

MÓDULO DE PROYECTO



**ALONSO
DE AVELLANEDA**
Alonso de Avellaneda

**Técnico Superior en Desarrollo de
Aplicaciones Multiplataforma**

MediGest Pro



Autor: Manuel Sánchez Romero
Autor: Ana Anastasia Bratkiv Bratkiv
Autor: María Martín Tadeo

Profesor tutor: Javier Sanz Rodríguez

Alcalá de Henares, diciembre de 2025

MÓDULO DE PROYECTO

MÓDULO DE PROYECTO



Técnico Superior en Desarrollo de Aplicaciones Multiplataforma

MediGest Pro

Autor: Manuel Sánchez Romero
Autor: Ana Anastasia Bratkiv Bratkiv
Autor: María Martín Tadeo

Profesor tutor: Javier Sanz Rodríguez

Alcalá de Henares, diciembre de 2025

MÓDULO DE PROYECTO

Contenido

INTRODUCCIÓN	17
ESTUDIO DEL MERCADO	19
Panorama Estratégico y Conceptos Fundamentales.....	19
Benchmarking Funcional y Estándares de Mercado	19
Desafíos Críticos: Seguridad y Arquitectura	21
Conclusión Estratégica	22
TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS.....	23
FrameWork	23
Lenguaje de programación.....	23
Base de datos.....	24
Librerías.....	25
IDE	25
Herramientas de diseño.....	26
Control de versiones	27
Otras	27
DESARROLLO DEL CONTENIDO DEL PROYECTO	28
Análisis	28
Casos de uso.....	28
Diagrama UML	35
Diseño de la base de datos.....	36
Diagrama Entidad-Relación	37
Diagrama Relacional.....	38
Base de datos en XAMPP	38
Funcionamiento de la aplicación	41
Estructura.....	41
Inicio de sesión.....	43
Dashboard.....	45
Correo electrónico	49
Pacientes.....	52
Citas	55
Facturación	57
Administrador	59
CONCLUSIONES Y MEJORAS	63
Conclusiones Generales.....	63
Evaluación del Trabajo y Cumplimiento de Objetivos	63
Limitaciones de la Aplicación.....	64

Aprendizaje Técnico y Profesional.....	64
Mejoras Futuras	65
BIBLIOGRAFÍA.....	66
ANEXOS	67
Anexo I – Script SQL de la base de datos	67
Anexo II – Código de validación del registro de pacientes	72
Anexo III – Mensajes informativos	73
III.1. Registrar nuevo paciente	73
III.2. Agendar cita.....	74
III.3. Generar factura	74
III.4. Editar paciente.....	74
III.5. Editar y eliminar cita	75
Anexo IV – Código de generar factura	75
Anexo V – Correo	79
V.1. Envío.....	79
V.2. Estilo de plantilla.....	80
Anexo VI – Código de generar pdf de informes médicos.....	81



INTRODUCCIÓN

En el entorno sanitario actual, la optimización de los recursos y la eficiencia administrativa resultan tan críticas como la propia calidad asistencial. Bajo esta premisa nace *MediGest Pro*, una solución de software integral diseñada específicamente para modernizar la gestión de clínicas y consultas privadas. Se trata de una aplicación de escritorio desarrollada sobre la tecnología WPF (*Windows Presentation Foundation*), concebida para actuar como el núcleo digital de la consulta, centralizando en una única plataforma operaciones que tradicionalmente se encontraban dispersas.

El origen de esta propuesta reside en la necesidad de solucionar la problemática habitual de la gestión manual o el uso de herramientas ofimáticas desconectadas. La dependencia de agendas en papel y hojas de cálculo independientes no solo ralentiza el flujo de trabajo diario, sino que aumenta exponencialmente el riesgo de errores humanos, como la duplicidad de citas, la pérdida de información de pacientes o la dificultad para obtener una visión clara de la disponibilidad médica. *MediGest Pro* responde a esta situación proporcionando un entorno automatizado, seguro y visualmente intuitivo que garantiza la integridad de los datos y agiliza la toma de decisiones.

En cuanto a su funcionalidad, el sistema abarca el ciclo administrativo y clínico completo para el uso exclusivo del personal interno. La aplicación permite desde la gestión de altas y modificaciones de pacientes y sus historiales médicos, hasta la coordinación de una agenda de citas interactiva con notificaciones automatizadas por correo electrónico. Asimismo, integra un módulo de facturación capaz de generar informes en PDF y un dashboard estratégico que, mediante gráficos estadísticos, ofrece una visión global del rendimiento de la clínica en tiempo real, todo ello respaldado por una base de datos MySQL en red local.

Para garantizar la solidez y coherencia del resultado final, el proceso de construcción del software se ha regido por una metodología en Cascada. Este enfoque secuencial ha permitido avanzar de manera ordenada y rigurosa a través de las fases de análisis de requisitos, diseño del sistema, implementación

del código y verificación, asegurando que cada etapa quedase perfectamente definida y validada antes de proceder con la siguiente.

Finalmente, la presente documentación se estructura para reflejar fielmente el recorrido técnico realizado, siguiendo el modelo de la metodología en Cascada mencionada anteriormente. El documento comienza con el Estudio del Mercado para contextualizar el dominio, seguido de la justificación detallada de las Tecnologías y Herramientas seleccionadas. Posteriormente, el cuerpo principal aborda el Diseño y Desarrollo de la Solución, cubriendo el Análisis de Requisitos y los diagramas UML/Entidad-Relación, para luego describir el funcionamiento e implementación de los módulos clave. La documentación concluye con el Despliegue y Pruebas de la aplicación, y finaliza con las Conclusiones, Limitaciones y Líneas Futuras del proyecto.



MediGest Pro

ESTUDIO DEL MERCADO

Panorama Estratégico y Conceptos Fundamentales

El mercado global de software para la gestión clínica está dominado por las soluciones basadas en la Nube (SaaS), con líderes como *Medesk* y *Clinic Cloud* estableciendo el estándar funcional. El desarrollo de *MediGest Pro* como un Sistema de Gestión Clínica (CMS) de escritorio (WPF) con una base de datos local (MySQL) es una elección estratégica que se contrapone a esta tendencia y requiere una justificación sólida.

- SaaS (Software as a Service)

Modelo dominante donde el proveedor asume la responsabilidad del mantenimiento, la seguridad, las copias de seguridad y la infraestructura de TI (Gasto Operativo - OPEX). Ofrece alta disponibilidad y escalabilidad elástica.

- Arquitectura Local (Desktop/WPF/MySQL)

Modelo propuesto. Ofrece control absoluto sobre el dato y rendimiento nativo superior en la interfaz (UX). Implica que el cliente final asume el 100% del riesgo de TI (seguridad, mantenimiento, recuperación ante desastres) y representa una Inversión de Capital - CAPEX.

- Segmento Objetivo

El sistema local está dirigido a clínicas o consultorios que priorizan el rendimiento nativo, exigen el almacenamiento físico de la información *in situ* y valoran la inversión CAPEX y el potencial de personalización ilimitada.

Benchmarking Funcional y Estándares de Mercado

La funcionalidad del sistema propuesto debe alcanzar la **paridad funcional** con los líderes del SaaS y buscar la **diferenciación operativa** a través del rendimiento y el control de datos, superando las siguientes expectativas del mercado.

Concepto Relevante	Requisito del Mercado (Benchmark)	Implicación para <i>MediGest Pro</i>
HCE y Documentación	Debe ser la fuente única de verdad y permitir el almacenamiento seguro de multimedia.	El core del sistema debe ser robusto, seguro y extremadamente rápido gracias a WPF.
Automatización de Comunicación	Crucial para la rentabilidad (reduce el ausentismo). Requiere recordatorios configurables por email y/o SMS.	Cumple con la comunicación inmediata, incluyendo confirmaciones y envío de mensajes; los recordatorios programados 24/7 se proyectan como mejoras futuras.
Business Intelligence (BI)	Va más allá de la contabilidad básica. Requiere ofrecer informes financieros detallados y métricas visuales (KPIs) para la toma de decisiones estratégicas.	Oportunidad para usar la potencia gráfica de WPF para crear un Dashboard nativo y avanzado, superando las limitaciones de rendimiento web.



Desafíos Críticos: Seguridad y Arquitectura

La elección de la arquitectura local impone desafíos de seguridad y mantenimiento que deben ser proactivamente mitigados por el propio software.

- **Seguridad y Cumplimiento Normativo (GDPR/Local)**

La arquitectura local otorga al profesional la ventaja estratégica del control total y la soberanía sobre sus datos. *MediGest Pro* está diseñado para convertir esta autonomía en la máxima seguridad, ya que no depende de terceros, implementando un módulo de seguridad asistida que garantiza la protección de la información crítica mediante:

- Protección de la Confidencialidad: Cifrado de datos en reposo obligatorio en la base de datos MySQL, además de hashing unidireccional para las contraseñas, protegiendo la información sensible desde su almacenamiento.
- Acceso Controlado: Control de acceso riguroso basado en roles (como médico, administrador) para garantizar el cumplimiento del principio de mínimo privilegio y la trazabilidad de la información.

- **Rendimiento y Despliegue**

WPF ofrece una interfaz de usuario avanzada y una respuesta instantánea, que es el principal diferenciador de la **Experiencia de Usuario (UX)** frente a la latencia web. El desafío técnico reside en la logística de actualización del software en las máquinas locales (despliegue centralizado y silencioso).

- **MySQL como Backend**

La selección de **MySQL** como el sistema de gestión de bases de datos (DBMS) se justifica por su **robustez, fiabilidad y amplio soporte comunitario**. Para el desarrollo y despliegue del entorno local se utiliza el paquete **XAMPP**, que facilita la configuración del servicio MySQL en la infraestructura de la clínica.

Al ser una plataforma de código abierto y local, ofrece una ventaja estratégica crucial para el modelo de licencia perpetua (**CAPEX**):

- **Autonomía y Propiedad del Dato:** La instalación local de MySQL otorga al cliente **control total y directo sobre sus datos**, un factor decisivo para clínicas con estrictas políticas de privacidad.
- **Integración y Auditoría:** La familiaridad de MySQL permite la **integración nativa con herramientas de gestión de bases de datos profesionales**, apelando a clientes que desean auditar o personalizar su propio *schema* de datos.
- **Seguridad de Acceso:** La seguridad de los datos de acceso está garantizada mediante el uso de técnicas de **hashing unidireccional** para el almacenamiento de contraseñas, asegurando que las credenciales de los usuarios nunca se almacenen en texto plano.

El desafío operativo de esta elección es el riesgo de inactividad, ya que la continuidad del servicio depende del hardware del servidor local. Para mitigarlo, es imperativo que las conexiones entre el cliente WPF y el servidor MySQL estén obligatoriamente cifradas, garantizando la protección de los datos en tránsito.

Conclusión Estratégica

La viabilidad de *MediGest Pro* radica en ofrecer a las clínicas una herramienta de **alto rendimiento nativo** con **control total sobre sus datos** (Modelo CAPEX).

El éxito de la solución se garantiza mediante dos estrategias clave:

1. **Seguridad Asistida:** Integrar la **automatización de tareas de seguridad** para simplificar el manejo de riesgos para el usuario final.
2. **Hoja de Ruta Clara:** Reconocer que las funcionalidades de **movilidad** (acceso desde el móvil) y **comunicación programada** (recordatorios automáticos 24/7) son las principales **Líneas Futuras de Desarrollo (LFD)** necesarias para igualar completamente la oferta de la competencia en el mercado.



TECNOLOGÍAS Y HERRAMIENTAS UTILIZADAS

Para el desarrollo del proyecto se ha hecho uso de varias tecnologías, herramientas y servicios que han permitido cubrir tanto la parte de programación como la gestión de la base de datos y el diseño del sistema.

FrameWork

- Windows Presentation Foundation

Hacemos uso de WPF para el diseño de la interfaz gráfica ya que es un framework que permite crear aplicaciones de escritorio modernas mediante XAML y facilita la separación entre la interfaz y la parte lógica. Utiliza un motor de representación basado en vectores permitiendo así el desarrollo de interfaces escalabres e independientes de la resolución.

WPF proporciona un amplio conjunto de funcionalidades como controles visuales, enlace de datos, gráficos, estilos, plantillas... Al formar parte del entorno .NET, permite integrar de forma sencilla los componentes de la API.

Actualmente existen dos implementaciones de WPF:

- WPF en .NET, versión de código abierto que hace requiere Visual Studio 2019 y superiores.
- WPF en .NET Framework 4, incluida junto al framework en sistemas Windows.

Lenguaje de programación

- C#

Es un lenguaje de programación moderno y orientado a objetos desarrollado por Microsoft como parte de la plataforma .NET. Se caracteriza por su sintaxis clara, seguridad y facilidad para crear aplicaciones de escritorio, web o servicios.

En este proyecto se ha utilizado C# porque es el lenguaje principal soportado por Visual Studio del que hablaremos más adelante. Se integra con WPF, lo que

facilita el desarrollo de la lógica, la comunicación con la base de datos y la gestión de eventos.



Base de datos

- XAMPP

Para el desarrollo del proyecto, la base de datos se ha implementado utilizando XAMPP, un paquete que incluye Apache, MariaDB/MySQL, PHP y Perl. Esta elección se debe a que XAMPP ofrece un entorno local sencillo y rápido de configurar para realizar pruebas durante el desarrollo.

La facilidad en la instalación, configuración y que incluye todos los componentes necesarios permite tener en pocos minutos un entorno completo de servidor de forma gratuita y al ser local permite al usuario desarrollar y probar sin conexión a internet. La integración con phpMyAdmin facilita la gestión visual de la base de datos, es ligero y multiplataforma estando disponible para varios sistemas operativos.



- Apache

Es un servidor web incluido en WAMPP que permite ejecutar servicios y aplicaciones en un entorno local. En este proyecto se utiliza principalmente para habilitar phpMyAdmin. Su integración facilita su uso sin necesidad de realizar configuraciones avanzadas y permite trabajar en un entorno de desarrollo estable y accesible desde el navegador.





MediGest Pro

- MySQL

Es un sistema de gestión de base de datos relacional muy usado por su rendimiento, facilidad y estabilidad. En este proyecto se ha empleado para almacenar, gestionar toda la información y crear la base de datos. Su integración dentro de XAMPP ha permitido administrar la base de datos de forma sencilla durante el desarrollo.



Librerías

- Itext

Librería utilizada para manipular y crear documentos PDF de forma programática. Hemos hecho uso de ella para la creación de informes personalizados con los datos de los pacientes y la información relevante del sistema. Gracias a ella se facilita la creación de documentos sin necesidad de otras herramientas externas.



IDE

- Visual Studio

Entorno de desarrollo integrado (IDE) utilizado para desarrollar y mantener la aplicación. Ofrece múltiples herramientas avanzadas para la edición de código, depuración y diseño de interfaces con C# y WPF. Su integración con .NET y capacidad de compilación, ejecución y prueba de la aplicación desde un mismo

entorno hace que sea una herramienta completa y eficiente para el desarrollo del proyecto



Herramientas de diseño

- Lucidchart

Es una herramienta para crear diagramas online de forma gratuita ofreciendo múltiples plantillas de diagramas facilitando así su elaboración. Requiere de una suscripción sin coste para su uso y poder guardar los diagramas. En este proyecto se ha utilizado para el diseño de modelo Entidad-Relación por su facilidad de uso y capacidad de exportación.



- Ddiagram

Esta plataforma permite diseñar modelos relacionales de forma gratuita a través de un script donde están definidas la creación de las tablas y las relaciones entre ellas. Permite importar diferentes tipos de bases de datos al igual que exportar el diagrama y el código generado.



- Miro

Es una pizarra digital colaborativa online que permite crear un espacio de trabajo visual e infinito donde un grupo o equipo puede planificar, organizar ideas, hacer diagramas, bocetos, mapas mentales, presentar flujos de trabajo y colaborar en tiempo real. Gracias a su interfaz sencilla y a múltiples herramientas se convierte en un entorno ideal para el diseño, planificación de proyectos,



MediGest Pro

brainstorming, gestión ágil y documentación visual. Hemos hecho uso de esta herramienta para la creación del diagrama UML de manera encilla, intuitiva y gratuita.



Control de versiones

- Git Hub

Esta plataforma online de control de versiones basada en Git permite almacenar, gestionar y colaborar en proyectos de software. Facilita el seguimiento de cambios, la organización del código y el trabajo en equipo mediante los repositorios. Su uso ha permitido mantener el código actualizado, realizar copias de seguridad, desarrollar en paralelo y a la vez y llevar un control ordenado de las distintas versiones del proyecto.



Otras

- Gmail

Este servicio de correo de Google nos ha permitido enviar mensajes de forma rápida y segura. Su uso nos ha permitido desarrollar la comunicación directa entre los perfiles profesionales y los pacientes. Gracias a su fiabilidad y su facilidad de integración lo convierten en una herramienta útil para la funcionalidad de notificaciones del sistema.



DESARROLLO DEL CONTENIDO DEL PROYECTO

Análisis

Casos de uso

ID	1
Caso de uso	Iniciar sesión
Actores	Médico, Recepcionista
Propósito	Entrar en la aplicación
Descripción	<ol style="list-style-type: none"> El sistema muestra el formulario de inicio de sesión. El usuario introduce el usuario y la contraseña en los campos. El usuario pulsa el botón de "Iniciar Sesión". El sistema validará si los datos son correctos.
Precondiciones	El usuario debe estar registrado en la base de datos con un rol asignado (médico, recepcionista o admin)
Postcondiciones	Se establece la sesión del usuario, almacenando su ID, nombre, apellidos y rol
Extensiones síncronas	Si el usuario o la contraseña no son válidos, el sistema mostrará el mensaje "Usuario o contraseña incorrectos". Si los campos están vacíos, muestra "Por favor, introduce usuario y contraseña"

ID	2
Caso de uso	Registrar nuevo paciente
Actores	Recepcionista
Propósito	Añadir un nuevo paciente a la base de datos
Descripción	<ol style="list-style-type: none"> Seleccionar el botón "Registrar nuevo paciente" desde el <i>Dashboard</i>. Rellenar los campos con la información del paciente: nombre, apellido, DNI, CIPA, número de historia clínica y de seguridad social y fecha de nacimiento. El sistema asigna automáticamente la fecha de ingreso actual y un correo por defecto. Pulsar el botón "Guardar"
Precondiciones	El usuario con el que has iniciado sesión debe ser un recepcionista.
Postcondiciones	El paciente queda registrado en la base de datos con la fecha de ingreso actual.
Extensiones síncronas	Si el DNI no cumple el algoritmo de validación español (8 números + letra correcta), el sistema muestra "DNI Invalido". Si la fecha de nacimiento es posterior a la fecha actual, muestra "No puedes



	registrar pacientes que todavía no hayan nacido". Si ocurre un error al guardar, se muestra el mensaje de error detallado.
--	--

ID	3
Caso de uso	Agendar cita
Actores	Recepcionista
Propósito	Añadir una nueva cita a la base de datos
Descripción	<ol style="list-style-type: none">1. Seleccionar el botón "Agendar cita" desde el <i>Dashboard</i>.2. Seleccionar la fecha en el calendario (solo permite fechas desde hoy en adelante).3. Seleccionar el paciente del desplegable.4. Seleccionar el médico del desplegable.5. Seleccionar la hora (franjas de 09:00 a 20:30 en intervalos de 30 minutos).6. Seleccionar el estado de la cita.7. Opcionalmente, añadir observaciones.8. Pulsar "Agendar".9. El sistema envía automáticamente un correo al paciente con los detalles de la cita.
Precondiciones	El usuario con el que has iniciado sesión debe ser un recepcionista. Deben existir pacientes y médicos registrados en el sistema.
Postcondiciones	La cita queda registrada en la base de datos asociada al recepcionista que la creó. Se envía un correo electrónico al paciente con la confirmación.
Extensiones síncronas	Si no se selecciona fecha, médico u hora, el sistema muestra mensajes de aviso. Si ya existe una cita para el mismo médico en la misma fecha y hora con el mismo estado, muestra "Ya existe una cita para este médico en esa fecha y hora". Si falla el envío del correo, la cita se guarda igualmente y se notifica el error.

