y foráneas en el modelo físico para que aparecieran correctamente en el EER Diagram.
- **Lentitud del entorno:** La ejecución del proyecto en Visual Studio, especialmente en modo de depuración, presentó cierta lentitud. Aunque no se usó el emulador Android por trabajar con Windows Forms, algunas acciones dentro del entorno requerían tiempos de carga prolongados.
- **Aprendizaje simultáneo de herramientas:** Fue necesario aprender desde 0 varias tecnologías como C#, Visual Studio, MySQL Workbench y los principios de arquitectura por

capas. Esta curva de aprendizaje requirió un esfuerzo adicional y fue uno de los principales desafíos del proyecto.

8.2 Posibles mejoras futuras

Planificación de tiempos más rigurosa: Distribuir el trabajo de forma más equitativa en el tiempo habría permitido una mejor gestión del estrés y mayor margen para pruebas y documentación.

Documentación paralela al desarrollo: Haber redactado la memoria de manera progresiva, conforme se completaban partes del proyecto, habría facilitado la entrega y reducido los bloqueos en la fase final.

Despliegue en entorno real: Como mejora futura, se plantea la posibilidad de realizar un despliegue real del sistema, publicando la aplicación para su instalación en equipos reales de academias o incluso preparar una versión para Android con .NET MAUI.

Pruebas automatizadas: La inclusión de pruebas unitarias y de integración ayudaría a validar los principales flujos de trabajo automáticamente, aumentando la fiabilidad del sistema.

9. PLANIFICACIÓN TEMPORAL POR FASES Y DESARROLLO

La planificación del proyecto *VriViYoga* se ha dividido en fases semanales, permitiendo un seguimiento progresivo y estructurado del desarrollo. A continuación, se presenta la tabla comparativa entre la duración estimada inicialmente y el tiempo real invertido en cada fase:

Fase/ Bloque	Tarea principal	Duración estimada	Duración real
1. Introducción	Redacción del planteamiento, justificación y análisis del sector	1 semana	1 semana
2. Definición del proyecto	Establecimiento de objetivos, contexto y público destinatario	1 semana	1 semana
3. Análisis de la aplicación	Selección de herramientas, metodología de trabajo y presupuesto	1 semana	1 semana

4. Diseño de la aplicación	Casos de uso, interfaz visual, navegación por roles	2 semanas	3 semanas
5. Base de datos	Diseño entidad-relación, creación en MySQL Workbench, pruebas	1 semana	2 semanas
6. Organización del código	Estructura de formularios, eventos y navegación	0.5 semanas	1 semana
7. Pruebas y validaciones	Verificación manual de las funcionalidades	0.5 semanas	1 semana
8. Dificultades y mejoras	Registro de errores encontrados y propuestas de solución	0.5 semanas	0.5 semanas
9. Temporalización	Elaboración de esta tabla de planificación y Gantt	0.5 semanas	0.5 semanas
10. Despliegue de la aplicación	Preparación para entrega, compresión de archivos, pruebas finales	1 semana	1 semana
11. Manual de usuario	Redacción guiada e ilustrada con capturas reales de la app	1 semana	1 semana
12. Mantenimiento	Definición de posibles futuras mejoras y soporte	0.5 semanas	0.5 semanas

10. DESPLIEGUE DE LA APLICACIÓN

El despliegue de la aplicación *VviviYoga* se ha realizado en un entorno local de desarrollo utilizando Visual Studio 2022 con formularios Windows Forms en lenguaje C#, y una base de datos MySQL gestionada mediante MySQL Workbench.

Debido a que la aplicación está pensada como una herramienta de escritorio para academias de yoga, el entorno de pruebas ha sido una instalación local sobre Windows, sin distribución en dispositivos móviles ni en la nube.

Entorno de desarrollo y pruebas:

Componente	Detalle
Sistema operativo	Windows 11 Home
IDE	Visual Studio 2022 Community Edition
Lenguaje de programación	C# (Windows Forms)
Motor de base de datos	MySQL Server (local)
Gestor de base de datos	MySQL Workbench
Tipo de conexión	Localhost (conexión directa al servidor MySQL local)
Método de despliegue	Ejecución directa desde Visual Studio (modo debug)

Las pruebas funcionales se han realizado ejecutando la aplicación desde el entorno de Visual Studio, permitiendo comprobar las funcionalidades tanto para alumnos como para instructores: inicio de sesión, registro, reserva, creación y modificación de clases, entre otras.

