APLICACIÓN WEB PARA LA GESTIÓN DE UNA CLÍNICA DENTAL

WEB APPLICATION FOR THE MANAGEMENT OF A DENTAL CLINIC

TRABAJO FIN DE GRADO
CURSO 2022-2023

AUTOR
BELÉN FERNÁNDEZ-RICO SÁNCHEZ

DIRECTOR
RAMÓN GONZÁLEZ DEL CAMPO RODRÍGUEZ BARBERO

DOBLE GRADO EN ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS E INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

APLICACIÓN WEB PARA LA GESTIÓN DE UNA CLÍNICA DENTAL

WEB APPLICATION FOR THE MANAGEMENT OF A DENTAL CLINIC

TRABAJO DE FIN DE GRADO EN INGENIERÍA INFORMÁTICA

AUTOR
BELÉN FERNÁNDEZ-RICO SÁNCHEZ

DIRECTOR
RAMÓN GONZÁLEZ DEL CAMPO RODRÍGUEZ BARBERO

CONVOCATORIA: JUNIO 2023

DOBLE GRADO EN ADMINISTRACIÓN Y DIRECCIÓN DE EMPRESAS E INGENIERÍA INFORMÁTICA
FACULTAD DE INFORMÁTICA
UNIVERSIDAD COMPLUTENSE DE MADRID

29 DE MAYO DE 2023

RESUMEN

Aplicación web para el desarrollo de una clínica dental

En este trabajo se ha desarrollado una aplicación web para facilitar el proceso de gestión de una clínica dental, mediante una serie de funcionalidades que ayudan con la administración del negocio.

La aplicación web permite la gestión de los distintos pacientes que acuden a la clínica y de los distintos colaboradores dentistas que trabajan en ella. También facilita la gestión del calendario de citas, que permite coordinar el horario de citas de los colaboradores y pacientes.

Por otro lado, se ha implementado un sistema de gestión del inventario, que regula la entrada y salida de materiales que se consumen con cada tratamiento. Asimismo, se ha incluido un módulo de contabilidad, que permite llevar un control de los distintos pagos realizados por los clientes y las facturas generadas para los dentistas por los servicios prestados.

Para llevar a cabo todas las tareas de gestión, la aplicación cuenta con un perfil de Administrador, que tiene acceso a todas las funcionalidades previamente mencionadas. Además, se han creado otros roles de usuarios específicos para pacientes y colaboradores, quienes pueden acceder a su propio perfil individual y consultar su información personal. Con esta personalización de la aplicación se pretende mejorar la calidad de la atención para pacientes y trabajadores, ofreciendo una experiencia adaptada a las necesidades de cada usuario.

La aplicación web se ha desarrollado utilizando las tecnologías React y Node.js, para la creación del front-end y back-end, respectivamente, que permiten un entorno de desarrollo moderno y eficiente.

Palabras clave

Aplicación web, gestión, clínica dental, personalización, roles de usuario, pacientes, colaboradores, facturación, React, Node js.

ABSTRACT

Web application for the management of a dental clinic

In this work, a web application has been developed to facilitate the management process of a dental clinic, through a series of functionalities that help with the administration of the business.

The web application allows the management of the different patients who come to the clinic and the different dental collaborators who work there. It also facilitates the management of the appointment calendar, which makes it possible to coordinate the appointment schedule of collaborators and patients.

On the other hand, an inventory management system has been implemented, which regulates the input and output of materials that are consumed with each treatment. An accounting module has also been included to keep track of the various payments made by customers and the invoices generated for dentists for services rendered.

To carry out all management tasks, the application has an Administrator profile, which has access to all the previously mentioned functionalities. In addition, other specific user roles have been created for patients and collaborators, who can access their own individual profile and consult their personal information. This personalization of the application is designed to improve the quality of care for patients and employees, offering an experience tailored to the needs of each user.

The web application has been developed using React and Node.js technologies, for the creation of the front-end and back-end, respectively, which allow for a modern and efficient development environment.