ID	4
Caso de uso	Generar factura
Actores	Recepcionista
Propósito	Generar una nueva factura en formato PDF
Descripción	<ol style="list-style-type: none">1. Seleccionar el botón "Generar factura" desde el <i>Dashboard</i>.2. Seleccionar el paciente del desplegable.3. Seleccionar la fecha de inicio y fecha de fin del período a facturar.4. Pulsar "Generar Factura".

	<p>5. El sistema genera un PDF con el detalle de las citas realizadas, incluyendo fecha, médico, especialidad, precio y observaciones.</p> <p>6. El PDF se guarda automáticamente en la carpeta "Facturaciones" del directorio de la aplicación.</p> <p>7. El PDF se abre automáticamente en el visor predeterminado del sistema.</p>
Precondiciones	Deben existir citas con estado "realizada" en el rango de fechas seleccionado.
Postcondiciones	Se genera un archivo PDF con nombre "Factura_YYYYMMDD_HHMM.pdf" en la carpeta Facturaciones.
Extensiones síncronas	Si no se selecciona un paciente, muestra "Selecciona un paciente". Si no se seleccionan las fechas, muestra "Selecciona las fechas de inicio y fin". Si la fecha de inicio es posterior a la de fin, muestra error. Si no hay citas realizadas en el rango de fechas, muestra "No hay citas realizadas con el paciente en ese rango de fechas"

ID	5
Caso de uso	Borrar cita
Actores	Recepcionista
Propósito	Eliminar una cita de la base de datos
Descripción	<p>1. Navegar a la ventana de "Citas" en el menú lateral izquierdo.</p> <p>2. Hacer click derecho sobre la cita que se desea eliminar.</p> <p>3. Confirmar el borrado en la ventana emergente pulsando "Sí".</p>
Precondiciones	La cita debe de estar en estado confirmada o pendiente, nunca realizada.
Postcondiciones	La cita es eliminada permanentemente de la base de datos. La tabla se refresca automáticamente.
Extensiones síncronas	Si el usuario es un médico, el sistema muestra "No tienes permisos para borrar Citas". Si el usuario pulsa "No" en la confirmación, la operación se cancela.

ID	6
Caso de uso	Ver informe médico
Actores	Médico
Propósito	Consultar los informes médicos existentes de un paciente



Descripción	<ol style="list-style-type: none"> 1. Navegar a la ventana de "Pacientes" en el menú lateral izquierdo. 2. Hacer click derecho sobre el paciente del que se quiere ver el informe. 3. Seleccionar "Ver Informes Médicos" en el menú contextual. 4. Se abre una nueva ventana mostrando todos los informes del paciente creados por el médico actual.
Precondiciones	El médico solo puede ver los pacientes con los que tiene informes asociados.
Postcondiciones	El médico ve una pestaña nueva con todos los informes generados para dicho paciente
Extensiones síncronas	Si el usuario es un recepcionista, el sistema muestra "Solo los Médicos pueden revisar los informes médicos asociados a ellos mismos de los Pacientes"

ID	7
Caso de uso	Ver citas
Actores	Médico, Recepcionista
Propósito	Consultar la lista de citas agendadas
Descripción	<ol style="list-style-type: none"> 1. Navegar a la ventana de "Citas" en el menú lateral izquierdo. 2. El sistema muestra la lista de citas en una tabla. 3. Opcionalmente, filtrar por nombre de paciente, fecha de cita o estado.
Precondiciones	Para médicos: solo se muestran las citas donde él es el médico asignado. Para recepcionistas: solo se muestran las citas que él mismo ha creado
Postcondiciones	El sistema muestra la lista de citas filtradas según los criterios seleccionados.
Extensiones síncronas	Los filtros son acumulativos y pueden combinarse.

ID	8
Caso de uso	Ver pacientes
Actores	Médico, Recepcionista
Propósito	Consultar la lista de pacientes registrados
Descripción	<ol style="list-style-type: none"> 1. Navegar a la ventana de "Pacientes" en el menú lateral izquierdo. 2. El sistema muestra la lista de pacientes. 3. Opcionalmente, filtrar por nombre, año y/o mes de nacimiento, fecha exacta de nacimiento.

Precondiciones	Para médicos: solo se muestran los pacientes con los que tienen al menos un informe médico asociado (sin duplicados). Para recepcionistas: se muestran todos los pacientes registrados.
Postcondiciones	El sistema muestra el listado de pacientes filtrado según los criterios seleccionados.
Extensiones síncronas	El filtro de fecha exacta y los filtros de año/mes son mutuamente excluyentes.

ID	9
Caso de uso	Modificar estado de una cita
Actores	Médico, Recepcionista
Propósito	Actualizar el estado de una cita (Pendiente, Realizada, Cancelada, Confirmada)
Descripción	<ol style="list-style-type: none"> Navegar a la ventana de "Citas" en el lateral izquierdo. Doble click en la cita a modificar. Cambiar la información de la cita en los campos de la ventana emergente. Pulsar "Cancelar" o "Guardar" cambios y confirmar.
Precondiciones	El usuario debe tener permisos para modificar citas
Postcondiciones	El estado de la cita es actualizado en la base de datos
Extensiones síncronas	<p>Si un médico intenta cambiar el estado de la cita a cualquier opción diferente de "Realizada", el sistema mostrará un mensaje de error.</p> <p>Si un Recepcionista intenta guardar la cita sin seleccionar una fecha u hora válida el sistema mostrará un mensaje de error.</p> <p>Si se detecta una cita existente con las mismas características, muestra un aviso de duplicado y no permite guardar la cita.</p>

ID	10
Caso de uso	Enviar correo a paciente
Actores	Médico, Recepcionista
Propósito	Comunicarse con el paciente por correo electrónico
Descripción	<ol style="list-style-type: none"> Navegar a la ventana de "Pacientes" en el lateral izquierdo. Click derecho en el paciente seleccionado. Seleccionar la opción "Enviar correo".



	4. Redactar el contenido y pulsar "Cancelar" para descartar o "Aceptar" para enviar el correo.
Precondiciones	El paciente debe tener una dirección de correo electrónico válida registrada.
Postcondiciones	El correo es enviado al paciente utilizando una plantilla HTML con el logo de la aplicación.
Extensiones síncronas	Si la dirección de correo no es válida o falta, el sistema muestra un mensaje de error

ID	11
Caso de uso	Crear usuario
Actores	Administrador
Propósito	Añadir un nuevo usuario (Médico, Recepcionista o Administrador) al sistema
Descripción	<ol style="list-style-type: none">Desde el panel de administración, seleccionar el botón "Crear usuario".Rellenar los campos: login, contraseña, confirmación de contraseña y rol.Pulsa el botón de "Guardar".Si el rol es "médico", se abre automáticamente el formulario para crear los datos del médico (nombre, apellidos, especialidad, número colegiado, correo corporativo).Si el rol es "recepcionista", se abre automáticamente el formulario para crear los datos del recepcionista.
Precondiciones	El usuario con el que has iniciado sesión debe ser un Administrador.
Postcondiciones	El nuevo usuario es registrado en la base de datos. Si es médico o recepcionista, también se crea el registro correspondiente en su tabla.
Extensiones síncronas	Si el nombre de usuario ya existe o los datos son inválidos, el sistema mostrará un mensaje de error.

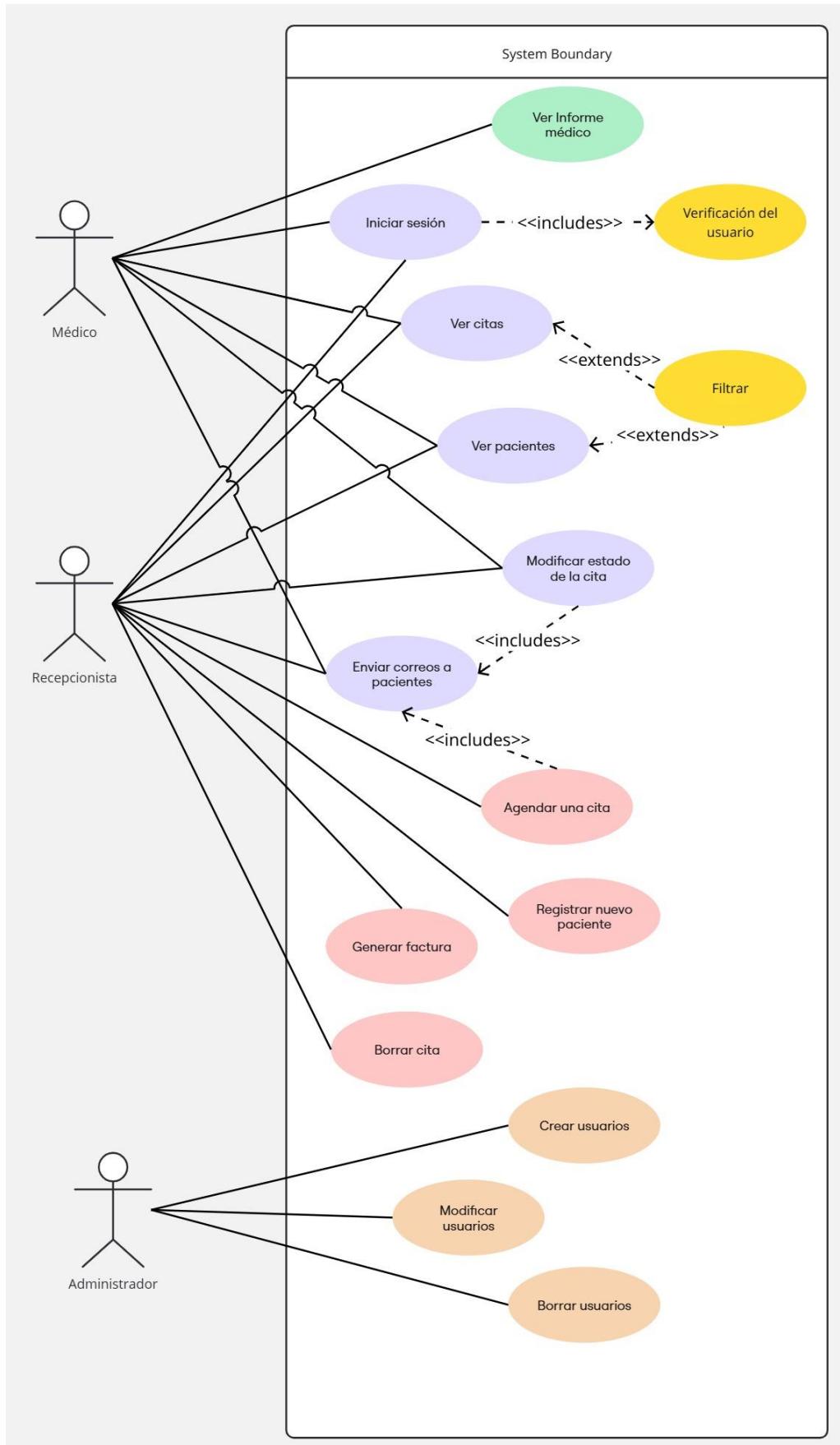
ID	12
Caso de uso	Modificar usuario
Actores	Administrador
Propósito	Actualizar los datos de un usuario existente
Descripción	<ol style="list-style-type: none">Desde el panel de administración, navegar a la ventana "Usuarios".Hacer doble click en el usuario que desea modificar.Se abre una ventana de edición con el login actual.

	<p>4. Modificar el campo de login 5. Pulsar "Guardar" para confirmar o "Cancelar" para descartar.</p> <p>Para cambiar contraseña</p> <ol style="list-style-type: none"> 1. Hacer click derecho en el usuario. 2. Seleccionar "Cambiar Contraseña". 3. Introducir la nueva contraseña en el formulario.
Precondiciones	El usuario con el que has iniciado sesión debe ser un Administrador.
Postcondiciones	Los datos del usuario son actualizados en la base de datos.
Extensiones síncronas	Si no se encuentra el usuario en la base de datos, muestra "No se encontró el usuario en la base de datos".

ID	13
Caso de uso	Borrar usuario
Actores	Administrador
Propósito	Eliminar un usuario del sistema
Descripción	<ol style="list-style-type: none"> 1. Desde el panel de administración, navegar a la ventana "Usuarios". 2. Hacer click derecho en el usuario que se desea eliminar. 3. Seleccionar "Borrar Usuario" en el menú contextual. 4. Confirmar el borrado en la primera ventana emergente. 5. Confirmar nuevamente en la segunda ventana de confirmación. 6. Si el usuario es un médico, se abre la ventana "ReemplazarMedico" para reasignar sus citas/informes a otro médico. 7. Si el usuario es un recepcionista, se abre la ventana "ReemplazarRecepcionista" para reasignar sus citas a otro recepcionista.
Precondiciones	El usuario con el que has iniciado sesión debe ser un Administrador.
Postcondiciones	El usuario es eliminado de la base de datos. Si era médico o recepcionista, se muestra ventana de reemplazo para gestionar dependencias.
Extensiones síncronas	Si se intenta borrar el propio usuario activo, el sistema muestra "No te puedes borrar a ti mismo" e impide la operación. Se requiere doble confirmación antes de proceder con el borrado.



Diagrama UML



Diseño de la base de datos

Para la creación de la base de datos primero debíamos identificar de forma clara cuáles eran nuestras entidades principales y los elementos más relevantes y necesarios de cada una para posteriormente establecer las relaciones entre ellas y la cardinalidad.

Partimos de una relación jerárquica conformada por la entidad Usuario actuando como supertipo y recogiendo en ella los atributos compartidos por las subentidades.

- id = identificador único y clave primaria
- login = nombre de usuario necesario la autentificación en el inicio de sesión
- password = contraseña
- rol = función que desempeñan (administrador, médico o recepcionista)

A partir de la entidad padre se definen las subentidades Recepcionista y Médico añadiendo información específica de su perfil profesional. Cada usuario debe pertenecer de forma obligatoria a una única subentidad definida por el rol que cumplen siendo así una jerarquía exclusiva total.

El usuario con el rol de administrador se encarga de realizar las tareas de crear, modificar y eliminar usuarios. No tiene ningún atributo propio que lo diferencie de los demás usuarios más los que le otorga el ser un usuario de nuestra base de datos.

La entidad Recepcionista representa a los trabajadores que realizan la gestión diaria de citas pudiendo gestionar varias y cada una debe ir asociada a un único recepcionista. Es importante saber el nombre, apellidos, teléfono y email.

Unos datos cruciales a la hora de guardar las citas son la fecha y la hora a la que tendrán lugar, el estado en el que se encuentra (pendiente, confirmada, realizada, cancelada) y observaciones importantes que se deban anotar. A su



vez las citas van asociadas a un único médico, pero este atiende varias consultas.

De los médicos se debe conocer el nombre, apellidos, número de colegiado y el correo corporativo. Cada médico tiene una especialidad de la que se sabe el nombre, una pequeña descripción de cada una y el coste que supone ser de esa especialidad que más a delante se verá reflejado en las facturas. Cada médico debe redactar los informes médicos donde debe aparecer el motivo de consulta, el diagnóstico, tratamiento, la fecha en la que se crea y si es necesario añadir nuevas observaciones.

Para poder hacer la gestión de los pacientes es necesario saber el nombre, apellidos, DNI, fecha de nacimiento, número de historia clínica, el CIPA, número de la seguridad social y el correo personal de cada uno. El paciente puede acudir a varias citas médicas y tener varios informes médicos.

Una vez tenemos claro cómo queremos estructurar nuestra base de datos de forma interna, desarrollamos el Modelo Entidad-Relación que se mostrará a continuación con todos los atributos, relaciones y cardinalidades.

Diagrama Entidad-Relación

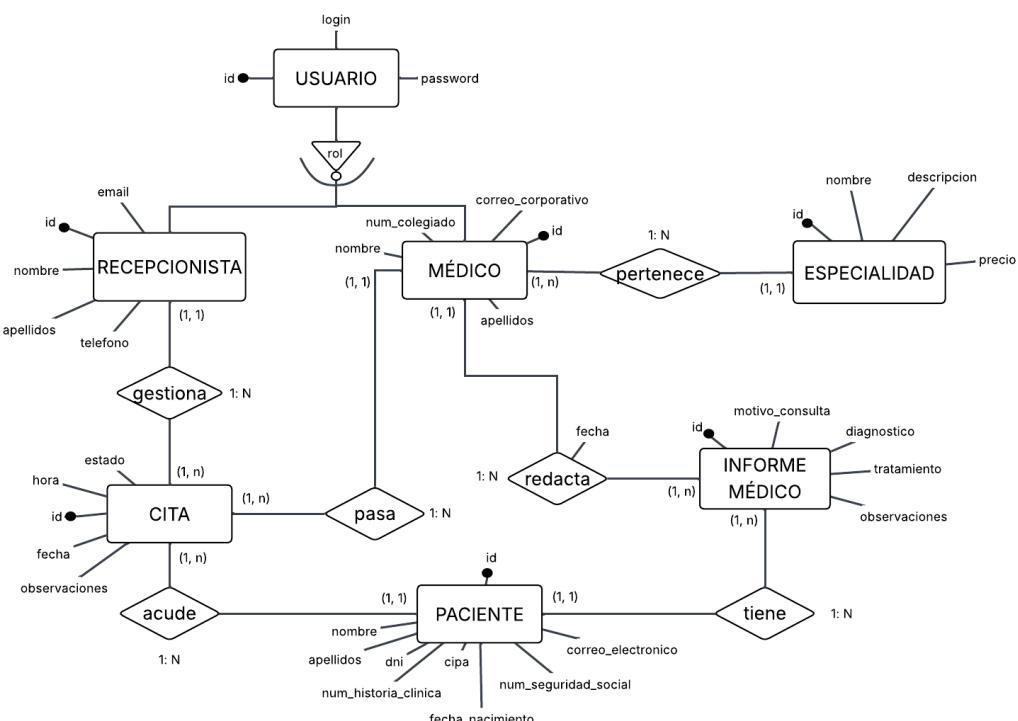
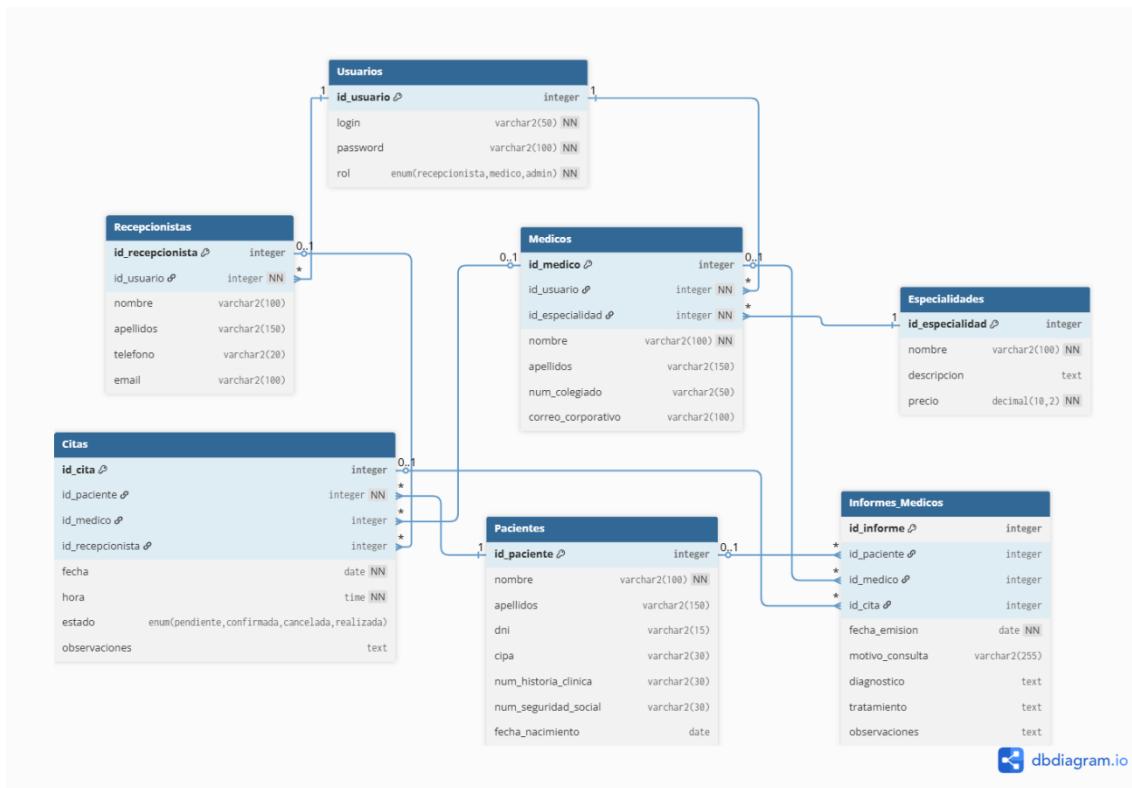


Diagrama Relacional

A través del diagrama anterior realizamos el paso a tablas que serán finalmente las que formarán nuestra base de datos. Gracias al diagrama relacional podemos ver rápidamente el número de tablas, los atributos, las claves primarias y las foráneas de cada una y de que tabla proviene cada una.