Requisitos mínimos para ejecución

- Ordenador con Windows 10 o superior
- .NET Framework instalado (versión requerida por Visual Studio 2022)
- MySQL Server en local con las tablas configuradas según el modelo del proyecto.
- Acceso a Visual Studio o compilación previa del proyecto en formato .exe.

Limitaciones encontradas

- No se ha generado aún un archivo ejecutable (.exe) para instalación independiente.
- La aplicación requiere que la base de datos esté correctamente configurada y funcionando localmente.
- No se ha implementado un mecanismo de despliegue automático ni en red, ni en la nube.

Futuras opciones de despliegue

- Compilar la aplicación en un archivo ejecutable .exe para facilitar su uso sin Visual Studio.
- Incluir una opción de configuración automática de la base de datos o adaptarla a una conexión remota.
- Distribuir la aplicación a través de una carpeta comprimida .zip con instrucciones de uso.

11. MANUAL DE USUARIO

Este manual está dirigido a los futuros usuarios de la aplicación *VriviYoga* que puede ser utilizada por alumnos e instructores de academias de yoga. La interfaz está diseñada para ser clara, accesible y adaptada a las necesidades reales del entorno educativo.

11.1 Requisitos para el uso de la aplicación

- Sistema operativo Windows 10 o superior
 - La base de datos MySQL debe de estar activa en localhost con los datos configurados.
 - El usuario debe ejecutar el archivo .exe de VriviYoa o ejecutando desde Visual Studio.

11.2 Inicio de sesión

1. Al abrir la aplicación aparece la pantalla de inicio de sesión.
2. El usuario debe introducir su correo electrónico y contraseña.

Según su rol, accederá a una interfaz distinta:

- Alumno: verá clases disponibles, podrá reservar y cancelar reservas.
- Instructor: gestionará clases que imparte (crear, modificar, eliminar, consultar alumnos inscritos).

Si no tiene cuenta, puede hacer clic en el enlace “Crear cuenta”.

11.3 Registro de nuevos usuarios

1. Desde la pantalla inicial, pulsar el enlace “Crear cuenta”.
2. Se abrirá un formulario con los siguientes campos:
 - Correo electrónico
 - Contraseña
 - Rol (alumno o instructor)
3. Al hacer clic en “Registrarse” el sistema validará los datos y almacenará el nuevo usuario.

11.4 Funcionalidades del rol Alumno

- Ver clases disponibles: el botón “Actualizar clases” muestra todas las clases del sistema.
- Reservar clase: cada fila tiene un botón “Reservar”. Al hacer clic, se guarda la reserva en la base de datos.

- Ver mis reservas: muestra un listado exclusivo de las clases que ha reservado ese alumno.
- Cancelar reserva en la vista reservas, cada fila incluye un botón “Cancelar”.

11.5 Funcionalidades del rol instructor

- Crear clase: introducir descripción, fecha, hora e instructor, luego pulsar “Crear clase”.
- Ver clases creadas: botón para listar todas las clases asignadas a ese instructor.
- Modificar clase: seleccionar una fila de la tabla y modificar los datos.
- Eliminar clase: seleccionar y pulsar el botón correspondiente para eliminarlas.
- Ver alumnos inscritos: muestra una tabla con los alumnos que han reservado cada clase.

11.6 Consejos de uso

- Al cerrar una ventana, no se cierra la sesión automáticamente. Es recomendable cerrar sesión manualmente si se implementa esta opción.
- Comprobar que la base de datos esté activa antes de iniciar la aplicación.
- Evitar dejar campos vacíos para prevenir errores de validación.

12. MANTENIMIENTO

El mantenimiento de la aplicación VviviYoga es fundamental para asegurar su correcto funcionamiento a lo largo del tiempo, así como su capacidad de adaptarse a nuevas necesidades o entornos tecnológicos. A continuación se detallan las estrategias y recomendaciones para el mantenimiento técnico y funcional del sistema.

12.1 Tipos de mantenimiento

1. Mantenimiento correctivo

Consiste en corregir errores detectados durante el uso real de la aplicación. Algunos ejemplos podrían incluir:

- Fallos en la validación de campos vacíos.

- Problemas de conexión con la base de datos.
- Errores en la visualización de datos en las tablas.

2. Mantenimiento adaptativo

Se refiere a los ajustes necesarios para que la aplicación funcione correctamente ante cambios en el entorno, tales como:

- Actualizaciones del sistema operativo Windows.
- Cambios en la versión de MySQL O Visual Studio.
- Modificaciones en la estructura de la base de datos.