Keywords

Web application, management, dental clinic, personalization, user roles, patients, collaborators, billing, React, Node js.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1 - Introducción

1.1 Motivación

La gestión de una clínica dental es un proceso complejo que incluye la planificación de las citas de los pacientes, el control de inventario de los materiales empleados en los tratamientos y la gestión financiera y contable del negocio. Existen varias aplicaciones en el mercado, especializadas en las tareas mencionadas, que facilitan la gestión de este tipo de negocios. Estas aplicaciones permiten tener un usuario administrador que, a través de módulos, puede gestionar distintas partes de la actividad de la clínica dental en función de las necesidades concretas.

Tras consultar la opinión sobre estas aplicaciones a los dentistas y a algunos de los pacientes fieles de una clínica dental localizada en un barrio céntrico de Madrid, se ha identificado una necesidad que las aplicaciones existentes no están cubriendo. Esta necesidad consiste en la posibilidad de una personalización de la aplicación, de forma que no solo pueda acceder a ella el administrador de la clínica, sino que también los propios dentistas y pacientes puedan acceder a su propio perfil individual.

Esta idea es el factor principal para el desarrollo de una nueva aplicación web que siga ayudando a gestionar la clínica dental, y además que cada una de las personas que interactúan con la clínica (dentistas y pacientes) puedan también tener un perfil individual en la aplicación para consultar su información personal.

De esta forma se fomenta la participación de los trabajadores y clientes en la gestión de la clínica dental, ofreciéndoles la oportunidad de llevar un control personal sobre su historial de citas, tratamientos pendientes de realizar, así como de los cobros y pagos realizados, respectivamente. Gracias a estos perfiles individuales se puede mejorar la calidad de atención y la eficiencia en la gestión de la clínica. Como consecuencia esto puede verse reflejado en una mayor satisfacción por parte del cliente, que en el sector dental es un factor clave para el buen funcionamiento del negocio.

En resumen, el objetivo del Trabajo de Fin de Grado es desarrollar una aplicación web que ayude a la gestión de una clínica dental pero con la posibilidad de personalización, a través de un perfil individual diferenciando el tipo de usuario que inicie sesión en la aplicación: Administrador, Dentista o Paciente. De esta forma mejorar la calidad de atención y la eficiencia en la gestión.

1.2 Objetivos

El objetivo principal del trabajo es el desarrollo de una aplicación web para la gestión de una clínica dental personalizada para cada tipo de usuario: Administrador, colaborador o paciente, para mejorar la calidad de atención.

Los objetivos específicos son:

- Permitir la visualización y gestión de los datos del paciente, como datos personales, historial médico, próximas citas y pagos pendientes de realizar.
- Ofrecer un usuario individual para los pacientes para que puedan acceder a la información mencionada en el punto anterior.
- Permitir la visualización y gestión de los datos de los dentistas colaboradores, como datos personales, próximas citas, historial de sus pacientes y facturación.
- Ofrecer un usuario individual para los colaboradores para que puedan acceder a la información mencionada en el punto anterior.
- Permitir la visualización y gestión de los datos de los proveedores, como datos personales, pedidos realizados y facturación.
- Facilitar gestión de la facturación y registro de cobros de los pacientes y pagos a colaboradores y proveedores.
- Facilitar la gestión del inventario cuando se producen entradas o salidas de materiales
- Proporcionar una interfaz intuitiva y fácil de usar para todos los usuarios.