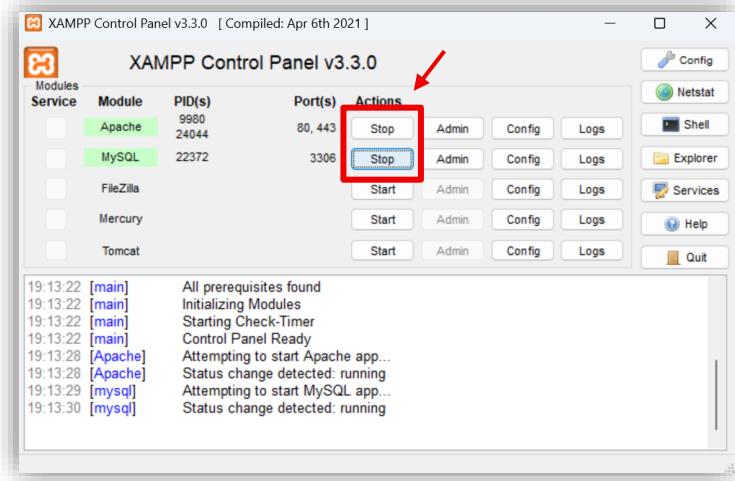
Base de datos en XAMPP

El proceso de implantación es:

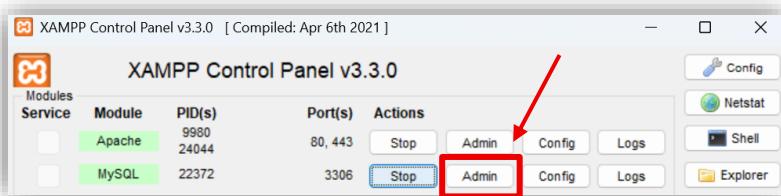
1. Instalación de XAMPP en el equipo.
2. Inicio del módulo de apache y del módulo de MySQL desde el panel de control de XAMPP.



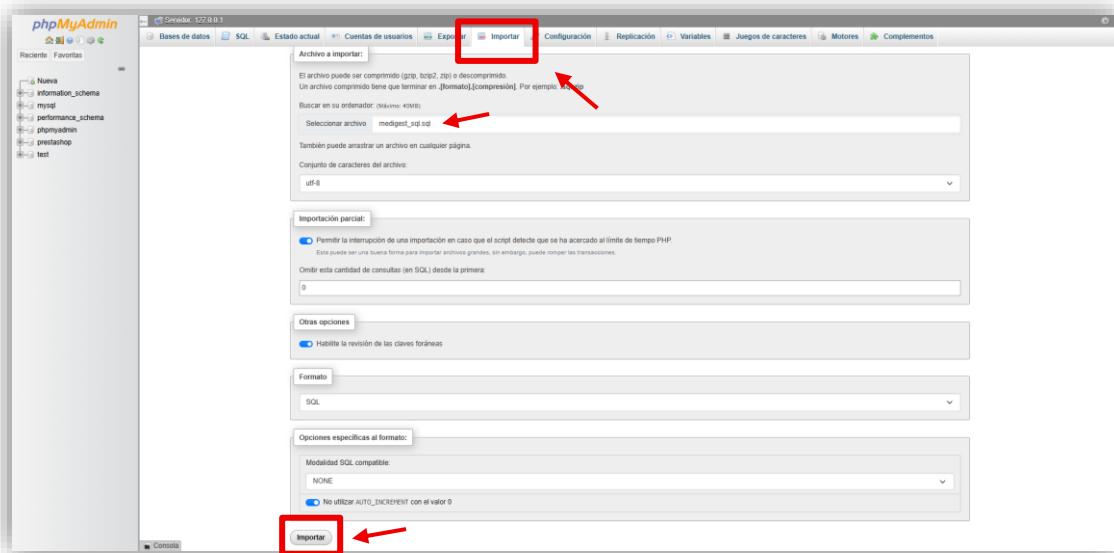
MediGest Pro



3. Acceder a phpMyAdmin a través de <http://localhost/phpmyadmin> o a través del panel de control en el módulo de MySQL en el botón de “Admin”



4. Creación de la base de datos del proyecto bien importando o escribiendo el script.



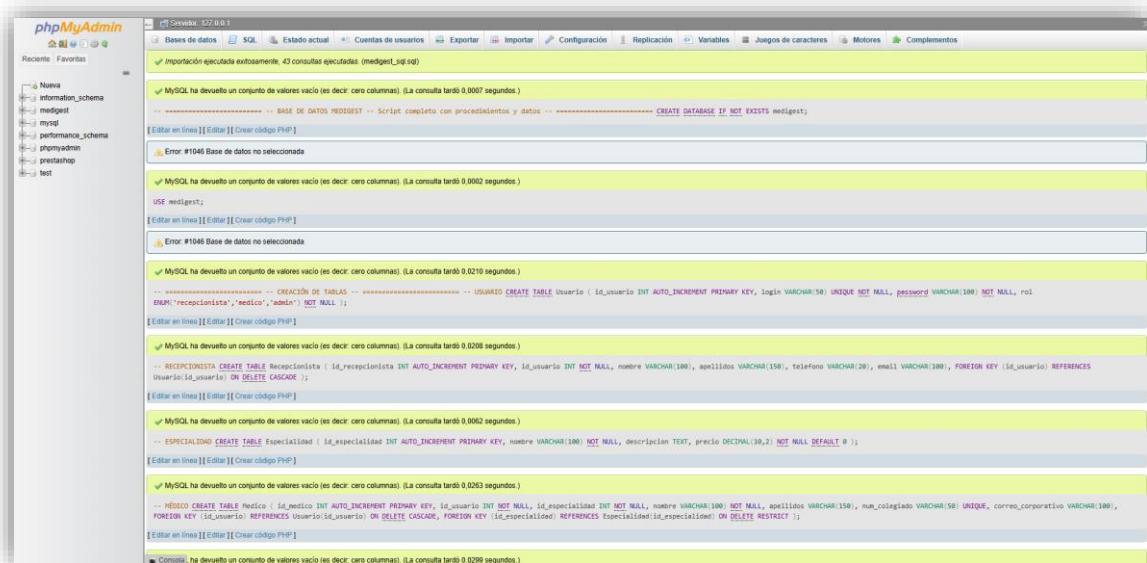
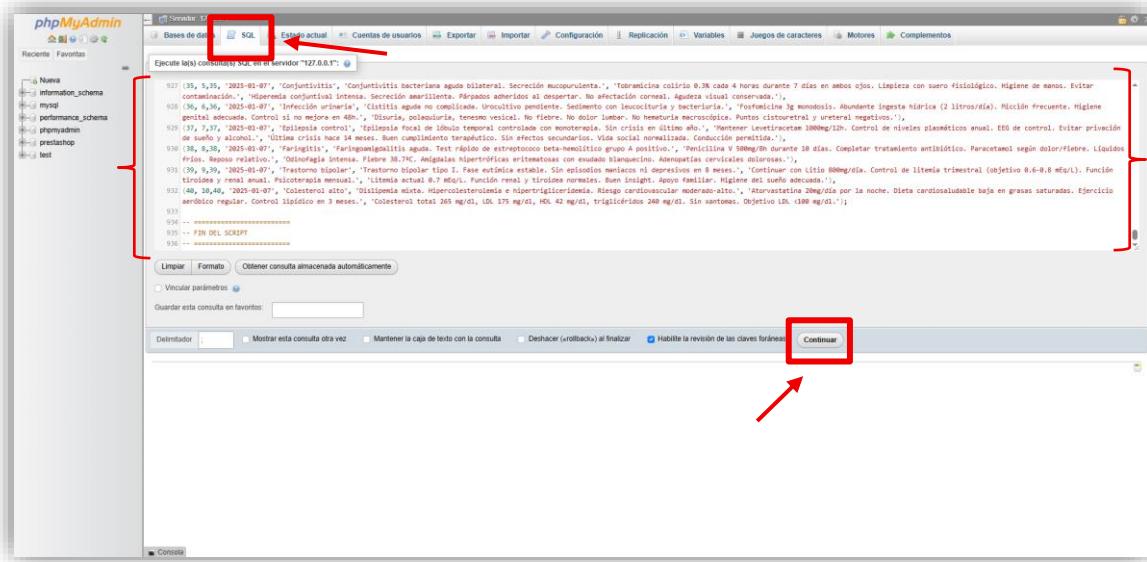




Tabla	Acción	Filas	Tipo	Cotejamiento	Tamaño	Résiduo a depurar
cita	Examinar Estructura Buscar Insertar Vaciar Eliminar	480	InnoDB utf8mb4_general_ci	64.0 KB	-	
especialidad	Examinar Estructura Buscar Insertar Vaciar Eliminar	10	InnoDB utf8mb4_general_ci	16.0 KB	-	
informe_medico	Examinar Estructura Buscar Insertar Vaciar Eliminar	40	InnoDB utf8mb4_general_ci	64.0 KB	-	
medico	Examinar Estructura Buscar Insertar Vaciar Eliminar	10	InnoDB utf8mb4_general_ci	64.0 KB	-	
paciente	Examinar Estructura Buscar Insertar Vaciar Eliminar	200	InnoDB utf8mb4_general_ci	80.0 KB	-	
receptionista	Examinar Estructura Buscar Insertar Vaciar Eliminar	4	InnoDB utf8mb4_general_ci	32.0 KB	-	
usuario	Examinar Estructura Buscar Insertar Vaciar Eliminar	14	InnoDB utf8mb4_general_ci	32.0 KB	-	
7 tablas	Número de filas	758	InnoDB utf8mb4_general_ci	352.0 KB	0 B	

El script completo de creación de la base de datos utilizada en este proyecto puede encontrarse en el [Anexo I - Script SQL de la base de datos](#).

Funcionamiento de la aplicación

Estructura

A continuación, se verá la estructura interna del proyecto, como está organizado y los principales archivos creados para su correcto funcionamiento.

- **Clases:** en esta carpeta se encuentran todas las clases que vimos en el modelo relacional con sus atributos y sus métodos propios.
- **Data:** La clase *MediGestContext* actúa como el contexto de *Entity Framework Core* y define todas las tablas del sistema mediante propiedades *DbSet*. También configura la conexión a la base de datos MySQL, indicando la cadena de conexión y la versión del servidor. Gracias a este contexto, el programa puede consultar, insertar, modificar y eliminar datos de la base de datos utilizando las clases del modelo en lugar de escribir consultas SQL manuales.

```

    <using> System;
    <using> System.Collections.Generic;
    <using> System.Linq;
    <using> System.Text;
    <using> System.Threading.Tasks;
    <using> MediGest.Clases;
    <using> Microsoft.EntityFrameworkCore;
    <using> Pomelo.EntityFrameworkCore.MySql;

    <namespace MediGest.Data
    {
        <public class MediGestContext : DbContext
        {
            <public DbSet<Paciente> Paciente { get; set; }
            <public DbSet<Medico> Medico { get; set; }
            <public DbSet<Repcionista> Repcionista { get; set; }
            <public DbSet<Especialidad> Especialidad { get; set; }
            <public DbSet<Cita> Cita { get; set; }
            <public DbSet<Informe_Medico> Informe_Medico { get; set; }
            <public DbSet<Usuario> Usuario { get; set; }

            <protected override void OnConfiguring(DbContextOptionsBuilder options)
            {
                options.UseMySQL(
                    "server=127.0.0.1;port=3306;database=medigest;user=root;",
                    new MySqlServerVersion(new Version(10, 4, 32))
                );
            }
        }
    }
}

```

- **Facturaciones e informes médicos:** en ellas se encuentran las facturas y los informes generados en ejecuciones anteriores.
- **Pages:** están alojadas todas las interfaces desarrolladas en XAML y su funcionalidad en su correspondiente archivo *.xaml.cs*. Una de las razones por las que se diseñó esta estructura es para no definir en cada una de las interfaces el menú de opciones, siendo este fijo mientras que el llamado *MainFrame.Navigate* es el que cambia según la selección del menú.



MediGest Pro

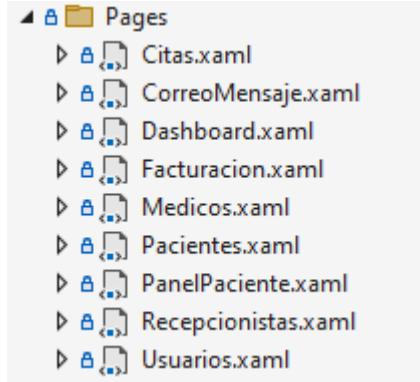
```
public MainWindow()
{
    InitializeComponent();
    //Por Defecto Inicializamos la Page de Dashboard
    MainFrame.Navigate(new Dashboard());
    SetActiveButton(BtnDashboard);
}

//Eventos para las Pages
private void BtnMostrarDashboard_Click(object sender, RoutedEventArgs e)
{
    MainFrame.Navigate(new Dashboard());
    SetActiveButton(BtnDashboard);
}

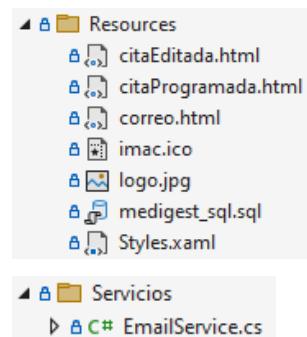
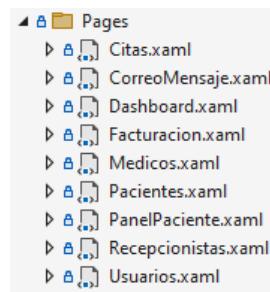
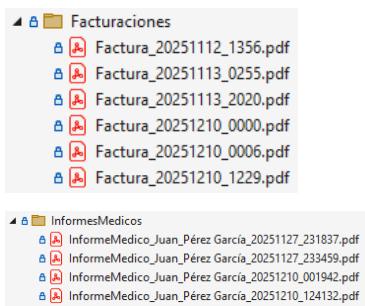
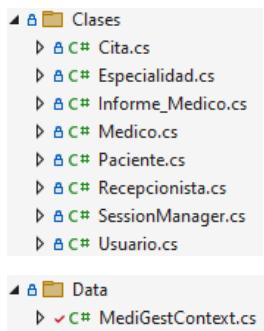
private void BtnMostrarPacientes_Click(object sender, RoutedEventArgs e)
{
    MainFrame.Navigate(new Pacientes());
    SetActiveButton(BtnPacientes);
}

private void BtnMostrarCitas_Click(object sender, RoutedEventArgs e)
{
    MainFrame.Navigate(new Citas());
    SetActiveButton(BtnCitas);
}

private void BtnMostrarFacturacion_Click(object sender, RoutedEventArgs e)
{
    MainFrame.Navigate(new Facturacion());
    SetActiveButton(BtnFacturacion);
}
```



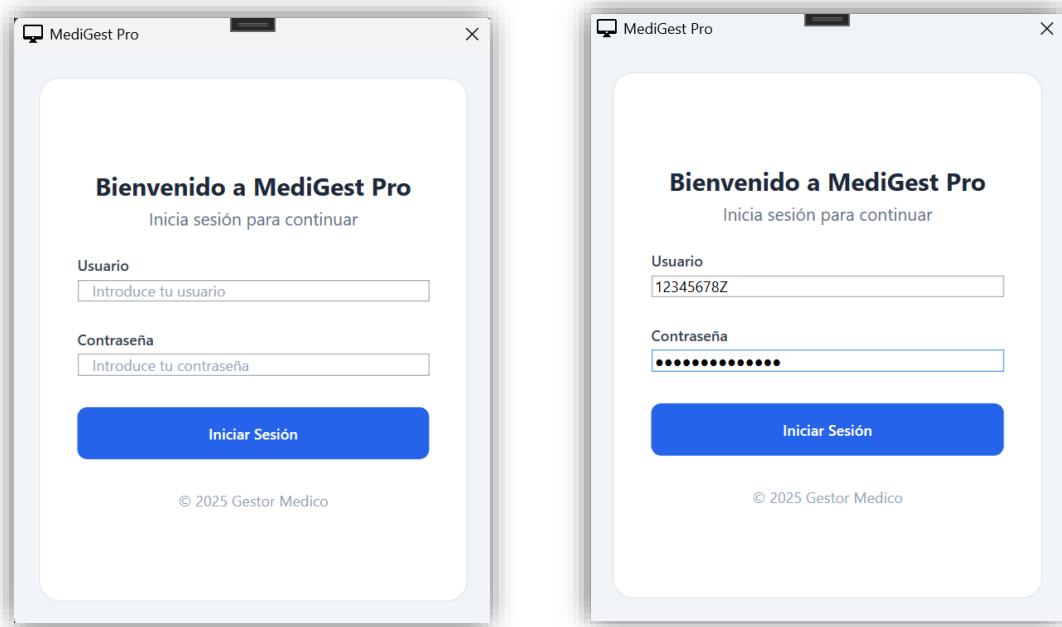
- Resources:** en esta carpeta están las plantillas creadas para el envío de correos, la base de datos, el logo, los estilos...
- Servicios:** es la ubicación del servicio de email que hace la conexión, el envío del correo y la gestión de esta funcionalidad.



Inicio de sesión

Como ya hemos visto, tenemos dos tipos de usuarios y esto será crucial para el desarrollo de las funcionalidades que podrá desempeñar cada uno, pudiendo tener algunas en común pero no todas.

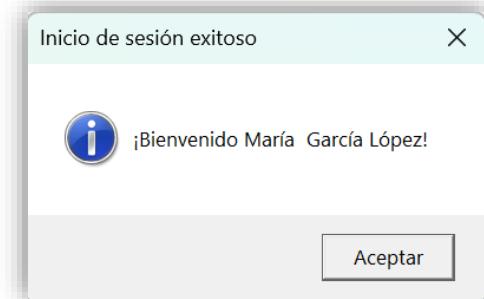
A continuación, iremos navegando por las distintas interfaces de la aplicación y así poder ver el diseño, el desarrollo y diferenciar mejor las acciones permitidas por cada usuario.



El usuario y contraseña que se ve en esta imagen es un usuario con el rol de recepcionista. Es importante activar los módulos correspondientes de XAMPP para así dar acceso a la base de datos y poder hacer la primera validación para acceder a las demás funcionalidades del proyecto. Una vez verificada la identidad del usuario accederemos a si nombre para darle la bienvenida de una manera más personalizada para acto seguido cargar la siguiente interfaz.

Nuestras contraseñas en la base de datos están encriptadas con una función Hash por lo que debemos convertirlas y así poder comparar con la base de datos.