3. Mantenimiento evolutivo

Implica añadir nuevas funcionalidades o mejorar las existentes. Algunas ideas para futuras versiones de VriviYoga:

- Envío automático de recordatorios por correo electrónico.
- Incorporación de estadísticas para instructores (clases impartidas, alumnos inscritos).
- Integración con plataformas móviles o desarrollo de App Android.

4. Mantenimiento preventivo

Consiste en realizar acciones para evitar errores futuros, como:

- Revisar el código periódicamente para eliminar dependencias innecesarias.
- Realizar copias de seguridad regulares de la base de datos.
- Documentar las funciones más importantes para facilitar futuras intervenciones.

12.2 Herramientas y buenas prácticas

- Visual Studio 2022: IDE utilizado para depurar, actualizar y mantener el código fuente.
- MySQL Workbench: permite gestionar la base de datos, realizar copias de seguridad y aplicar cambios estructurales.

- Control de versiones (opcional en futuras versiones): se recomienda el uso de GitHub para gestionar los cambios en el código y trabajar en equipo si se amplía el proyecto.

12.3 Documentación de soporte

La aplicación incluye una estructura clara de archivos y nombres de formularios, facilitando que otros desarrolladores puedan intervenir.

13. CONCLUSIONES

El desarrollo de la aplicación *VriviYoga* ha sido enriquecedor para mi ya que me ha permitido aplicar de forma práctica los conocimientos adquiridos durante el ciclo DAM. A lo largo del proyecto se han superado distintos retos técnicos y organizativos, logrando construir una aplicación funcional para la gestión de clases de yoga.

Entre los logros más relevantes destacan:

- La creación de una interfaz clara con roles diferenciados: alumno e instructor.
- La implementación de funciones esenciales como registro, login, reserva, cancelación y gestión de clases.
- La correcta conexión con una base de datos relacional en MySQL, garantizando persistencia y organización de los datos.

La elaboración de una memoria técnica completa que documenta el proceso de desarrollo. *VriviYoga* demuestra la capacidad de transformar una necesidad real en una solución digital y marca un importante avance en la formación profesional del desarrollador.

14. GLOSARIO DE SIGLAS Y ABREVIATURAS

Sigla / Abreviatura	Significado
DAM	Desarrollo de Aplicaciones Multiplataforma
IDE	Entorno de Desarrollo Integrado
UI / GUI	Intefaz de Usuario / Interfaz Gráfica de Usuario
SQL	Structured Query Language (Lenguaje de Consulta Estructurada)
PK	Primary Key (Clave primaria)
FK	Foreign Key (Clave foránea)

MySQL	Sistema de gestión de bases de datos relacional
MAUI	.NET Multi-plataform App UI (No usado en versión final de este proyecto)
VS / Visual Studio	Visual Studio, entorno de desarrollo utilizado para programar la aplicación
ER /E-R	Entidad – Relación (modelo de diseño de bases de datos)
TFG	Trabajo de Fin de Grado

15. BIBLIOGRAFÍA

Sitios web y documentación oficial

- Microsoft Docs – Documentación de Visual Studio
<https://learn.microsoft.com/es-es/visualstudio/>
- MySQL Documentation – Referencia oficial de MySQL
<https://dev.mysql.com/doc/>
- MySQL Workbench – Manual de usuario
<https://dev.mysql.com/doc/workbench/en/>
- C# Programming Guide – Microsoft Learn
<https://learn.microsoft.com/en-us/dotnet/csharp/>
- Stack Overflow – Comunidad de desarrolladores
<https://stackoverflow.com/>
- GitHub – Repositorio para control de versiones
<https://github.com/>

Artículos y contexto del sector

- Statista (2024). Número de practicantes de yoga a nivel mundial.
<https://www.statista.com/>
- Yoga Journal. Beneficios del yoga en el bienestar físico y mental.
<https://www.yogajournal.com/>

16. ANEXOS

A. Pantalla de inicio de sesión



The login screen features a purple background. In the top left, the text "VviviYoga" is written in a stylized, handwritten font. To the right of the text is a black silhouette of a person in a yoga pose, with their arms raised and hands joined at the tips, forming a tree-like structure with many leaves. Below the text "Correo electrónico:" is a white input field. Below the text "Contraseña:" is another white input field. To the right of these fields is a purple button with the text "Iniciar sesión" in white. Below the button is a blue link that says "Crear Cuenta".