1.3 Plan de trabajo

Para cumplir con los objetivos mencionados en el apartado anterior el desarrollo del trabajo se ha dividido en las siguientes tareas con unos plazos marcados:

Nombre actividad	Código actividad	Fecha Inicio	Duración en días	Fecha Fin
Definición de funcionalidades	Actividad 1	16/01/2023	14	30/01/2023
Desarrollo de los bocetos de pantallas para la aplicación	Actividad 2	23/01/2023	14	06/02/2023
Desarrollo de la funcionalidad "Gestión de usuarios"	Actividad 3	20/02/2023	21	13/03/2023
Desarrollo de la funcionalidad "Gestión de facturación"	Actividad 4	06/03/2023	21	27/03/2023
Desarrollo de la funcionalidad "Gestión de inventario"	Actividad 5	09/03/2023	10	19/03/2023
Desarrollo de la funcionalidad "Gestión de citas"	Actividad 6	13/03/2023	21	03/04/2023
Desarrollo del borrador de la memoria	Actividad 7	03/04/2023	45	18/05/2023
Correcciones para la memoria definitiva	Actividad 8	18/05/2023	10	28/05/2023

Tabla 1.3—1. Plan de trabajo

En el siguiente diagrama de Gantt se muestra la planificación seguida estos meses de manera gráfica:

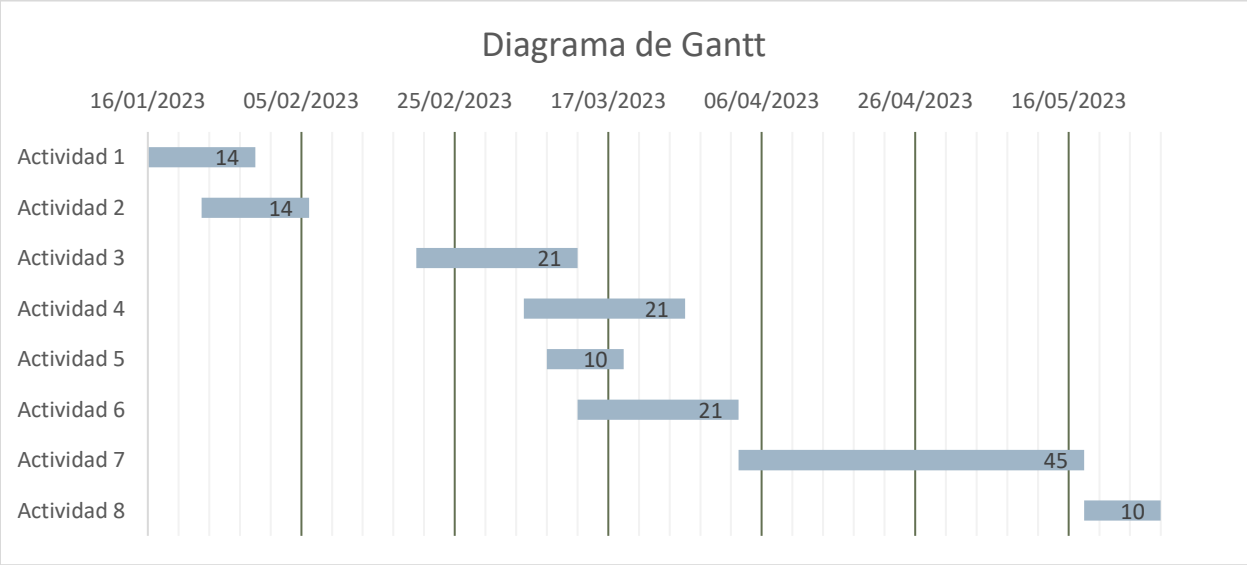


Figura 1.3—1. Diagrama de Gantt

1.4 Estructura del documento

El documento sigue la estructura que se presenta a continuación:

- En este capítulo (Capítulo 1- Introducción) se expone la motivación para el desarrollo de la aplicación web y se plantean los objetivos que se pretenden conseguir con el trabajo.
- En el “Capítulo 2 - Estado del arte”, se explica porque es necesaria una aplicación de gestión en las clínicas dentales, se analizan las aplicaciones existentes en el mercado y se plantea una propuesta de innovación proponiendo una funcionalidad que no existe en los software que se encuentran en el mercado.
- En el “Capítulo 3 - Desarrollo de la aplicación web” se recogen todos los pasos que se han llevado a cabo durante la creación de la aplicación.
    - En primer lugar en el apartado "3.1" se detallan las funcionalidades que se llevarán a cabo.
    - En el apartado "3.2" se explican de manera detallada las tecnologías utilizadas para el desarrollo de la aplicación. Centrándose en la explicación de React y Node.js.
    - En el punto "3.3" se explicará cómo esta creada la base de datos y las relaciones utilizadas entre las tablas. Este apartado se complementa con la información incluida en el "Apéndice A - ", que incluye de manera detallada la explicación de cada tabla.
    - En el apartado "3.4" se incluyen algunos bocetos de las pantallas diseñadas al inicio del trabajo, antes de comenzar con el desarrollo de la aplicación definitiva.
    - En el punto "3.5", se muestran los casos de uso que forman la aplicación mediante cuatro diagramas de casos de uso. Este apartado se complementa con la información incluida en el "Apéndice B - ", que incluye de manera detallada la explicación de cada caso de uso.

    - En el punto "3.6" se explica de manera detallada la estructura del proyecto y se muestran algunas de las pantallas más relevantes de la aplicación web.

- Por último, el “Capítulo 4 - ”, incluye un análisis del proceso de desarrollo y se incluyen propuestas de desarrollo futuro.
- Además, se incluye un último apéndice, “Apéndice C - ”, que recoge las credenciales para poder acceder a la aplicación web.

Introduction

Motivation

The management of a dental clinic is a complex process that includes the planning of patient appointments, inventory control of the materials used in the treatments and the financial and accounting management of the business. There are several applications on the market, specialised in the aforementioned tasks, that facilitate the management of this type of business. These applications allow an administrator user who, through modules, can manage different parts of the dental clinic's activity according to specific needs.

After consulting the opinion of dentists and some of the loyal patients of a dental clinic located in a central district of Madrid about these applications, a need has been identified that the existing applications are not covering. This need consists of the possibility of personalising the application, so that not only the administrator of the clinic can access it, but also the dentists and patients themselves can access their own individual profile.

This idea is the main factor for the development of a new web application that continues to help manage the dental clinic, and also that each of the people who interact with the clinic (dentists and patients) can also have an individual profile in the application to consult their personal information.

In this way, the participation of employees and customers in the management of the dental clinic is encouraged, offering them the opportunity to keep personal control over their appointment history, pending treatments, as well as collections and payments made, respectively. Thanks to these individual profiles, the quality of care and the efficiency of practice management can be improved. As a result, this can be reflected in increased customer satisfaction, which in the dental sector is a key factor for the smooth running of the business.

In summary, the objective of the Final Degree Project is to develop a web application that helps in the management of a dental clinic but with the possibility of personalisation, through an individual profile differentiating the type of user who logs into the application: Administrator, Dentist or Patient. This will improve the quality of care and management efficiency.

Goals

The main objective of the work is the development of a web application for the management of a dental clinic personalised for each type of user: Administrator, collaborator or patient, in order to improve the quality of care.

The specific objectives are:

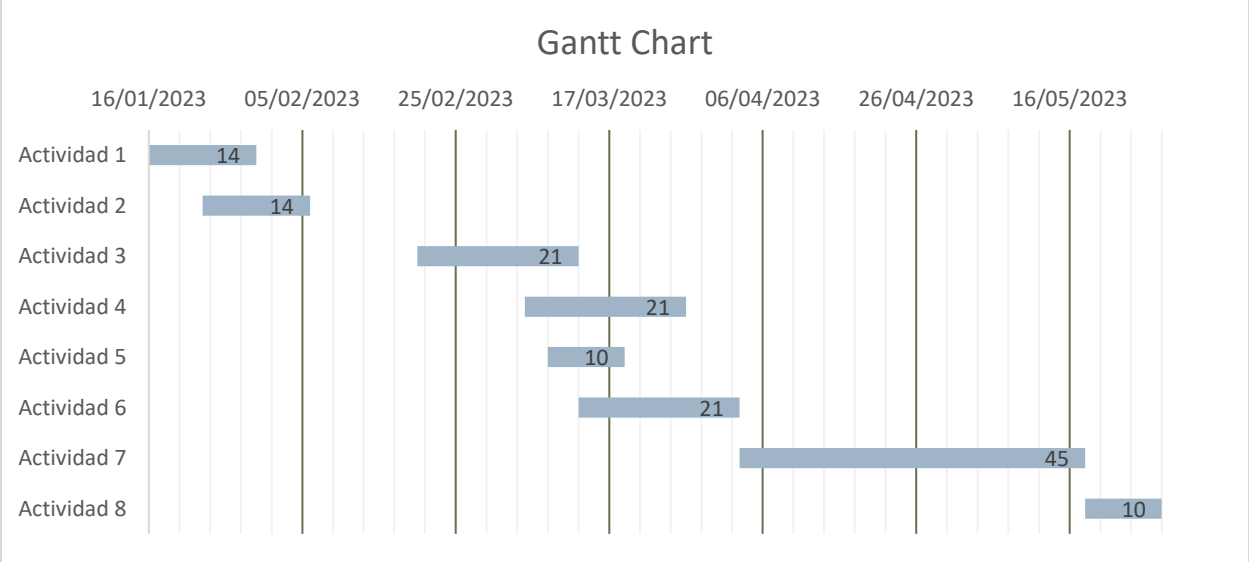
- To allow the visualisation and management of patient data, such as personal data, medical history, upcoming appointments and payments to be made.
- To provide an individual user for patients so that they can access the information mentioned in the previous point.
- Enable the visualisation and management of collaborating dentists' data, such as personal data, upcoming appointments, patient history and invoicing.
- Provide an individual user for the collaborators so that they can access the information mentioned in the previous point.
- Enable the display and management of supplier data, such as personal details, orders placed and invoicing.
- Facilitate the management of invoicing and the recording of patient collections and payments to collaborators and suppliers.
- Facilitate inventory management when incoming or outgoing materials occur.
- Provide an intuitive and user-friendly interface for all users.

Work plan

In order to meet the objectives mentioned in the previous section, the development of the work has been divided into the following tasks with set deadlines:

Activity name	Activity code	Start date	Duration in days	End date
Definition of functionalities	Activity 1	16/01/2023	14	30/01/2023
Development of the sketches of screens for the application	Activity 2	23/01/2023	14	06/02/2023
Development of the functionality "User management"	Activity 3	20/02/2023	21	13/03/2023
Development of the functionality "Billing management"	Activity 4	06/03/2023	21	27/03/2023
Development of the functionality "Inventory Management"	Activity 5	09/03/2023	10	19/03/2023
Development of the functionality “Appointment Management”	Activity 6	13/03/2023	21	03/04/2023
Development of the draft report	Activity 7	03/04/2023	45	18/05/2023
Corrections for the final report	Activity 8	18/05/2023	10	28/05/2023

The following Gantt chart shows the planning followed during these months in a graphical way:



Document structure

The document follows the structure presented below:

- In this chapter (Chapter 1- Introduction), the motivation for the development of the web application is set out and the objectives to be achieved with the work are stated.
- In "Chapter 2 - State of the art", it explains why a management application is necessary in dental clinics, analyses the existing applications on the market and proposes a proposal for innovation by proposing a functionality that does not exist in the software available on the market.
- In "Chapter 3 - Development of the web application", all the steps that have been carried out during the creation of the application are described.
    - First of all, in section "3.1" the functionalities that will be implemented are detailed.
    - In section "3.2" the technologies used for the development of the application are explained in detail. Focusing on the explanation of React and Node.js.
    - Section "3.3" explains how the database is created and the relationships used between the tables. This section is complemented with the information included in "Appendix A", which includes a detailed explanation of each table.
    - Section "3.4" includes some sketches of the screens designed at the beginning of the work, before starting the development of the final application.
    - In section "3.5", the use cases that make up the application are shown by means of four use case diagrams. This section is complemented by the information included in "Appendix B", which includes a detailed explanation of each use case.
    - Section "3.6" explains in detail the structure of the project and shows some of the most relevant screens of the web application.