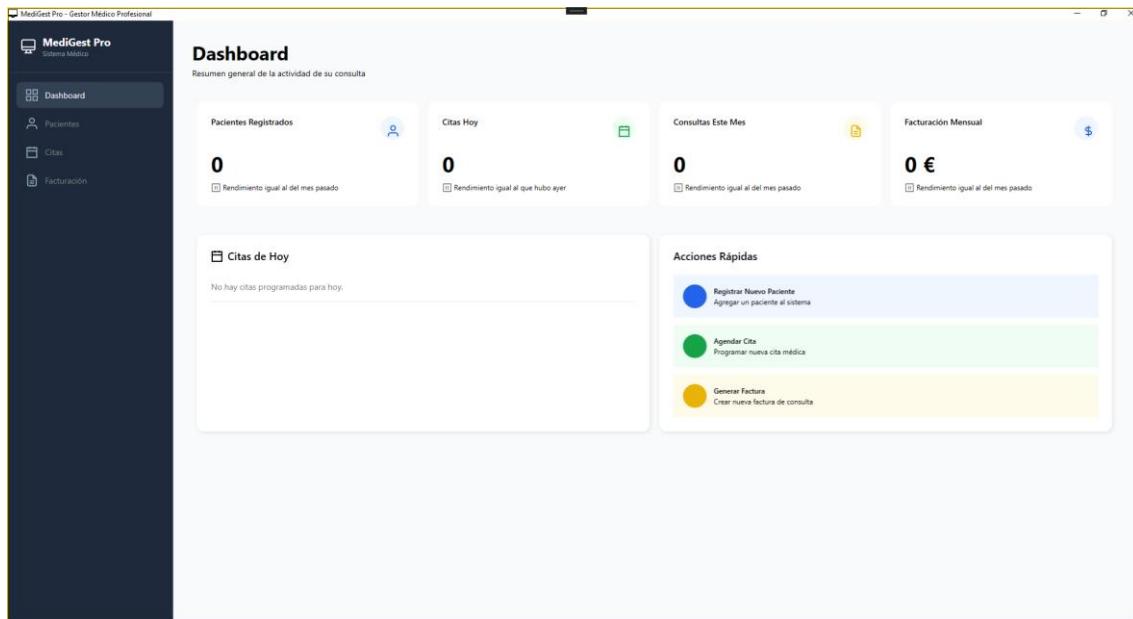
```
// --- Función auxiliar para generar hash SHA256 ---
private string CalcularSHA256(string texto)
{
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] bytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(texto));
        StringBuilder sb = new StringBuilder();
        foreach (byte b in bytes)
            sb.Append(b.ToString("x2")); // convierte a hexadecimal
        return sb.ToString();
    }
}
```





MediGest Pro

Dashboard



En el lateral izquierdo vemos nuestro menú de navegación o *MainWindow* que nos permitirá ir navegando por las distintas interfaces (*Pages*) de nuestra aplicación. Cada vez que un usuario inicie sesión esta será la interfaz a la que nos dirigirá por defecto llamada *Dashboard*.

Podemos ver en la parte superior una serie de valores que inicialmente se encuentran a 0. Esto se debe a que son valores pensados para ver la diferencia entre el mes anterior y el actual, apareciendo inmediatamente debajo el porcentaje negativo o positivo con respecto al mes anterior.

En la zona inferior derecha vemos las llamadas “Acciones rápidas” que son accesibles únicamente por los recepcionistas y si se intenta acceder como médico a ellas se mostrará un mensaje de que no tenemos los permisos necesarios.

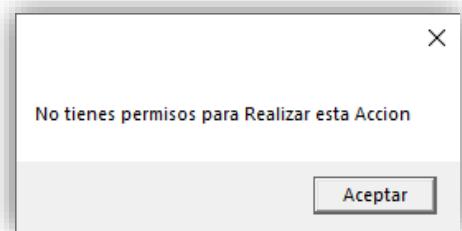
```

private void BtnRegistrarPacienteClick(object sender, RoutedEventArgs e)
{
    if (SessionManager.Rol == "Medico")
    {
        MessageBox.Show("No tienes permisos para Realizar esta Accion");
    }
    else {
        var form = new PanelPaciente();
        form.ShowDialog();
        RefrescarDashboard();
    }
}

private void BtnAgendarCitaClick(object sender, RoutedEventArgs e)
{
    if (SessionManager.Rol == "Medico")
    {
        MessageBox.Show("No tienes permisos para Realizar esta acción");
    }
    else {
        var form = new AgendarCita();
        form.ShowDialog();
        RefrescarDashboard();
    }
}

private void BtnGenerarFacturaClick(object sender, RoutedEventArgs e)
{
    if (SessionManager.Rol == "Medico")
    {
        MessageBox.Show("No tienes permisos para Realizar esta acción");
        return;
    }
    var form = new GenerarFactura();
    form.ShowDialog();
    RefrescarDashboard();
}

```



Iniciando sesión como recepcionistas podemos registrar nuevos pacientes introduciendo los datos en los campos que veremos a continuación. Durante este proceso se realizan diversas validaciones para asegurar la consistencia de datos. Entre ellas se comprueba que los campos obligatorios estén completos, la fecha de nacimiento sea anterior a la fecha actual o que el DNI sea válido. En caso de error el sistema mostrará un mensaje informativo.

El código completo que implementa estas validaciones puede consultarse en el [Anexo II – Código de validación del registro de pacientes](#) y los mensajes informativos en el [Anexo III.1 Registrar nuevo paciente](#).

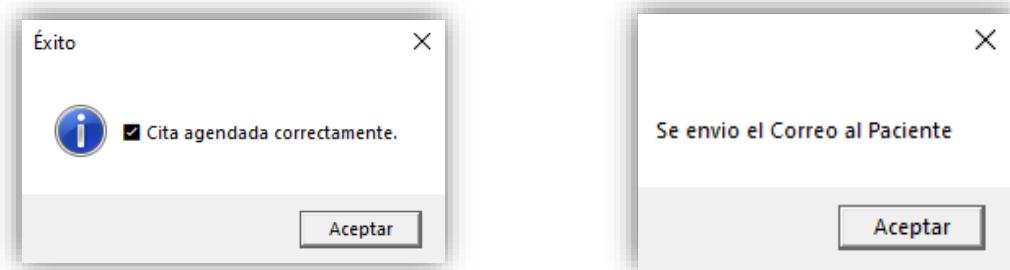


The screenshot shows two windows side-by-side. On the left is the 'Crear Paciente' (Create Patient) dialog box. It contains fields for Nombre (Name), Apellidos (Last Name), DNI (DNI), CIPA (CIPA), N° Historia Clínica (Clinical History No.), N° Seguridad Social (Social Security No.), Fecha nacimiento (Date of Birth), and Correo (Email). The 'Guardar' (Save) button is at the bottom right. On the right is an 'Exito' (Success) message box with the text 'Paciente registrado correctamente.' (Patient registered correctly.) and an 'Aceptar' (Accept) button.

Al igual que en la creación de nuevos pacientes, en el registro de nuevas citas, se verifica que todos los campos hayan sido completados correctamente. Entre las validaciones implementadas destaca la restricción que impide al usuario seleccionar fechas anteriores a la actual, evitando así la creación de citas vencidas. Asimismo, el sistema comprueba que no existan citas duplicadas o que interfieran en la planificación garantizando la coherencia.

Los mensajes informativos generados por la aplicación pueden consultarse en el [Anexo III.2 Agendar cita](#).

The screenshot shows the 'Agendar Cita' (Schedule Appointment) dialog box. It has two main sections: 'Seleccionar Fecha' (Select Date) on the left and 'Detalles de la Cita' (Appointment Details) on the right. The 'Seleccionar Fecha' section displays a calendar for December 2025, with the 12th highlighted. The 'Detalles de la Cita' section includes dropdown menus for 'Paciente' (Patient) set to 'Andrea Palacios Jimenez', 'Médico' (Doctor) set to 'Ana Rodríguez Pérez', 'Hora' (Hour) set to '10:00', and 'Estado' (Status) set to 'Pendiente' (Pending). There is also a text area for 'Observaciones' (Observations) containing 'Acudir en ayunas'. A 'Agendar Cita' (Schedule Appointment) button is located at the bottom right.



Como vimos en la creación de la base de datos los médicos, recepcionistas y pacientes tiene un correo el cual utilizamos para notificar a los pacientes sobre nuevas citas, modificaciones... El primer caso en el que vemos que se establece esta comunicación entre el personal del centro con un paciente es al agendar una nueva cita.

Si en el día de hoy existen citas programadas se verán a la izquierda en nuestro *Dashboard*, apareciendo tanto las creadas en el momento como las que se encuentras en la base de datos con fecha de hoy mostrando datos significativos. Esta información es visible para todos los usuarios.



Por último, la generación de facturas. Las facturas serán generadas teniendo en cuenta una fecha de inicio y otra de fin, por lo que todas las citas de un paciente seleccionado que hayan tenido lugar en ese periodo de tiempo aparecerán en la factura. En todo momento el sistema informa al usuario mediante mensajes si la creación de la factura ha sido exitosa o no, la ubicación en la que se guarda la factura creada... para una vez generada correctamente



MediGest Pro

esta se mostrará de forma automática pudiendo ver así el aspecto final sin necesidad de buscarla en su ubicación.

Para visualizar los mensajes del sistema acudir al [Anexo III.3. Generar factura](#).

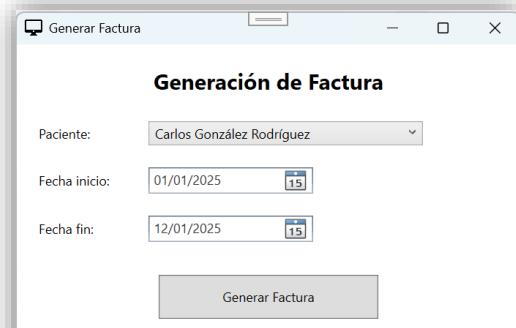
```
<TextBlock Text="Generación de Factura"
FontSize="20" FontWeight="Bold"
HorizontalAlignment="Center" Margin="0,0,0,20"/>

<StackPanel Orientation="Horizontal" Margin="0,50,0,10">
<TextBlock Text="Paciente:" Width="100" VerticalAlignment="Center"/>
<ComboBox x:Name="CmbPaciente" Width="250"/>
</StackPanel>

<StackPanel Orientation="Horizontal" Margin="0,10,0,10" Grid.Row="1">
<TextBlock Text="Fecha inicio:" Width="100" VerticalAlignment="Center"/>
<DatePicker x:Name="FechaInicioPicker" Width="150"/>
</StackPanel>

<StackPanel Orientation="Horizontal" Margin="0,10,0,10" Grid.Row="2">
<TextBlock Text="Fecha fin:" Width="100" VerticalAlignment="Center"/>
<DatePicker x:Name="FechaFinPicker" Width="150"/>
</StackPanel>

<Button x:Name="BtnGenerarFactura"
Grid.Row="3"
Content="Generar Factura"
Width="200" Height="40"
HorizontalAlignment="Center"
Click="BtnGenerarFactura_Click"
Margin="0,20,0,0"/>
```



Si se desea se puede consultar el [Anexo IV – Código de generar factura](#) dónde se encontrará el código de la generación del documento PDF.

Correo electrónico

La secuencia interna para enviar los correos es la misma en todos los casos realizados en este proyecto. Gracias a los campos de correo que

almacenamos en la base de datos podemos obtener de manera individual el valor del emisor y el receptor de este email. Para cada caso se han desarrollado plantillas en lenguaje html con un estilo similar, pero con variación en el contenido del mensaje pudiendo así diferenciar la finalidad de ese correo. Dentro de cada plantilla disponemos de campos que serás sustituidos por valores reales de la base de datos para realizar correos personalizados con el nombre del paciente al que va dirigido, el motivo del email, datos importantes que deba conocer el paciente en ese momento, el logo...

Para ello ha sido necesario crear dos correos en Gmail: medicosmedigestinforma@gmail.com para los médicos y recepcionistas y pacientesmedigest@gmail.com para los pacientes. Esto se ha hecho para controlar esos envíos y recepciones de correos, visualizar el contenido, el formato, vista de las pruebas realizadas... Una vez creados los correos ha sido necesario generar una contraseña de aplicación del correo que tomará el papel de emisor. Durante el desarrollo de esta funcionalidad ha sido necesario desarrollar una clase llamada *EmailService.cs* donde se carga la plantilla correspondiente, se recoge el usuario y contraseña de aplicación del emisor, envía el correo...

Para poder ver el contenido de esta clase se debe consultar el [Anexo V.1 – Envío.](#)

En las imágenes anteriores comprobamos que se ha enviado correctamente la notificación por correo una nueva cita siento este el resultado visto desde el correo del médico y desde el correo del paciente.



MediGest Pro

The screenshot shows a Gmail inbox with a single sent email. The subject of the email is "Cita Médica Programada". The email is from "medicosmedigestinforma@gmail.com" to "pacientesmedigest@gmail.com". The date is Dec 10, 2025, 12:51PM. The subject is "Cita Médica Programada" and the "mailed-by" is "gmail.com". The body of the email starts with "Hola Andrea Palacios Jimenez," followed by a message in Spanish confirming the appointment details: Fecha: 10/12/2025, Hora: 09:00, Médico: Ana Rodriguez Pérez, Especialidad: Cardiología, Observaciones: Acudir en ayunas, Estado: Pendiente. It also includes a note asking the patient to let them know if they can't attend and ends with "Hasta pronto,".

A continuación, veremos las distintas plantillas que hemos creado para las diferentes casuísticas donde hacemos uso de ellas. En primero lugar vemos el formato que tiene el mensaje recibido cuando se crea una cita, en segundo lugar, cuando una cita asignada a un paciente es modificada y por último cuando un médico quiere notificar algo a alguno de sus pacientes de manera personalizada teniendo que redactar él el cuerpo del mensaje.

Para poder ver al estilo por el que se rigen todas las plantillas deberá consultarse el [Anexo V.2 Estilo de plantilla](#).

```
<body>
<div class="container">
    <!-- LOGO -->
    <div class="logo">
        
    </div>

    <div class="header">
        <span> Cita Médica Programada
    </div>

    <div class="content">
        <p>Hola <span class="highlight">{{PacienteNombre}}</span>,</p>
        <p>Se ha registrado correctamente tu cita médica con los siguientes detalles:</p>
        <p>
            <b>Fecha:</b> {{Fecha}} <br>
            <b>Hora:</b> {{Hora}} <br>
            <b>Médico:</b> {{MedicoNombre}} <br>
            <b>Especialidad:</b> {{Especialidad}}<br>
            <b>Observaciones:</b> {{Observaciones}}<br>
            <b>Estado:</b> {{Estado}}
        </p>
        <p>Si no puedes asistir, te agradecemos que nos avises con antelación.</p>
        <p>Hasta pronto,</p>
    </div>

    <div class="footer">
        Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo.
    </div>
</div>
</body>
</html>
```

The screenshot shows a generated HTML email with the subject "Cita Médica Programada". The recipient is "Hola Andrea Palacios Jimenez,". The email details a scheduled medical appointment for Andrea Palacios Jimenez on December 10, 2025, at 09:00 AM with Dr. Ana Rodriguez Perez, Cardiologist, with instructions to fast before the appointment. The footer of the email states "Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo."

```

<body>
  <div class="container">
    <!-- LOGO -->
    <div class="logo">
      
    </div>

    <div class="header">
      <span> Actualización de Cita Médica
    </div>

    <div class="content">
      <p>Hola <span class="highlight">{{PacienteNombre}}</span>,</p>
      <p>Queremos informarte que el estado de tu cita ha cambiado a:</p>
      <p><b>{{NuevoEstado}}</b></p>

      <p>📅 <b>Fecha:</b> {{Fecha}}</p>
      <p>⌚ <b>Hora:</b> {{Hora}}</p>
      <p>👤 <b>Médico:</b> {{MedicoNombre}}</p>

      {{MensajeOpcional}}

      <p>Si necesitas modificar tu cita o tienes alguna duda, contáctanos.</p>
    </div>

    <div class="footer">
      <p>Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo.</p>
    </div>
  </div>
</body>
</html>

```



Actualización de Cita Médica

Hola Luis Martínez López,

Queremos informarte que el estado de tu cita ha cambiado a:

✓ Cancelada

📅 Fecha: 09/12/2025

⌚ Hora: 16:00

👤 Médico: Carmen Moreno López

⚠️ Tu cita ha sido cancelada.

Si necesitas modificar tu cita o tienes alguna duda, contáctanos.

...

Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo.

```

<body>
  <div class="container">
    <!-- LOGO -->
    <div class="logo">
      
    </div>

    <div class="header">
      MediGest Pro - Consulta Médica
    </div>

    <div class="content">
      <p>Hola <span class="highlight">{{PacienteNombre}}</span>,</p>
      <p>{{Mensaje}}</p>
      <p>Atentamente,</p>
      <p><b>{{MedicoNombre}}</b></p>
    </div>

    <div class="footer">
      <p>Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo.</p>
    </div>
  </div>
</body>
</html>

```



MediGest Pro - Consulta Médica

Hola Ángel Castro Sánchez,

El dia 11 de diciembre de 2025 habrá un cambio en el nombre de la medicación

Atentamente,

Ana Rodríguez Pérez

Este correo ha sido generado automáticamente por MediGest Pro. Por favor, no responda a este correo.

Pacientes

Nuestra interfaz de pacientes tiene como finalidad listar a los pacientes que están dados de alta. En este punto podemos observar otra de las diferencias entre iniciar sesión como médico o como recepcionista, ya que estos primeros únicamente podrán ver los pacientes que tengan asignados, mientras que los recepcionistas verán a todos los que están en el sistema. Otra de las grandes diferencias entre ambos son las acciones que van a poder realizar, siendo el médico el que puede ver los informes médicos, generar un PDF de ellos y enviarles un correo mientras que el recepcionista podrá modificar y ver los datos personales que me muestren en el *datagrid*. Ambos usuarios podrán filtrar por varios campos que son:

- Nombre
- Fecha completa de cumpleaños
- Año de nacimiento



MediGest Pro

- Mes de nacimiento

Introduce nombre del Paciente a Bus	Seleccione una fecha	Año	Mes	Buscar	Limpiar
-------------------------------------	----------------------	-----	-----	--------	---------

Médico

Nombre	DNI	CIPA	Historia Clínica	Seguridad Social	Fecha de Nacimiento
Juan Pérez García	12345678Z	CIPA001	HC001	281234567890	3/15/1985 12:00:00 AM
José Hernández López	11234567J	CIPA011	HC011	281234567800	5/12/1993 12:00:00 AM
Ángel Castro Sánchez	12345679T	CIPA021	HC021	281234567810	8/15/1996 12:00:00 AM
Andrés Garrido Lopez	13456789D	CIPA031	HC031	281234567820	12/22/1987 12:00:00 AM

Recepcionista

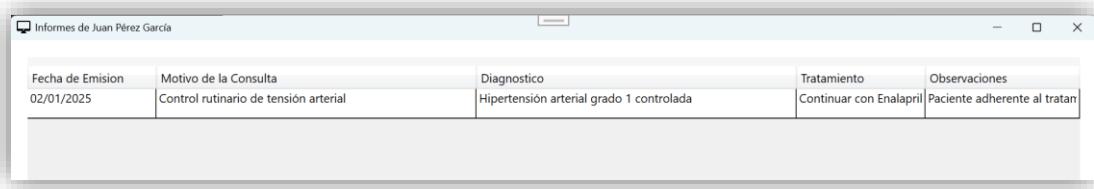
Nombre	DNI	CIPA	Historia Clínica	Seguridad Social	Fecha de Nacimiento
Juan Pérez García	12345678Z	CIPA001	HC001	281234567890	3/15/1985 12:00:00 AM
Maria López Martinez	23456789A	CIPA002	HC002	281234567891	7/22/1990 12:00:00 AM
Carlos González Rodriguez	34567890B	CIPA003	HC003	281234567892	11/30/1978 12:00:00 AM
Ana Fernández Sánchez	45678901C	CIPA004	HC004	281234567893	1/10/1995 12:00:00 AM
Luis Martínez López	56789012D	CIPA005	HC005	281234567894	6/18/1982 12:00:00 AM
Elena Sánchez Pérez	67890123E	CIPA006	HC006	281234567895	9/25/1988 12:00:00 AM
David Rodríguez García	78901234F	CIPA007	HC007	281234567896	12/5/1975 12:00:00 AM
Laura García Martínez	89012345G	CIPA008	HC008	281234567897	4/14/1992 12:00:00 AM
Miguel Martín Fernández	90123456H	CIPA009	HC009	281234567898	8/20/1980 12:00:00 AM
Carmen Jiménez Ruiz	01234567I	CIPA010	HC010	281234567899	2/28/1987 12:00:00 AM
José Hernández López	11234567J	CIPA011	HC011	281234567800	5/12/1993 12:00:00 AM
Isabel Díaz Moreno	21234567K	CIPA012	HC012	281234567801	10/3/1984 12:00:00 AM
Antonio Moreno González	31234567L	CIPA013	HC013	281234567802	1/19/1976 12:00:00 AM
Rosa Álvarez Sánchez	41234567M	CIPA014	HC014	281234567803	7/8/1991 12:00:00 AM
Francisco Romero Pérez	51234567N	CIPA015	HC015	281234567804	3/27/1989 12:00:00 AM
Pilar Torres García	61234567O	CIPA016	HC016	281234567805	11/16/1983 12:00:00 AM
Manuel Navarro Martínez	71234567P	CIPA017	HC017	281234567806	6/23/1994 12:00:00 AM
Dolores Gil Rodríguez	81234567Q	CIPA018	HC018	281234567807	9/1/1979 12:00:00 AM
Ramón Serrano López	91234567R	CIPA019	HC019	281234567808	12/30/1986 12:00:00 AM
Teresa Blanco Fernández	02345678S	CIPA020	HC020	281234567809	4/7/1977 12:00:00 AM
Ángel Castro Sánchez	12345679T	CIPA021	HC021	281234567810	8/15/1996 12:00:00 AM
Mercedes Ortiz García	22345678U	CIPA022	HC022	281234567811	1/24/1981 12:00:00 AM
Rafael Rubio Martínez	32345678V	CIPA023	HC023	281234567812	5/2/1988 12:00:00 AM
Concepción Molina Pérez	42345678W	CIPA024	HC024	281234567813	10/19/1974 12:00:00 AM
Javier Morales López	52345678X	CIPA025	HC025	281234567814	12/14/1992 12:00:00 AM

Habiendo iniciado sesión con una cuenta de médico veremos las acciones que se pueden hacer, cómo se hacen y con el botón derecho sobre el paciente podremos acceder a ellas.