Figura 18: Inicio de sesión (diseño visual)

B. Pantalla de registro de usuarios



The registration screen features a purple background. In the top right, the text "¡Únete a VviviYoga!" is written in a stylized, handwritten font. To the left of the text is a black silhouette of a person in a yoga pose, with their arms raised and hands joined at the tips, forming a tree-like structure with many leaves. Below the text "Correo electrónico:" is a white input field. Below the text "Contraseña:" is another white input field. To the right of these fields is a purple button with the text "Registrarse" in white. Below the button is a white dropdown menu with the text "Rol:" to its left.

Figura 19: Registro de usuario (diseño visual)

C. Pantallas de visualización de reservas (alumno)



	id	Descripción	Fecha	Hora	Instructor
▶	1	Clase básica de i...	10/06/2024	09:00:00	Tomás Sánchez
	2	Ejercicios de equi...	11/06/2024	11:00:00	Rocío Pérez
	3	Yoga prenatal	01/06/2024	09:00:00	Iris Hernandez
	4	Clase de relajació...	13/06/2024	19:00:00	Tomás Sánchez
	5	Serie dinámica tr...	14/06/2024	07:30:00	Rocío Pérez
	6	Clase enfocada e...	15/06/2024	17:00:00	Iris Hernández
	7	Transiciones sua...	16/06/2024	08:00:00	Tomás Sánchez
	8	Adaptado para p...	17/06/2024	10:30:00	Rocío Pérez

Actualizar clases Ver mis reservas

Figura 20: Visualización de reservas 1 (alumnos) (diseño visual)



	Descripción	Fecha	Hora	Instructor	Reservar
▶	Clase básica de i...	10/06/2024	09:00:00	Tomás Sánchez	Reservar
	Ejercicios de equi...	11/06/2024	11:00:00	Rocío Pérez	Reservar
	Yoga prenatal	01/06/2024	09:00:00	Iris Hernandez	Reservar
	Clase de relajació...	13/06/2024	19:00:00	Tomás Sánchez	Reservar
	Serie dinámica tr...	14/06/2024	07:30:00	Rocío Pérez	Reservar
	Clase enfocada e...	15/06/2024	17:00:00	Iris Hernández	Reservar
	Transiciones sua...	16/06/2024	08:00:00	Tomás Sánchez	Reservar
	Adaptado para p...	17/06/2024	10:30:00	Rocío Pérez	Reservar

Actualizar clases Ver mis reservas

Figura 21: Visualización de reservas 2 (alumno) (diseño visual)

D. Pantallas de creación de clase (instructor)

Descripción

jueves, 29 de mayo de 2025 16:22:37

Nombre del instructor **Ver mis clases**

	id	descripcion	fecha	hora
▶	1	Clase básica de i...	10/06/2024	09:00:00
	2	Ejercicios de equi...	11/06/2024	11:00:00
	3	Yoga prenatal	01/06/2024	09:00:00
	4	Clase de relajació...	13/06/2024	19:00:00
	5	Serie dinámica tr	14/06/2024	07:30:00

Crear clase
Modificar
Eliminar
Ver alumnos

Figura 22: Visualización de creación de clase 1 (instructor) (diseño visual)

Descripción

jueves, 29 de mayo de 2025 16:22:37

Nombre del instructor **Ver mis clases**

	descripcion	fecha	hora	instructor
▶	ase básica de i...	10/06/2024	09:00:00	Tomás Sánchez
	ercicios de equi...	11/06/2024	11:00:00	Rocío Pérez
	oga prenatal	01/06/2024	09:00:00	Iris Hernandez
	ase de relajació...	13/06/2024	19:00:00	Tomás Sánchez
	arie dinámica tr	14/06/2024	07:30:00	Rocío Pérez

Crear clase
Modificar
Eliminar
Ver alumnos

Figura 23: Visualización de creación de clase 2 (instructor) (diseño visual)

E. Pantalla de clases (instructor)

Descripción

jueves , 29 de mayo de 2025 16:29:09

Nombre del instructor **Ver mis clases**

	id	descripcion	fecha	hora
	4	Clase de relajació...	13/06/2024	19:00:00
	5	Serie dinámica tr...	14/06/2024	07:30:00
▶	6	Clase enfocada e...	15/06/2024	17:00:00
	7	Transiciones sua...	16/06/2024	08:00:00
	8	Adaptado para n...	17/06/2024	10:30:00

Crear clase
Modificar
Eliminar
Ver alumnos

Alumnos inscritos a esta clase ✕

laura@example.com

Aceptar

Figura 24: Visualización de alumnos inscritos a clases (instructor) (diseño visual)

DAM: VriviYoga

Virginia Ruiz Escaño