Nombre	DNI
Juan Pérez García	12345678Z
José Hernández López	11234567J
Ángel Castro Sánchez	12345679T

[Ver Informes Médicos](#)
[Generar PDF de Informes Médicos](#)
[Enviar correo](#)

[Ver informes médicos](#)



Como vimos anteriormente la generación de pdf una vez generado y guardado en la ubicación indicada mostramos automáticamente el contenido generado.

Generar PDF de Informes Médicos



HISTORIAL MEDICO DEL PACIENTE

DATOS DEL PACIENTE

Nombre completo: Juan Pérez García
DNI: 12345678Z
CIPA: CIPA001
Nº Historia Clínica: HC001
Nº Seguridad Social: 281234567890
Fecha de Nacimiento: 15/03/1985
Edad: 40 años

RESUMEN DE CONSULTAS

Total de consultas registradas: 1
Primer consulta: 02/01/2025
Última consulta: 02/01/2025

HISTORIAL DE INFORMES MEDICOS

INFORME #1 - ID: 1

Fecha de emisión: 02/01/2025
Fecha de cita: 02/01/2025 - 09:00:00
Médico: Ana Rodríguez Pérez
Especialidad: Cardiología
Nº Colegiado: 282801234

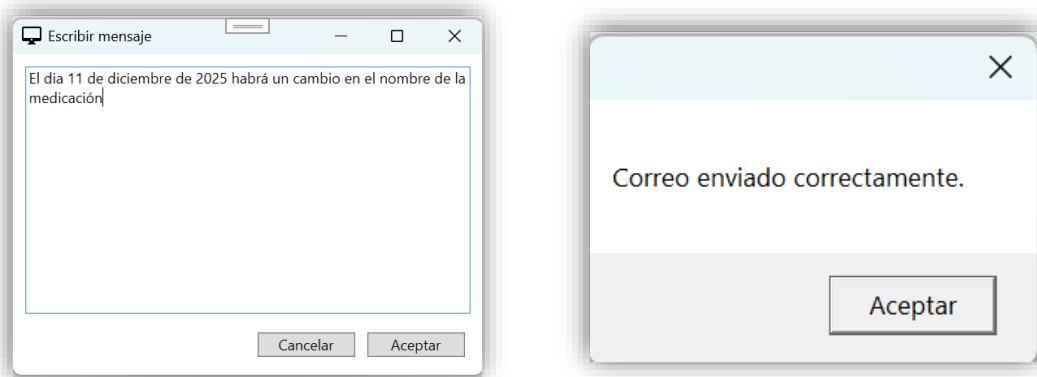
Motivo de consulta:
Control rutinario de tensión arterial
Diagnóstico:

Hipertensión arterial grado 1 controlada
Tratamiento prescrito:
Continuar con Enalapril 10mg/día. Dieta hiposódica. Control en 3 meses.
Observaciones:
Paciente adherente al tratamiento. Tensión arterial 135/85 mmHg.

Documento generado el 10/12/2025 a las 22:49
DOCUMENTO CONFIDENCIAL - USO MEDICO EXCLUSIVO

El código para generar este PDF se encuentra en el anexo y se puede consultar exactamente en [Anexo VI – Código de generar pdf de informes médicos](#).

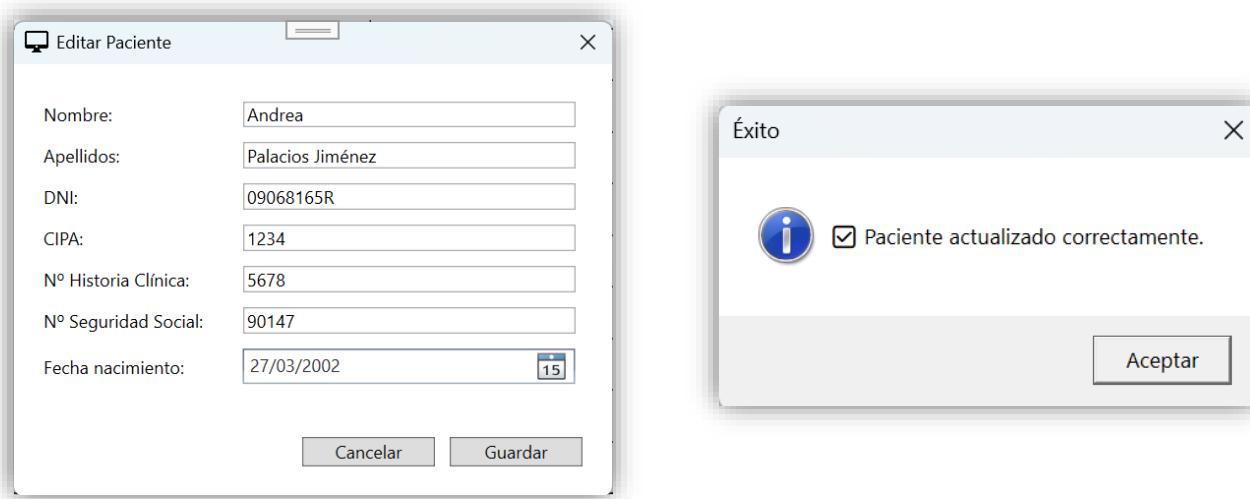
Cuando un médico desea enviar un email a un paciente deberá ser él mismo el que redacte el mensaje, el cual se visualizará una vez enviado al correo en la plantilla creada para este caso. Dicha plantilla es *correo.html*.



Como ya mencionamos, el usuario recepcionista únicamente puede modificar los datos de los pacientes haciendo doble click sobre el paciente que se desea modificar. Al igual que en la creación de nuevos usuarios controlamos las mismas cosas (campos vacíos, la fecha de nacimiento, DNI válido...).

Estos mensajes enviados por el sistema se pueden consultar en el [Anexo](#)

[III.4 Editar paciente.](#)



Citas

Muy similar a la interfaz de los pacientes. Visualización y filtrado de las citas que hay en la base de datos y al igual que en pacientes, el recepcionista puede ver todas las citas registradas mientras que el médico únicamente las que tiene asignadas y solo podrá cambiar el estado de las citas que sean posteriores a la fecha de hoy. Ambos pueden filtrar por los mismos campos que en pacientes.

Médico

Fecha	Hora	Paciente	Médico	Estado	Observaciones
02/01/2025	16:00:00	Laura García Martínez	Rosa Álvarez Romero	realizada	Chequeo general
03/01/2025	16:00:00	Dolores Gil Rodríguez	Rosa Álvarez Romero	realizada	Dolor abdominal
06/01/2025	16:00:00	Antonia Muñoz Martínez	Rosa Álvarez Romero	realizada	Gripe
07/01/2025	16:00:00	Inmaculada Herrera López	Rosa Álvarez Romero	realizada	Faringitis
08/01/2025	16:00:00	Ángeles Pascual Pérez	Rosa Álvarez Romero	realizada	Gastritis
09/01/2025	16:00:00	Catalina Lorenzo Sánchez	Rosa Álvarez Romero	realizada	Bronquitis crónica
10/01/2025	16:00:00	Sofía Bravo Martínez	Rosa Álvarez Romero	realizada	Anemia
13/01/2025	16:00:00	Araceli Estévez García	Rosa Álvarez Romero	realizada	Reflujo gástricoesofágico
14/01/2025	16:00:00	Lorena Suárez López	Rosa Álvarez Romero	realizada	Colitis ulcerosa
15/01/2025	16:00:00	Gema Velázquez Pérez	Rosa Álvarez Romero	realizada	Úlcera péptica
16/01/2025	16:00:00	Rubén Crespo Sánchez	Rosa Álvarez Romero	realizada	Síndrome de intestino irritable
17/01/2025	16:00:00	Tania Romero Fernández	Rosa Álvarez Romero	realizada	Hepatitis viral
20/01/2025	16:00:00	Mónica Delgado Martínez	Rosa Álvarez Romero	realizada	Cirrosis hepática
21/01/2025	16:00:00	Tatiana Molina García	Rosa Álvarez Romero	realizada	Pancreatitis aguda
22/01/2025	16:00:00	Leopoldo Muñoz López	Rosa Álvarez Romero	realizada	Enfermedad de Crohn
23/01/2025	16:00:00	Hilario Bravo Pérez	Rosa Álvarez Romero	realizada	Diverticulitis
24/01/2025	16:00:00	Cipriano Ferrer Sánchez	Rosa Álvarez Romero	realizada	Apendicitis
27/01/2025	16:00:00	Saturnino Estévez Martínez	Rosa Álvarez Romero	realizada	Coletiásis
28/01/2025	16:00:00	Laureano Pastor García	Rosa Álvarez Romero	realizada	Obstrucción intestinal
29/01/2025	16:00:00	Pantaleón Benito López	Rosa Álvarez Romero	realizada	Hernia hiatal
03/02/2025	16:00:00	Dolores Gil Rodríguez	Rosa Álvarez Romero	realizada	Estreñimiento crónico
04/02/2025	16:00:00	Antonia Muñoz Martínez	Rosa Álvarez Romero	realizada	Enfermedad celíaca
05/02/2025	16:00:00	Inmaculada Herrera López	Rosa Álvarez Romero	realizada	Hemorroides
06/02/2025	16:00:00	Ángeles Pascual Pérez	Rosa Álvarez Romero	realizada	Fisura anal
07/02/2025	16:00:00	Catalina Lorenzo Sánchez	Rosa Álvarez Romero	realizada	Fistula anal

Recepcionista

Fecha	Hora	Paciente	Médico	Estado	Observaciones
29/01/2025	10:30:00	Abundio Crespo Martínez	Miguel Díaz Hernández	realizada	Pitiriasis versicolor
07/02/2025	16:30:00	Adrián Hidalgo Pérez	Pedro Torres Navarro	realizada	Trastorno dismórfico corporal
11/04/2025	12:00:00	Agustín Giménez Sánchez	Francisco Jiménez Fernández	pendiente	Terapia ocupacional
07/01/2025	12:00:00	Agustín Giménez Sánchez	Francisco Jiménez Fernández	realizada	Ataxia
07/03/2025	12:00:00	Aitana Lorenzo Pérez	Francisco Jiménez Fernández	pendiente	Polisomnografía
06/02/2025	10:00:00	Alberto Cruz Pérez	Elena Sánchez Martín	realizada	Adenoiditis
10/02/2025	12:00:00	Alejandro Román García	Francisco Jiménez Fernández	realizada	Neuritis óptica
06/01/2025	16:30:00	Alfonso Iglesias Sánchez	Pedro Torres Navarro	realizada	Depresión
02/04/2025	16:30:00	Alfonso Iglesias Sánchez	Pedro Torres Navarro	pendiente	Terapia grupo
20/01/2025	11:30:00	Alicia Luna López	Carmen Moreno López	realizada	Prolapso uterino
11/02/2025	16:30:00	Álvaro Soler Martínez	Pedro Torres Navarro	realizada	Síndrome de Tourette
05/02/2025	11:30:00	Amparo León García	Carmen Moreno López	realizada	Adenomiosis
25/12/2025	13:30:00	Andrea Palacios Jiménez	Isabel Ruiz García	confirmada	
05/02/2025	09:00:00	Andrés Garrido López	Ana Rodríguez Pérez	realizada	Síncope cardiogénico
02/04/2025	09:00:00	Ángel Castro Sánchez	Ana Rodríguez Pérez	pendiente	Ecocardiograma transesofágico
06/01/2025	09:00:00	Ángel Castro Sánchez	Ana Rodríguez Pérez	realizada	Arritmia cardíaca
06/02/2025	16:00:00	Ángeles Pascual Pérez	Rosa Álvarez Romero	realizada	Fisura anal
12/02/2025	10:30:00	Angustias Gallardo Martínez	Miguel Díaz Hernández	realizada	Tiña corporis
27/01/2025	10:30:00	Anselmo Soler Sánchez	Miguel Díaz Hernández	realizada	Pitiriasis rosada
04/02/2025	16:00:00	Antonia Muñoz Martínez	Rosa Álvarez Romero	realizada	Enfermedad celíaca
03/01/2025	10:00:00	Antonio Moreno González	Elena Sánchez Martín	realizada	Control de crecimiento
01/04/2025	10:00:00	Antonio Moreno González	Elena Sánchez Martín	pendiente	Vacuna papiloma humano
28/01/2025	11:00:00	Aquilina Flores Sánchez	Isabel Ruiz García	realizada	Queratitis
09/04/2025	16:00:00	Araceli Estévez García	Rosa Álvarez Romero	pendiente	Tránsito intestinal
13/01/2025	16:00:00	Araceli Estévez García	Rosa Álvarez Romero	realizada	Reflujo gástricoesofágico

Siendo recepcionista podemos modificar y borrar las citas. Para acceder a estas acciones debemos saber que el botón derecho nos permite borrar comprobando antes si estamos seguros y el doble click modificar. Aquellas citas que tengan un estado como que ya han sido realizadas no podrán modificarse,

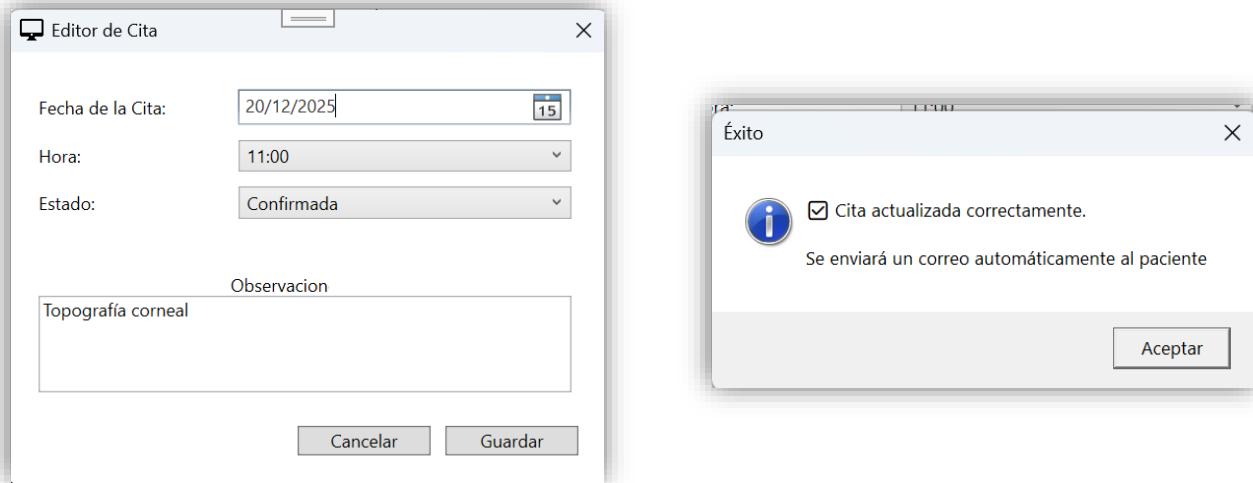


MediGest Pro

pero si ser eliminadas, al igual que no se podrá asignar el estado de “realizada” a una cita que tiene una fecha posterior a la actual.

Lo mensajes que muestra el sistema pueden consultarse en el [Anexo III.5.](#)

[Editar y eliminar cita.](#)



Cuando una cita es modificada, el paciente debe saber esa información por lo que notificamos por correo los cambios gracias a la plantilla que diseñamos para ello llamada *citaEditada.html*.

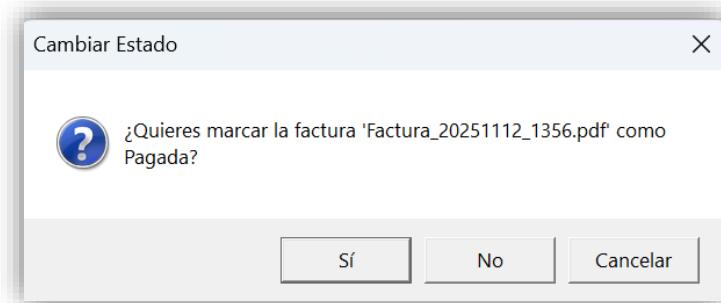
Facturación

En esta última pantalla podemos ver las facturas generadas, los ingresos totales que se han generado, aquello que está pendiente por cobrar y los que ya se han cobrado.

The screenshot shows the 'Facturación' (Billing) screen of the MediGest Pro system. On the left is a sidebar with icons for Dashboard, Pacientes, Citas, and Facturación (which is selected). The main area has a title 'Facturación' and a subtitle 'Consulta el estado de las facturas emitidas'. It displays three summary boxes: 'Ingresos Totales' (645,00 €), 'Pendientes' (645,00 €), and 'Pagadas' (0,00 €). Below these is a table titled 'Facturas emitidas' showing a list of issued invoices with columns for 'Factura', 'Estado' (Status), and 'Monto (€)' (Amount). The table lists several invoices, all marked as 'Pendiente' (Pending) with amounts ranging from 45 to 180.

Factura	Estado	Monto (€)
Factura_20251121_1356.pdf	Pendiente	90
Factura_20251121_0255.pdf	Pendiente	180
Factura_20251121_2020.pdf	Pendiente	45
Factura_20251210_0000.pdf	Pendiente	90
Factura_20251210_0000.pdf	Pendiente	80
Factura_20251210_1229.pdf	Pendiente	100
Factura_20251210_2021.pdf	Pendiente	60

Solo los recepcionistas pueden marcar facturas como que ya se han cobrado haciendo doble click sobre ellas y confirmando que se han pagado. Esta misma acción se puede hacer de manera inversa y una factura que consta como pagada pase a estar en un estado de pendiente de cobro.



Factura	Estado	Monto (€)
Factura_20251112_1356.pdf	Pagada	90
Factura_20251113_0255.pdf	Pendiente	180
Factura_20251113_2020.pdf	Pendiente	45
Factura_20251210_0000.pdf	Pendiente	90
Factura_20251210_0006.pdf	Pendiente	80
Factura_20251210_1229.pdf	Pendiente	100
Factura_20251210_2021.pdf	Pendiente	60

```

private void lstFacturas_MouseDoubleClick(object sender, System.Windows.Input.MouseEventArgs e)
{
    if (SessionManager.Rol == "Medico") {
        MessageBox.Show("No tienes permiso para cambiar el estado de las Facturas");
        return;
    }

    if (lstFacturas.SelectedItem is FacturaItem facturaSeleccionada)
    {
        // Mostrar opciones de estado
        var opcion = MessageBox.Show(
            $"¿Quieres marcar la factura '{facturaSeleccionada.Nombre}' como Pagada?",
            "Cambiar Estado",
            MessageBoxButton.YesNoCancel,
            MessageBoxIcon.Question
        );

        if (opcion == MessageBoxResult.Yes)
            facturaSeleccionada.Estado = "Pagada";
        else if (opcion == MessageBoxResult.No)
            facturaSeleccionada.Estado = "Pendiente";
        else
            return;

        // Refrescar el ListView
        lstFacturas.Items.Refresh();

        // Recalcular los totales
        RecacularTotales();
    }
}

```



MediGest Pro

```
1 referencia
private void RecalcularTotales()
{
    double totalGeneral = 0, totalPendientes = 0, totalPagadas = 0;

    foreach (FacturaItem item in lstFacturas.Items)
    {
        totalGeneral += item.Monto;
        if (item.Estado == "Pendiente") totalPendientes += item.Monto;
        else if (item.Estado == "Pagada") totalPagadas += item.Monto;
    }

    txtTotalIngresos.Text = $"{totalGeneral:F2} €";
    txtPendientes.Text = $"{totalPendientes:F2} €";
    txtPagadas.Text = $"{totalPagadas:F2} €";
}
```

Administrador

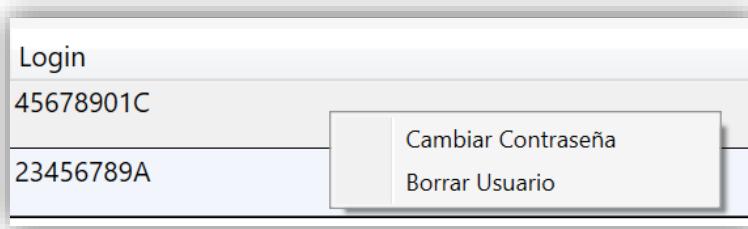
El rol del administrador no es está mas que para hacer la gestión de los usuarios, modificar y borrar.

- **Usuarios**

The screenshot shows the 'Panel del Administrador' (Administrator Panel) for 'MediGest Pro Admin'. On the left, there's a sidebar with 'Panel Administrativo del Sistema Médico' and three menu items: 'Usuarios' (selected), 'Médicos', and 'Recepcionistas'. The main area has a header with 'Rol' dropdown, 'Crear' (Create) button, 'Buscar' (Search) button, and 'Limpiar' (Clear) button. Below the header is a message: 'Click derecho en una fila para el Menú de opciones' (Right-click on a row for options menu). A table lists users with columns: 'Login', 'Contraseña' (Password), and 'Rol'. The table contains 18 rows of user data.

Login	Contraseña	Rol
45678901C	*****	admin
23456789A	*****	receptionista
12345678Z	*****	receptionista
87523388Q	*****	receptionista
34567890B	*****	receptionista
57142851H	*****	receptionista
56789012D	*****	medico
67890123E	*****	medico
78901234F	*****	medico
89012345G	*****	medico
90123456H	*****	medico
01234567I	*****	medico
11234567J	*****	medico
21234567K	*****	medico
31234567L	*****	medico
41234567M	*****	medico

Con el botón derecho desplegamos las acciones de cambiar contraseña y borrar usuario y haciendo doble click podemos modificar el nombre de usuario.



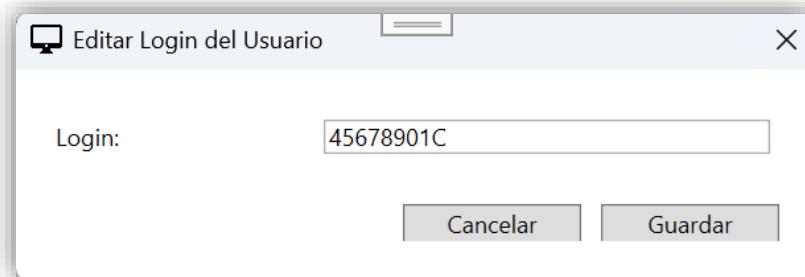
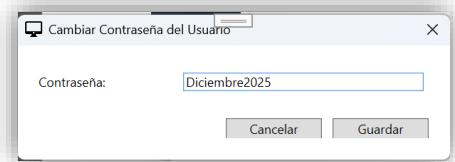
Al cambiar la contraseña debemos encriptar la nueva para así mantener la seguridad en nuestra base de datos y la concordancia con el resto del funcionamiento del sistema.

```
1 referencia
private void btnGuardar_Click(object sender, RoutedEventArgs e)
{
    if (txtPassword.Text == "") {
        MessageBox.Show("La contraseña no puede estar vacía");
        return;
    }
    var passwordHash = CalcularSHA256(txtPassword.Text);
    using (var db = new MediGestContext())
    {
        var user = db.Usuario.FirstOrDefault(u =>
            u.Id_usuario == usuarioActual.Id_usuario);

        if (user == null) {
            MessageBox.Show("No se encontró el usuario en la base de datos.",
                "Error", MessageBoxButton.OK, MessageBoxImage.Error);
            return;
        }

        if (user.Password == passwordHash)
        {
            MessageBox.Show("Esa ya es la Contraseña Actual introduzca una diferente");
            return;
        }

        user.Password = passwordHash;
        db.SaveChanges();
        MessageBox.Show("Contraseña Actualizada");
        this.Close();
    }
}
```



Para la creación de nuevos usuarios debemos seleccionar el botón que se encuentra en la misma sección reservada para la búsqueda por filtrado. Tras completar todos los campos correctamente se creará un nuevo usuario con el rol asignado que se verá automáticamente en el *datagrid*.



MediGest Pro

Rol

Crear

Buscar

Limpiar

Click derecho en una fila para el Menu de opciones

Crear Usuario

Login:	<input type="text" value="09068165R"/>
Contraseña:	<input type="password" value="usuario1prueba"/>
Confirmacion:	<input type="password" value="*****"/>
Rol:	<input type="text" value="Repcionista"/>

Cancelar

Guardar

Procediendo a la Creacion del Repcionista...

Aceptar

Repcionista

Crear

Buscar

Limpiar

Click derecho en una fila para el Menu de opciones

Login	Contraseña	Rol
12345678Z	*****	repcionista
87523388Q	*****	repcionista
34567890B	*****	repcionista
57142851H	*****	repcionista
09068165R	*****	repcionista

- Médicos

Introduce nombre del Medico a Buscar

Especialidad

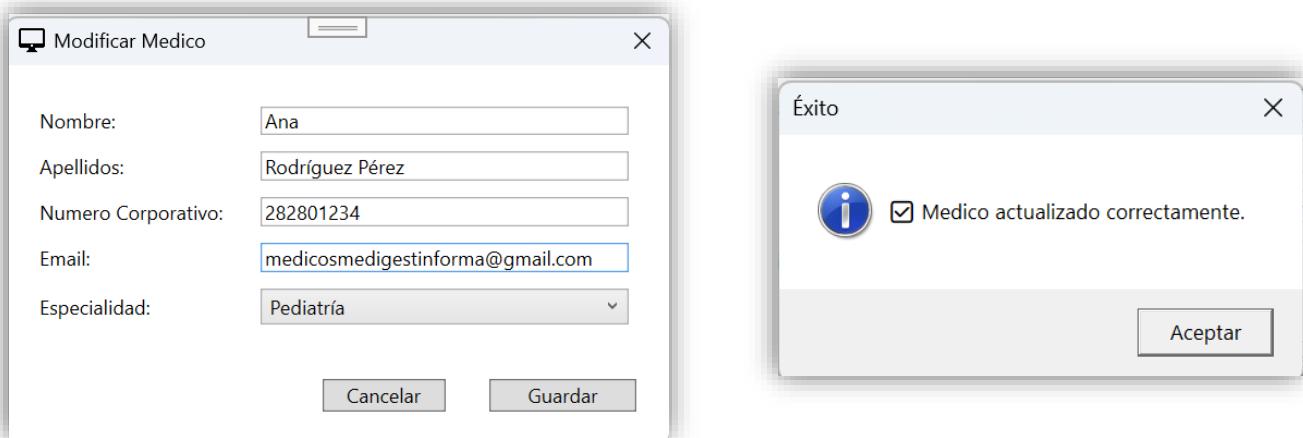
Buscar

Limpiar

Click derecho en una fila para el Menu de opciones

Nombre	Numero de Colegiado	Correo Corporativo
Ana Rodriguez Pérez	282801234	ana.rodriguez@medigest.es
Javier Gonzalez Moreno	282802345	javier.gonzalez@medigest.es
Elena Sánchez Martín	282803456	elena.sanchez@medigest.es
Miguel Díaz Hernández	282804567	miguel.diaz@medigest.es
Isabel Ruiz García	282805678	isabel.ruiz@medigest.es
Carmen Moreno López	282806789	carmen.moreno@medigest.es
Francisco Jiménez Fernández	282807890	francisco.jimenez@medigest.es
Rosa Álvarez Romero	282808901	rosa.alvarez@medigest.es
Pedro Torres Navarro	282809012	pedro.torres@medigest.es
Lucía Ramírez Gil	282810123	lucia.ramirez@medigest.es

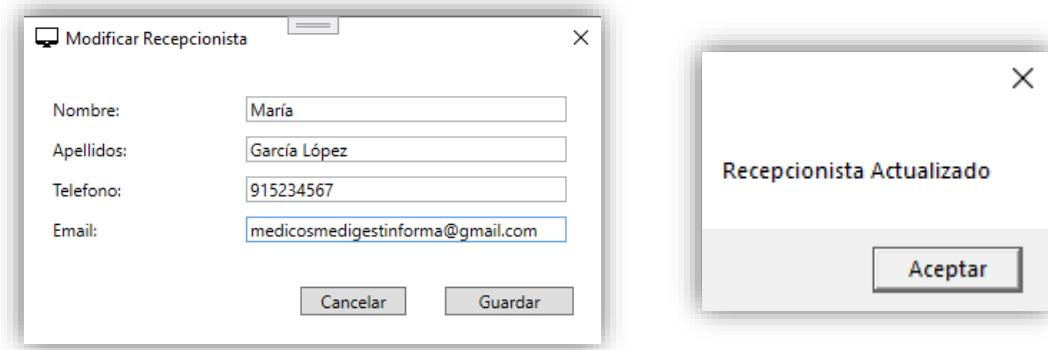
Con doble click podemos acceder a la funcionalidad de modificar los datos de estos usuarios.



- **Recepcionistas**

Como en las anteriores se realizan las mismas acciones permitiendo cambiar datos erróneos y corregir información mal guardada en la base de datos.

Panel del Administrador			
MediGest Pro Admin			
Panel Administrativo del Sistema Médico			
Usuarios	Introduce nombre del Recepcionista	Buscar	Limpiar
Medicos			
Recepcionistas			
	Maria Garcia Lopez	915234567	correo Electronico maria.garcia@medigest.es
	Carlos Fernandez Ruiz	914567890	carlos.fernandez@medigest.es
	Laura Martinez Sanchez	913456789	laura.martinez@medigest.es
	Antonio Lopez Jimenez	912345678	antonio.lopez@medigest.es





CONCLUSIONES Y MEJORAS

Conclusiones Generales

El desarrollo de *MediGest Pro* ha permitido cumplir con el objetivo principal planteado al inicio del proyecto: crear una solución de software integral, robusta y visualmente intuitiva para la gestión de clínicas privadas. La aplicación aborda eficazmente la problemática de la gestión manual y la dispersión de datos, centralizando la información de pacientes, citas, historiales médicos y facturación en una única plataforma de escritorio desarrollada en **WPF**, ofreciendo una experiencia de usuario fluida y sin latencias, algo crítico en el entorno sanitario.

Evaluación del Trabajo y Cumplimiento de Objetivos

Tras la fase de pruebas y despliegue, se confirma que el sistema cumple con los requisitos funcionales establecidos en el análisis:

- **Gestión Clínica y Administrativa:** Se ha implementado con éxito el ciclo completo de gestión, desde el registro de pacientes con validaciones de datos (DNI, fechas) hasta la asignación de citas médicas controlando solapamientos.
- **Seguridad:** Se ha garantizado la protección de acceso mediante un sistema de roles (Administrador, Médico, Recepcionista) y el almacenamiento seguro de contraseñas mediante algoritmos de **hashing (SHA-256)**, cumpliendo con los estándares básicos de seguridad.
- **Automatización y Digitalización:** La integración de librerías como *iTextSharp* ha permitido automatizar la generación de facturas e informes en PDF, y el uso del servicio SMTP de Gmail ha facilitado la comunicación directa con el paciente mediante notificaciones por correo electrónico.
- **Interfaz de Usuario:** El uso de XAML en WPF ha permitido separar la lógica del diseño, resultando en una interfaz moderna y escalable, con un Dashboard funcional que ofrece métricas en tiempo real.

Limitaciones de la Aplicación

A pesar de que el software es funcional, el proyecto presenta ciertas limitaciones inherentes a su arquitectura actual y al entorno de desarrollo utilizado:

1. **Dependencia del Entorno Local (XAMPP):** Actualmente, la base de datos se aloja utilizando XAMPP. Si bien esta herramienta es excelente para el desarrollo y pruebas por su facilidad de configuración (Apache + MySQL), **no es una solución apta para un entorno de producción real**. XAMPP carece por defecto de las configuraciones de seguridad avanzadas necesarias para datos sensibles y su rendimiento es limitado para una alta concurrencia.
2. **Accesibilidad:** Al ser una aplicación de escritorio pura (Desktop), el acceso está restringido a la red local donde se encuentra el servidor. Esto impide que los médicos puedan consultar agendas o historiales fuera de la clínica (movilidad limitada).
3. **Escalabilidad del Despliegue:** La instalación requiere configurar el entorno en cada equipo cliente, lo que dificulta el mantenimiento y las actualizaciones en comparación con una solución web.

Aprendizaje Técnico y Profesional

El desarrollo de *MediGest Pro* ha supuesto una consolidación de conocimientos técnicos fundamentales:

- **Desarrollo Full-Stack en .NET:** Se ha profundizado en el dominio del lenguaje C# y el patrón de diseño MVVM (implícito en la estructura Pages/Clases) para conectar la interfaz gráfica con la lógica de negocio.
- **Gestión de Datos Relacionales:** El diseño del diagrama Entidad-Relación y su implementación en MySQL ha reforzado la capacidad para estructurar bases de datos complejas con integridad referencial.
- **Uso de Librerías de Terceros:** La integración de librerías externas (iText, System.Net.Mail) ha enseñado la importancia de no "reinventar la rueda" y utilizar herramientas existentes para potenciar la funcionalidad de la aplicación.



Mejoras Futuras

Para evolucionar *MediGest Pro* hacia un producto comercial competitivo, se proponen las siguientes líneas de desarrollo futuro:

1. **Migración a Arquitectura Híbrida o Web:** Desarrollar una interfaz web o una API que permita el acceso seguro a los datos desde fuera de la clínica, facilitando la telemedicina y el acceso móvil para los profesionales.
2. **Backup Automático y Seguridad Avanzada:** Implementar un sistema de copias de seguridad (backups) automatizadas y programadas para evitar la pérdida de datos, una funcionalidad crítica que actualmente no está automatizada.
3. **Historia Clínica Avanzada:** Ampliar el módulo de informes médicos para permitir no solo texto, sino la adjunción de archivos multimedia (radiografías, análisis en PDF) directamente en la base de datos o en un servidor de archivos seguro.
4. **Pasarela de Pagos y Facturación Electrónica:** Mejorar el módulo de facturación actual para integrar pasarelas de pago digitales y adaptar las facturas a los estándares de factura electrónica obligatorios en futuras normativas.
5. **Optimización del Servidor de Datos:** Migrar la base de datos de XAMPP a un servidor MySQL/MariaDB dedicado y optimizado para producción, preferiblemente en un servidor Linux con medidas de seguridad reforzadas.

BIBLIOGRAFÍA

¿Qué es Windows Presentation Foundation? - WPF. (s/f). Microsoft.com. Recuperado el 4 de diciembre de 2025, de <https://learn.microsoft.com/es-es/dotnet/desktop/wpf/overview/>

Diagramming powered by intelligence. (s/f). Lucidchart. Recuperado el 4 de diciembre de 2025, de <https://www.lucidchart.com/>

A free database designer for developers and analysts. (s/f). Dbdiagram.io. Recuperado el 4 de diciembre de 2025, de <https://dbdiagram.io/home>

Archivex. (s/f). ComparaSoftware. Recuperado el 8 de diciembre de 2025, de <https://www.comparasoftware.com/archivex-clinical>

CRM de gestión de pacientes para clínicas y hospitales - TuoTempo. (s/f). Tuotempo.es. Recuperado el 8 de diciembre de 2025, de <https://www.tuotempo.es/>

Marqués, F. L. (2024, febrero 25). Cómo mejorar la Experiencia del Usuario (UX) en un software de clínica. *Clinic-cloud.com.* <https://clinic-cloud.com/blog/como-mejorar-la-experiencia-del-usuario-ux-en-un-software-de-clinica>

Medesk - Software Médico para Clínicas - Consultorios - IPS. (s/f). Medesk.net. Recuperado el 8 de diciembre de 2025, de <https://www.medesk.net/es/>

Mejore la Eficiencia de su Clínica con Medesk: Software de Historias Clínicas Gratuito en 2025. (s/f). Medesk.net. Recuperado el 8 de diciembre de 2025, de <https://www.medesk.net/es/blog/software-para-historias-clinicas-gratis/>

PaaS, IaaS y SaaS: ¿en qué se diferencian? (s/f). Google Cloud. Recuperado el 8 de diciembre de 2025, de <https://cloud.google.com/learn/paas-vs-iaas-vs-saas?hl=es>

Seguridad - Medesk. (s/f). Medesk.net. Recuperado el 8 de diciembre de 2025, de <https://www.medesk.net/es/security/>

Miro. (s/f). Miro.com. Recuperado el 8 de diciembre de 2025, de <https://miro.com/app/board/uXjVJgoPSfY=/>

Cómo crear una contraseña de aplicación en Gmail. (s/f). MegaCity20. Recuperado el 10 de diciembre de 2025, de <https://megacity20.com/support/106>



ANEXOS

Anexo I – Script SQL de la base de datos

```
-- =====
-- BASE DE DATOS MEDIGEST
-- Script completo con procedimientos y datos
-- =====

CREATE DATABASE IF NOT EXISTS medigest;
USE medigest;

-- =====
-- CREACIÓN DE TABLAS
-- =====

-- USUARIO
CREATE TABLE Usuario (
    id_usuario INT AUTO_INCREMENT PRIMARY KEY,
    login VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(100) NOT NULL,
    rol ENUM('repcionista','medico','admin') NOT NULL
);

-- RECEPCIONISTA
CREATE TABLE Recepcionista (
    id_repcionista INT AUTO_INCREMENT PRIMARY KEY,
    id_usuario INT NOT NULL,
    nombre VARCHAR(100),
    apellidos VARCHAR(150),
    telefono VARCHAR(20) UNIQUE,
    email VARCHAR(100),
    FOREIGN KEY (id_usuario) REFERENCES Usuario(id_usuario)
        ON DELETE CASCADE
);

-- ESPECIALIDAD
CREATE TABLE Especialidad (
    id_especialidad INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion TEXT,
    precio DECIMAL(10,2) NOT NULL DEFAULT 0
);

-- MÉDICO
CREATE TABLE Medico (
    id_medico INT AUTO_INCREMENT PRIMARY KEY,
    id_usuario INT NOT NULL,
    id_especialidad INT NOT NULL,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(150),
    num_colegiado VARCHAR(50) UNIQUE,
    correo_corporativo VARCHAR(100),
    FOREIGN KEY (id_usuario) REFERENCES Usuario(id_usuario)
        ON DELETE CASCADE,
    FOREIGN KEY (id_especialidad) REFERENCES Especialidad(id_especialidad)
        ON DELETE RESTRICT
);

-- PACIENTE
```

```

CREATE TABLE Paciente (
    id_paciente INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellidos VARCHAR(150),
    dni VARCHAR(15) UNIQUE,
    cipa VARCHAR(30),
    num_historia_clinica VARCHAR(30),
    num_seguridad_social VARCHAR(30),
    fecha_nacimiento DATE,
    fecha_ingreso DATE,
    correo VARCHAR(30)
);

-- CITA
CREATE TABLE Cita (
    id_cita INT AUTO_INCREMENT PRIMARY KEY,
    id_paciente INT NOT NULL,
    id_medico INT,
    id_recepcionista INT,
    fecha DATE NOT NULL,
    hora TIME NOT NULL,
    estado ENUM('pendiente','confirmada','cancelada','realizada') DEFAULT
    'pendiente',
    observaciones TEXT,
    FOREIGN KEY (id_paciente) REFERENCES Paciente(id_paciente),
    FOREIGN KEY (id_medico) REFERENCES Medico(id_medico),
    FOREIGN KEY (id_recepcionista) REFERENCES Recepcionista(id_recepcionista)
);

-- INFORME MÉDICO
CREATE TABLE Informe_Medico (
    id_informe INT AUTO_INCREMENT PRIMARY KEY,
    id_paciente INT,
    id_medico INT,
    id_cita INT,
    fecha_emision DATE NOT NULL,
    motivo_consulta VARCHAR(255),
    diagnostico TEXT,
    tratamiento TEXT,
    observaciones TEXT,
    FOREIGN KEY (id_paciente) REFERENCES Paciente(id_paciente),
    FOREIGN KEY (id_medico) REFERENCES Medico(id_medico),
    FOREIGN KEY (id_cita) REFERENCES Cita(id_cita)
        ON DELETE CASCADE
);

-- =====
-- PROCEDIMIENTO ALMACENADO
-- =====

DELIMITER //

CREATE PROCEDURE InsertarUsuario(
    IN p_login VARCHAR(50),
    IN p_password VARCHAR(100),
    IN p_rol ENUM('recepcionista','medico','admin')
)
BEGIN
    -- Encripta la contraseña usando SHA2 con 256 bits
    -- SHA2-256 es un algoritmo robusto y ampliamente utilizado
    INSERT INTO Usuario (login, password, rol)
    VALUES (p_login, SHA2(p_password, 256), p_rol);

```



MediGest Pro

END //

DELIMITER ;

-- =====
-- INSERCIÓN DE DATOS
-- =====

-- ESPECIALIDADES (con descripciones detalladas)

INSERT INTO Especialidad (nombre, descripcion, precio) VALUES
(‘Cardiología’, ‘La cardiología es la especialidad médica dedicada al estudio, diagnóstico y tratamiento de las enfermedades del corazón y del sistema circulatorio. Los cardiólogos realizan pruebas diagnósticas como electrocardiogramas, ecocardiogramas y pruebas de esfuerzo, además de tratar condiciones como hipertensión arterial, insuficiencia cardíaca, arritmias, enfermedad coronaria y valvulopatías. También se encargan de la prevención cardiovascular mediante el control de factores de riesgo.’,90.00),
(‘Traumatología’, ‘La traumatología y cirugía ortopédica se especializa en el diagnóstico y tratamiento de lesiones y enfermedades del sistema musculoesquelético, incluyendo huesos, articulaciones, ligamentos, tendones y músculos. Trata fracturas, luxaciones, lesiones deportivas, artrosis, hernias discales, escoliosis y malformaciones congénitas. Los traumatólogos realizan tanto tratamientos conservadores como intervenciones quirúrgicas para restaurar la función y movilidad del paciente.’,70.00),
(‘Pediatría’, ‘La pediatría es la especialidad médica que se ocupa de la salud integral de niños y adolescentes desde el nacimiento hasta los 18 años. Los pediatras realizan seguimiento del desarrollo físico, cognitivo y emocional, administran vacunas según el calendario establecido, diagnostican y tratan enfermedades infantiles agudas y crónicas, orientan sobre nutrición y prevención, y detectan precozmente alteraciones del crecimiento o desarrollo.’,60.00),

...

;

-- USUARIOS (usando el procedimiento)

-- ADMINS

CALL InsertarUsuario('45678901C', 'Admin123!', 'admin');
CALL InsertarUsuario('23456789A', 'Admin2024\$', 'admin');

-- Recepcionistas

CALL InsertarUsuario('12345678Z', 'Repcion2024!', 'repcionista');
INSERT INTO Recepcionista (id_usuario, nombre, apellidos, telefono, email)
VALUES (3, 'María', 'García López', '915234567', 'maria.garcia@medigest.es');

CALL InsertarUsuario('87523388Q', 'Repcion2024\$', 'repcionista');
INSERT INTO Recepcionista (id_usuario, nombre, apellidos, telefono, email)
VALUES (4, 'Carlos', 'Fernández Ruiz', '914567890',
'carlos.fernandez@medigest.es');

CALL InsertarUsuario('34567890B', 'Recep2024#', 'repcionista');
INSERT INTO Recepcionista (id_usuario, nombre, apellidos, telefono, email)
VALUES (5, 'Laura', 'Martínez Sánchez', '913456789',
'laura.martinez@medigest.es');

CALL InsertarUsuario('57142851H', 'Repcionista123!', 'repcionista');
INSERT INTO Recepcionista (id_usuario, nombre, apellidos, telefono, email)
VALUES (6, 'Antonio', 'López Jiménez', '912345678', 'antonio.lopez@medigest.es');

-- USUARIOS Y MÉDICOS (usando el procedimiento)

CALL InsertarUsuario('56789012D', 'Cardio2024!', 'medico');

```

INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (7, 1, 'Ana', 'Rodríguez Pérez', '282801234',
'ana.rodriguez@medigest.es');

CALL InsertarUsuario('67890123E', 'Trauma2024$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (8, 2, 'Javier', 'González Moreno', '282802345',
'javier.gonzalez@medigest.es');

CALL InsertarUsuario('78901234F', 'Pediatra24$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (9, 3, 'Elena', 'Sánchez Martín', '282803456',
'elena.sanchez@medigest.es');

CALL InsertarUsuario('89012345G', 'Dermato24!', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (10, 4, 'Miguel', 'Díaz Hernández', '282804567',
'miguel.diaz@medigest.es');

CALL InsertarUsuario('90123456H', 'Oftalmo24$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (11, 5, 'Isabel', 'Ruiz García', '282805678',
'isabel.ruiz@medigest.es');

CALL InsertarUsuario('01234567I', 'Gineco2024$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (12, 6, 'Carmen', 'Moreno López', '282806789',
'carmen.moreno@medigest.es');

CALL InsertarUsuario('11234567J', 'Neuro2024!', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (13, 7, 'Francisco', 'Jiménez Fernández', '282807890',
'francisco.jimenez@medigest.es');

CALL InsertarUsuario('21234567K', 'MedGen24$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (14, 8, 'Rosa', 'Álvarez Romero', '282808901',
'rosa.alvarez@medigest.es');

CALL InsertarUsuario('31234567L', 'Psiquiat24$', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (15, 9, 'Pedro', 'Torres Navarro', '282809012',
'pedro.torres@medigest.es');

CALL InsertarUsuario('41234567M', 'Endocri24!', 'medico');
INSERT INTO Medico (id_usuario, id_especialidad, nombre, apellidos,
num_colegiado, correo_corporativo)
VALUES (16, 10, 'Lucía', 'Ramírez Gil', '282810123',
'lucia.ramirez@medigest.es');

-- PACIENTES (200 pacientes con DNI's válidos españoles)
INSERT INTO Paciente (nombre, apellidos, dni, cipa, numHistoriaClinica,
numSeguridadSocial, fechaNacimiento, fechaIngreso, correo) VALUES

```



MediGest Pro

```
('Juan', 'Pérez García', '12345678Z', 'CIPA001', 'HC001', '281234567890', '1985-03-15', '2025-01-02', 'pacientesmedigest@gmail.com'),  
('María', 'López Martínez', '23456789A', 'CIPA002', 'HC002', '281234567891', '1990-07-22', '2025-01-03', 'pacientesmedigest@gmail.com'),  
('Carlos', 'González Rodríguez', '34567890B', 'CIPA003', 'HC003', '281234567892', '1978-11-30', '2025-01-04', 'pacientesmedigest@gmail.com'),  
('Ana', 'Fernández Sánchez', '45678901C', 'CIPA004', 'HC004', '281234567893', '1995-01-10', '2025-01-05', 'pacientesmedigest@gmail.com'),  
('Luis', 'Martínez López', '56789012D', 'CIPA005', 'HC005', '281234567894', '1982-06-18', '2025-01-06', 'pacientesmedigest@gmail.com'),  
('Elena', 'Sánchez Pérez', '67890123E', 'CIPA006', 'HC006', '281234567895', '1988-09-25', '2025-01-07', 'pacientesmedigest@gmail.com'),  
('David', 'Rodríguez García', '78901234F', 'CIPA007', 'HC007', '281234567896', '1975-12-05', '2025-01-08', 'pacientesmedigest@gmail.com'),  
('Laura', 'García Martínez', '89012345G', 'CIPA008', 'HC008', '281234567897', '1992-04-14', '2025-01-09', 'pacientesmedigest@gmail.com'),  
...  
;  
  
-- CITAS (500 citas con horarios realistas de lunes a viernes)  
INSERT INTO Cita (id_paciente, id_medico, id_recepcionista, fecha, hora, estado, observaciones) VALUES  
-- Enero 2025  
(1, 1, 1, '2025-01-02', '09:00:00', 'realizada', 'Control rutinario de tensión arterial'),  
(2, 2, 1, '2025-01-02', '09:30:00', 'realizada', 'Revisión de fractura de muñeca'),  
(3, 3, 2, '2025-01-02', '10:00:00', 'realizada', 'Vacunación infantil'),  
(4, 4, 2, '2025-01-02', '10:30:00', 'realizada', 'Dermatitis atópica'),  
(5, 5, 1, '2025-01-02', '11:00:00', 'realizada', 'Revisión postoperatoria de cataratas'),  
(6, 6, 1, '2025-01-02', '11:30:00', 'realizada', 'Consulta ginecológica anual'),  
(7, 7, 2, '2025-01-02', '12:00:00', 'realizada', 'Cefaleas recurrentes'),  
(8, 8, 2, '2025-01-02', '16:00:00', 'realizada', 'Chequeo general'),  
(9, 9, 1, '2025-01-02', '16:30:00', 'realizada', 'Evaluación psiquiátrica inicial'),  
(10, 10, 1, '2025-01-02', '17:00:00', 'realizada', 'Control de diabetes tipo 2'),  
(11, 1, 2, '2025-01-03', '09:00:00', 'realizada', 'Seguimiento de hipertensión'),  
(12, 2, 2, '2025-01-03', '09:30:00', 'realizada', 'Dolor lumbar crónico'),  
(13, 3, 1, '2025-01-03', '10:00:00', 'realizada', 'Control de crecimiento'),  
(14, 4, 1, '2025-01-03', '10:30:00', 'realizada', 'Acné juvenil'),  
(15, 5, 2, '2025-01-03', '11:00:00', 'realizada', 'Revisión de vista'),  
...  
;  
  
-- INFORMES MÉDICOS (Muestra de informes para casos realizados)  
INSERT INTO Informe_Medico (id_paciente, id_medico,id_cita, fecha_emision, motivo_consulta, diagnostico, tratamiento, observaciones) VALUES  
(1, 1, 1, '2025-01-02', 'Control rutinario de tensión arterial', 'Hipertensión arterial grado 1 controlada', 'Continuar con Enalapril 10mg/día. Dieta hiposódica. Control en 3 meses.', 'Paciente adherente al tratamiento. Tensión arterial 135/85 mmHg.'),  
(2, 2, 2, '2025-01-02', 'Revisión de fractura de muñeca', 'Consolidación ósea satisfactoria de fractura de Colles', 'Retirada de inmovilización. Fisioterapia rehabilitadora 3 sesiones semanales durante 6 semanas.', 'Buena evolución. Movilidad recuperada en un 70%. Continuar ejercicios en domicilio.'),  
(3, 3, 3, '2025-01-02', 'Vacunación infantil', 'Desarrollo ponderoestatural adecuado para la edad', 'Administración de vacuna hexavalente (DTaP-IPV-Hib-HepB) y antineumocócica conjugada. Próxima dosis en 2 meses.', 'Niño sano. Peso 8.2kg, talla 68cm. Percentil 50. Alimentación complementaria iniciada correctamente.'),  
(4, 4, 4, '2025-01-02', 'Dermatitis atópica', 'Dermatitis atópica moderada en fase de brote. Xerosis cutánea generalizada.', 'Emolientes tras baño diario.
```

Tacrolimus pomada 0.03% dos veces al día en lesiones activas. Antihistamínico oral si prurito nocturno.', 'Evitar irritantes. Uso de ropa de algodón. Reevaluación en 15 días.'),

```
...
;

-- =====
-- FIN DEL SCRIPT
-- =====
```

Anexo II – Código de validación del registro de pacientes

```
private void btnGuardar_Click(object sender, RoutedEventArgs e)
{
    if (txtApellidos.Text == "" || txtCIPA.Text == "" || txtDNI.Text == "" ||
    txtHistoria.Text == "" || txtNombre.Text == "" || txtSeguridadSocial.Text == "")
    {
        MessageBox.Show("Rellene todos los Campos", "Aviso", MessageBoxButton.OK,
        MessageBoxIcon.Warning);
        return;
    }

    if (!ValidarDNI(txtDNI.Text))
    {
        MessageBox.Show("DNI Invalido");
        return;
    }

    if (dpFechaNacimiento.SelectedDate > DateTime.Now)
    {
        MessageBox.Show("No puede elegir una fecha de nacimiento posterior a la
        de hoy.", "Aviso", MessageBoxButton.OK, MessageBoxIcon.Warning);
        return;
    }

    try
    {
        using (var db = new MediGestContext())
        {
            var pacienteDB = db.Paciente.FirstOrDefault(c => c.Id_paciente ==
pacienteActual.Id_paciente);

            if (pacienteDB == null)
            {
                MessageBox.Show("No se encontró el paciente en la base de
                datos.", "Error", MessageBoxButton.OK, MessageBoxIcon.Error);
                return;
            }

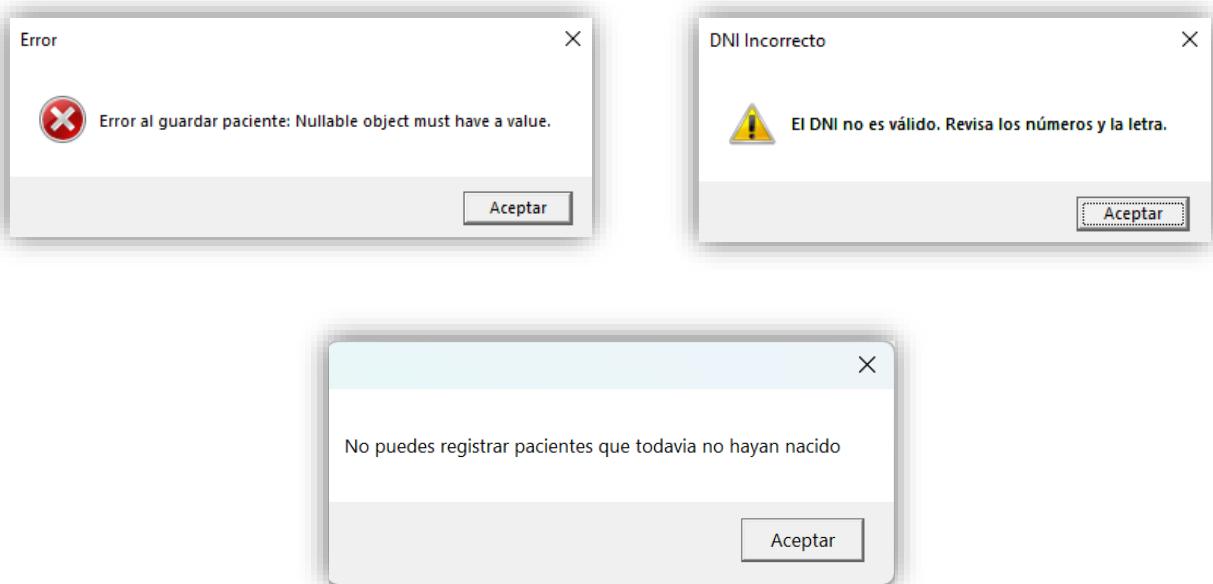
            // Actualizamos campos modificables
            pacienteDB.Nombre = txtNombre.Text;
            pacienteDB.Apellidos = txtApellidos.Text;
            pacienteDB.Dni = txtDNI.Text;
            pacienteDB.Cipa = txtCIPA.Text;
            pacienteDB.Num_historia_clinica = txtHistoria.Text;
            pacienteDB.Num_seguridad_social = txtSeguridadSocial.Text;
            pacienteDB.Fecha_nacimiento = dpFechaNacimiento.SelectedDate.Value;
            db.SaveChanges();
            MessageBox.Show("✓ Paciente actualizado correctamente.", "Éxito",
            MessageBoxButton.OK, MessageBoxIcon.Information);
            this.Close();
        }
    }
    catch (Exception ex)
```



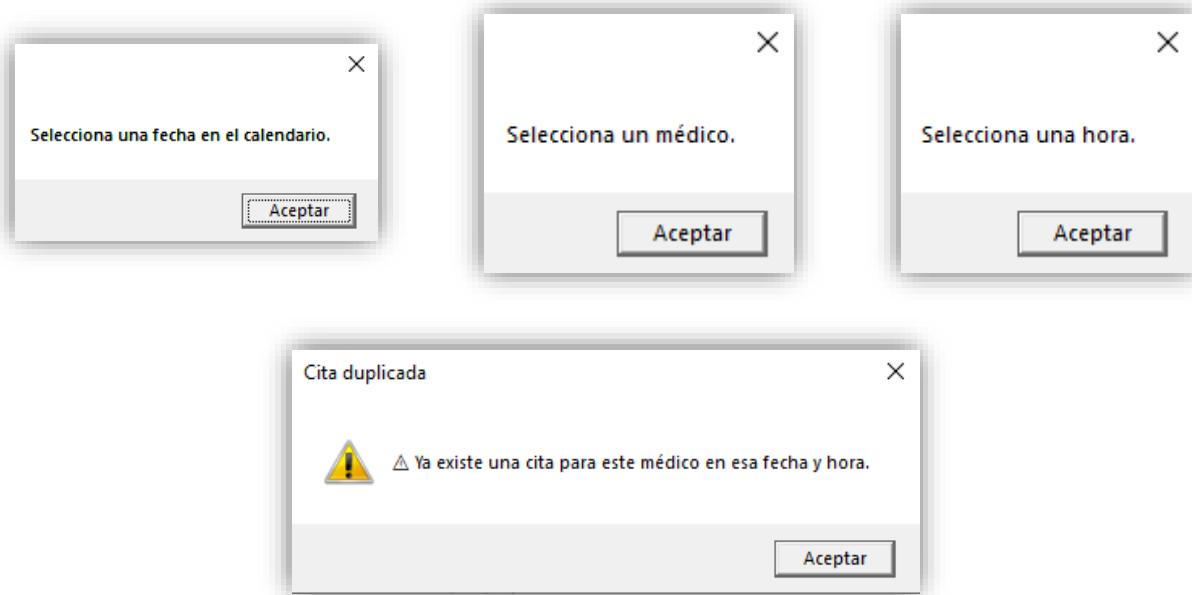
```
{  
    MessageBox.Show($"Error al guardar los  
cambios: \n{ex.InnerException?.Message ?? ex.Message}", "Error",  
    MessageBoxButtons.OK, MessageBoxIcon.Error);  
}  
  
}  
  
private bool ValidarDNI(string dni)  
{  
    if (string.IsNullOrWhiteSpace(dni))  
        return false;  
  
    dni = dni.ToUpper();  
  
    if (dni.Length != 9)  
        return false;  
  
    string numeros = dni.Substring(0, 8);  
    char letra = dni[8];  
  
    if (!int.TryParse(numeros, out int num))  
        return false;  
  
    string letrasValidas = "TRWAGMYFPDXBNJZSQVHLCKE";  
    int indice = num % 23;  
    char letraCorrecta = letrasValidas[indice];  
  
    return letra == letraCorrecta;  
}
```

Anexo III – Mensajes informativos

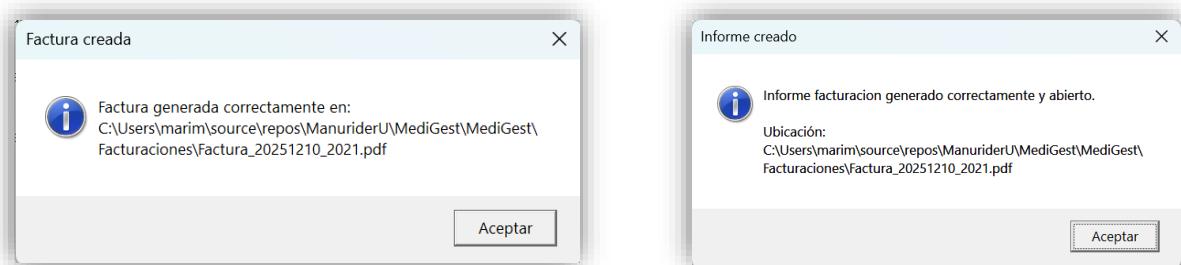
III.1. Registrar nuevo paciente



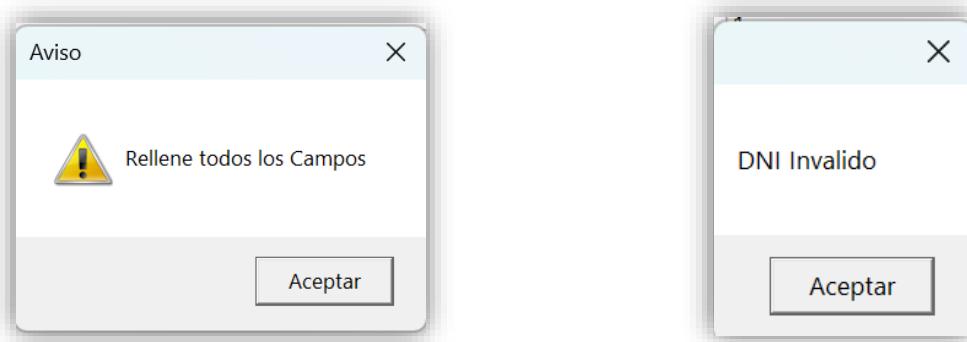
III.2. Agendar cita

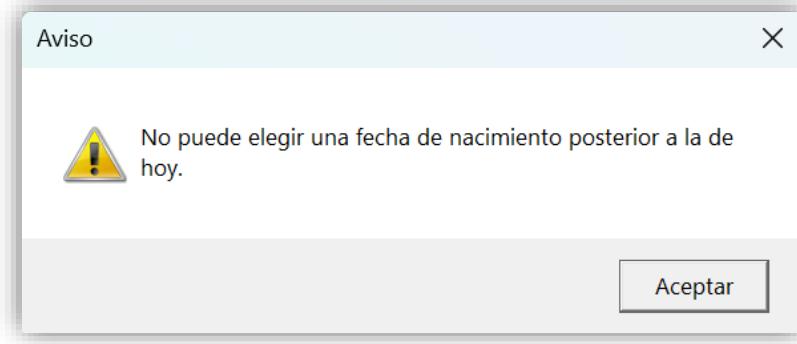


III.3. Generar factura

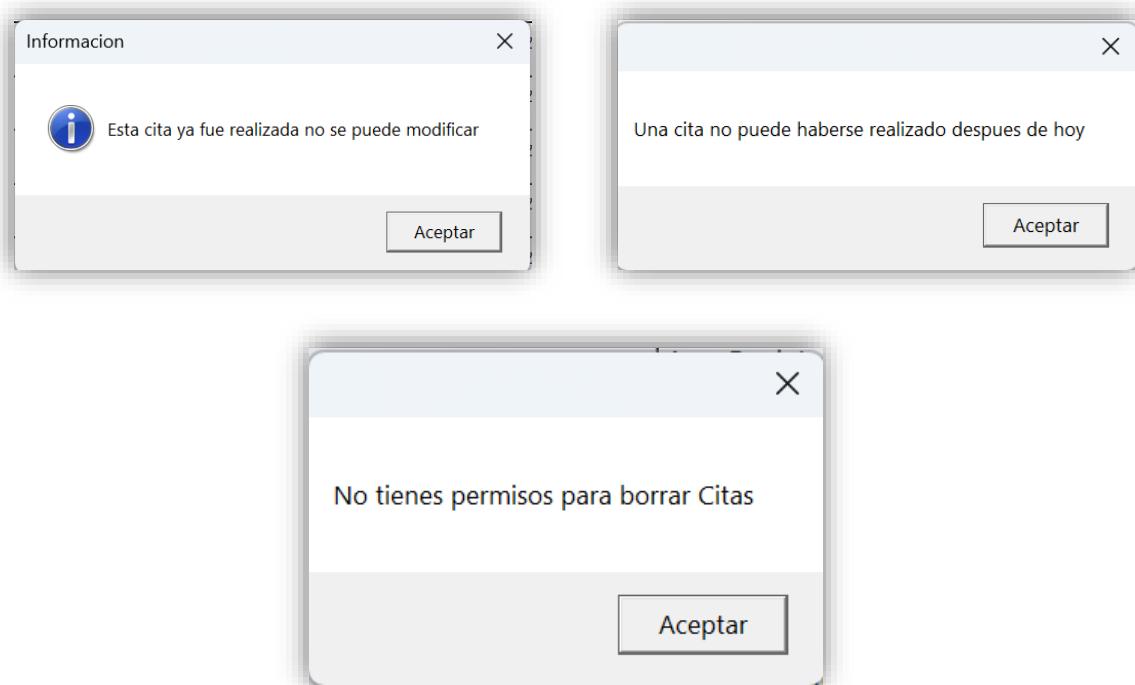


III.4. Editar paciente





III.5. Editar y eliminar cita



Anexo IV – Código de generar factura

```
namespace MediGest
{
    public class FacturaGenerator
    {
        private readonly MediGestContext db = new MediGestContext();

        public void GenerarFactura(int idPaciente, DateTime fechaInicio, DateTime
fechaFin, string rutaDestino)
        {
            using (var db = new MediGestContext())
            {
                // Obtener las citas del paciente entre las fechas
                String estado = "realizada";
                var citas = (from c in db.Cita
                            join m in db.Medico on c.Id_medico equals
m.Id_medico
```

```

equals e.Id_especialidad
&& c.Estado == estado
join e in db.Especialidad on m.Id_especialidad
equals e.Id_especialidad
where c.Id_paciente == idPaciente &&
c.Fecha >= fechaInicio && c.Fecha <= fechaFin
select new
{
    c.Fecha,
    Medico = m.Nombre + " " + m.Apellidos,
    Especialidad = e.Nombre,
    e.Precio,
    c.Observaciones
}).ToList();

if (!citas.Any())
    throw new Exception("No hay citas realizadas con el paciente
en ese rango de fechas.");

// Obtener datos del paciente
var paciente = db.Paciente.FirstOrDefault(p => p.Id_paciente ==
idPaciente);
if (paciente == null)
    throw new Exception("No se encontró el paciente.");

// Crear pdf del informe
Document doc = new Document(PageSize.A4, 50, 50, 50, 50);

using (FileStream fs = new FileStream(rutaDestino,
 FileMode.Create))
{
    PdfWriter.GetInstance(doc, fs);
    doc.Open();

    // ===== LOGO =====
    string projectPath =
System.IO.Path.GetFullPath(System.IO.Path.Combine(AppDomain.CurrentDomain.BaseDir
ecitory, @"..\..\.."));
    string rutaLogo = System.IO.Path.Combine(projectPath,
"Resources", "logo.jpg"); // Ajusta tu ruta
    if (File.Exists(rutaLogo))
    {
        iTextSharp.text.Image logo =
iTextSharp.text.Image.GetInstance(rutaLogo);

        // Puedes ajustar el tamaño
        logo.ScaleToFit(120f, 120f);

        // Alinear a la izquierda o centro:
        logo.Alignment = Element.ALIGN_LEFT; // o
Element.ALIGN_CENTER

        doc.Add(logo);
        doc.Add(new Paragraph("\n"));
    }
    // =====

    // Fuentes
    var tituloFont =
FontFactory.GetFont(FontFactory.HELVETICA_BOLD, 18, BaseColor.BLACK);
    var normalFont = FontFactory.GetFont(FontFactory.HELVETICA,
12, BaseColor.BLACK);
}

```



MediGest Pro

```
var boldFont =
FontFactory.GetFont(FontFactory.HELVETICA_BOLD, 12, BaseColor.BLACK);

    // Título principal
    Paragraph titulo = new Paragraph("FACTURA MÉDICA",
    tituloFont);
    titulo.Alignment = Element.ALIGN_CENTER;
    doc.Add(titulo);

    doc.Add(new Paragraph("\n"));
    doc.Add(new LineSeparator(1f, 100f, BaseColor.GRAY,
    Element.ALIGN_CENTER, -2));

    // Datos del paciente
    doc.Add(new Paragraph($"Paciente: {paciente.Nombre}
{paciente.Apellidos}", normalFont));
    doc.Add(new Paragraph($"DNI: {paciente.Dni}", normalFont));
    doc.Add(new Paragraph($"Periodo: {fechaInicio:dd/MM/yyyy} -
{fechaFin:dd/MM/yyyy}", normalFont));
    doc.Add(new Paragraph("\n"));

    // Título de sección
    Paragraph detalleTitulo = new Paragraph("Detalle de citas:",
    boldFont);
    detalleTitulo.SpacingBefore = 10;
    detalleTitulo.SpacingAfter = 5;
    doc.Add(detalleTitulo);

    // Crear tabla
    PdfPTable tabla = new PdfPTable(5);
    tabla.WidthPercentage = 100;
    tabla.SetWidths(new float[] { 25f, 35f, 25f, 15f, 35f });

    // Encabezados
    tabla.addCell(new PdfPCell(new Phrase("Fecha", boldFont)) {
BackgroundColor = BaseColor.LIGHT_GRAY });
    tabla.addCell(new PdfPCell(new Phrase("Médico", boldFont)) {
BackgroundColor = BaseColor.LIGHT_GRAY });
    tabla.addCell(new PdfPCell(new Phrase("Especialidad",
boldFont)) { BackgroundColor = BaseColor.LIGHT_GRAY });
    tabla.addCell(new PdfPCell(new Phrase("Precio", boldFont)) {
BackgroundColor = BaseColor.LIGHT_GRAY });
    tabla.addCell(new PdfPCell(new Phrase("Observaciones",
boldFont)) { BackgroundColor = BaseColor.LIGHT_GRAY });

    decimal total = 0;
    foreach (var cita in citas)
    {
        tabla.addCell(new
Phrase(cita.Fecha.ToString("dd/MM/yyyy"), normalFont));
        tabla.addCell(new Phrase(cita.Medico, normalFont));
        tabla.addCell(new Phrase(cita.Especialidad, normalFont));
        tabla.addCell(new Phrase($"{cita.Precio:C}",
normalFont));
        tabla.addCell(new Phrase(cita.Observaciones, normalFont));
        total += cita.Precio;
    }

    doc.Add(tabla);
    doc.Add(new Paragraph("\n"));

    // Monto
```

```

        Paragraph totalTxt = new Paragraph($"TOTAL A PAGAR:  

{total:C}", boldFont);
        totalTxt.Alignment = Element.ALIGN_RIGHT;
        doc.Add(totalTxt);

        doc.Add(new Paragraph("\n"));
        doc.Add(new LineSeparator(1f, 100f, BaseColor.GRAY,
Element.ALIGN_CENTER, -2));

        // Fecha de emisión
        Paragraph fechaEmision = new Paragraph($"Fecha de emisión:  

{DateTime.Now:dd/MM/yyyy HH:mm}", normalFont);
        fechaEmision.Alignment = Element.ALIGN_RIGHT;
        doc.Add(fechaEmision);

        // Cierre
        doc.Close();

        // ✅ Confirmación al usuario
        MessageBox.Show($"Factura generada correctamente  

en:\n{rutaDestino}",  

"Factura creada", MessageBoxButtons.OK,  

MessageBoxImage.Information);
    }

    try
    {
        ProcessStartInfo psi = new ProcessStartInfo
        {
            FileName = rutaDestino,
            UseShellExecute = true
        };
        Process.Start(psi);

        MessageBox.Show(
            $"Informe facturación generado correctamente y  

abierto.\n\nUbicación: {rutaDestino}",
            "Informe creado",
            MessageBoxButtons.OK,
            MessageBoxIcon.Information
        );
    }
    catch (Exception ex)
    {
        MessageBox.Show(
            $"El informe se generó correctamente  

en:\n{rutaDestino}\n\nPero no se pudo abrir automáticamente:\n{ex.Message}",
            "Informe creado",
            MessageBoxButtons.OK,
            MessageBoxIcon.Warning
        );
    }
}
}

```



Anexo V – Correo

V.1. Envío

```
using System.Net;
using System.Net.Mail;
using System.IO;

namespace MediGest.Servicios
{
    public class EmailService
    {
        private readonly string smtpUser;
        private readonly string smtpPass;

        public EmailService(string user)
        {
            smtpUser = user; // correo del médico
            smtpPass = "bydh ghmt ufrw lbmc"; // contraseña de aplicación de
Gmail
        }

        public void EnviarCorreo(string destinatario, string asunto, string
htmlBody, string rutaLogo)
        {
            MailMessage mail = new MailMessage();
            mail.From = new MailAddress(smtpUser);
            mail.To.Add(destinatario);
            mail.Subject = asunto;

            // *** IMPORTANTE: NO USAR mail.Body ni IsBodyHtml ***
            mail.Body = "";
            mail.IsBodyHtml = false;

            // Crear vista HTML
            AlternateView htmlView =
AlternateView.CreateAlternateViewFromString(htmlBody, null, "text/html");

            // Agregar logo como recurso CID
            if (File.Exists(rutaLogo))
            {
                LinkedResource logo = new LinkedResource(rutaLogo, "image/jpeg");
                logo.ContentId = "logo_medicus";
                logo.ContentType.MediaType = "image/jpeg";
                logo.TransferEncoding = System.Net.Mime.TransferEncoding.Base64;

                htmlView.LinkedResources.Add(logo);
            }

            mail.AlternateViews.Add(htmlView);

            using (SmtpClient smtp = new SmtpClient("smtp.gmail.com", 587))
            {
                smtp.Credentials = new NetworkCredential(smtpUser, smtpPass);
                smtp.EnableSsl = true;
                smtp.Send(mail);
            }
        }

        public string CargarPlantilla(string ruta)
        {
            return File.ReadAllText(ruta);
```

```

        }
    }
}
```

V.2. Estilo de plantilla

```

<style>
body {
    font-family: Arial, sans-serif;
    background-color: #f8fafc;
    margin: 0;
    padding: 0;
}

.container {
    background-color: #ffffff;
    width: 90%;
    max-width: 600px;
    margin: 20px auto;
    padding: 20px 30px;
    border-radius: 12px;
    box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.header {
    background-color: #0ea5e9;
    color: white;
    padding: 12px 20px;
    border-radius: 12px 12px 0 0;
    text-align: center;
    font-size: 20px;
    font-weight: bold;
}

.logo {
    text-align: center;
    margin-bottom: 10px;
}

.content {
    padding: 20px 0;
    font-size: 14px;
    color: #334155;
    line-height: 1.6;
}

.footer {
    margin-top: 20px;
    font-size: 12px;
    color: #64748b;
    text-align: center;
}

.highlight {
    color: #10b981;
    font-weight: bold;
}
</style>
```



Anexo VI – Código de generar pdf de informes médicos

```
namespace MediGest
{
    /// <summary>
    /// Lógica de interacción para GenerarFactura.xaml
    /// </summary>
    public partial class GenerarFactura : Window
    {
        public GenerarFactura()
        {
            InitializeComponent();
            CargarPacientes();
        }

        private void CargarPacientes()
        {
            using (var db = new MediGestContext())
            {
                var pacientes = db.Paciente
                    .Select(p => new { p.Id_paciente, NombreCompleto = p.Nombre +
" " + p.Apellidos })
                    .ToList();

                CmbPaciente.ItemsSource = pacientes;
                CmbPaciente.DisplayMemberPath = "NombreCompleto";
                CmbPaciente.SelectedValuePath = "Id_paciente";
            }
        }

        private void BtnGenerarFactura_Click(object sender, RoutedEventArgs e)
        {
            if (CmbPaciente.SelectedItem == null)
            {
                MessageBox.Show("Selecciona un paciente.", "Aviso",
MessageBoxButton.OK, MessageBoxIcon.Warning);
                return;
            }

            if (!FechaInicioPicker.SelectedDate.HasValue ||
!FechaFinPicker.SelectedDate.HasValue)
            {
                MessageBox.Show("Selecciona las fechas de inicio y fin.",
"Aviso", MessageBoxButton.OK, MessageBoxIcon.Warning);
                return;
            }

            DateTime inicio = FechaInicioPicker.SelectedDate.Value;
            DateTime fin = FechaFinPicker.SelectedDate.Value;

            if (inicio > fin)
            {
                MessageBox.Show("La fecha de inicio no puede ser posterior a la
fecha de fin.", "Error", MessageBoxButton.OK, MessageBoxIcon.Error);
                return;
            }

            int idPaciente = (int)CmbPaciente.SelectedValue;

            try
            {
                // Crear la ruta a la carpeta dentro del proyecto

```

```
        string projectPath =
System.IO.Path.GetFullPath(System.IO.Path.Combine(AppDomain.CurrentDomain.BaseDir
ecitory, @"..\..\.."));
        string carpetaFacturas = System.IO.Path.Combine(projectPath,
"Facturaciones");

        // Crear la carpeta si no existe
if (!Directory.Exists(carpetaFacturas))
    Directory.CreateDirectory(carpetaFacturas);

        // Nombre del archivo PDF
string nombreArchivo =
$"Factura_{DateTime.Now:yyyyMMdd_HHmm}.pdf";

        // Ruta completa final
string ruta = System.IO.Path.Combine(carpetaFacturas,
nombreArchivo);

        FacturaGenerator gen = new FacturaGenerator();
gen.GenerarFactura(idPaciente, inicio, fin, ruta);
    }
    catch (Exception ex)
    {
        MessageBox.Show($"Error al generar la factura:\n{ex.Message}",
                    "Error", MessageBoxButtons.OK,
MessageBoxImage.Error);
    }
}
}
```



MediGest Pro