- Finally, "Chapter 4" includes an analysis of the development process and proposals for future development.
- In addition, a final appendix, "Appendix C", which includes the credentials for accessing the web application, is included.

Capítulo 2 - Estado del arte

Antes de comenzar con una explicación detallada sobre el desarrollo de la aplicación web, es interesante conocer la situación actual del sector dental en relación con la gestión del negocio y como se lleva este proceso a cabo. También se analizarán las aplicaciones disponibles en el mercado que utilizan las clínicas actualmente como ayuda en este procedimiento de organización y control.

2.1 Introducción y antecedentes en la gestión de las clínicas dentales

La gestión eficiente en una clínica dental, como en cualquier negocio, es algo muy importante que hay que tener en cuenta. En el proceso de gestión hay que tener en cuenta distintos aspectos como:

- La coordinación de los distintos colaboradores que trabajan en la empresa. Estos colaboradores son dentistas especializados en diferentes áreas como: odontología general, ortodoncia, endodoncia, entre otras.
- La gestión eficiente del calendario de citas, coordinando los días que podrá acudir cada colaborador a la clínica y cuando citar a los pacientes con un colaborador concreto.
- Realización de los pedidos a los proveedores y gestión el inventario para disponer de los materiales necesarios para los tratamientos.
- El control de cobros de los tratamientos
- El control de gastos como: pagos a los colaboradores o pagos de facturas, entre otros gastos.

Un gran problema que existe es que la mayoría de las personas que trabajan en una clínica dental tienen altos conocimientos sobre odontología pero desconocen cómo administrar una empresa. Es por ello por lo que existen distintas soluciones tecnológicas que ayudan a los administradores de las clínicas (que en muchos casos es el propio dentista) a realizar las distintas funciones explicadas anteriormente de manera más sencilla. En el apartado “2.2 Aplicaciones de

referencia" se analizarán las funcionalidades que tienen las distintas aplicaciones existentes en el mercado. Y en el apartado "2.3 Propuesta de innovación" se presentará una idea que cubre una necesidad específica que ninguna de las aplicaciones del mercado actualmente aborda.

2.2 Aplicaciones de referencia

2.2.1 Clinic Cloud

Según la página oficial de Clinic Cloud, desarrollada por Doctoralia Internet SL, la aplicación ofrece las siguientes funcionalidades a través de distintos módulos a los que se puede acceder desde la nube (Doctoralia Internet SL, s.f.):

- Gestión de citas médicas: Gestiona las citas de los pacientes y permite enviarles recordatorios a través de SMS.
- Gestión de historias clínicas: La aplicación permite acceder a toda la historia clínica de un paciente.
- Módulo de odontogramas: Permite el almacenamiento de odontogramas.
- Contabilidad: Permite administrar la contabilidad y realizar presupuestos.
- Análisis de productividad: Conocer distintos datos relacionados con los procesos del negocio.
- Opciones de marketing: Gestiona el envió de campañas de marketing a través de correos masivos.

2.2.2 Bookgy

Según la página oficial de Bookgy, se trata de una aplicación con tecnología cloud que permite acceder al software desde cualquier lugar y dispositivo, desde la que se pueden realizar las siguientes acciones (Bookgy, s.f.):

- Programa de citas: Permite controlar las citas de cada día para cada uno de los dentistas.

- **Gestión de facturas y cobros:** Desde la aplicación se pueden controlar los cobros, deudas, caja diaria o gasto medio por cliente.
- **Recordatorios y comunicaciones:** Envía una notificación por SMS o correo a los pacientes para avisarles de su próxima cita.
- **Web, posicionamiento y marketing:** Ayuda a la clínica dental con el desarrollo de su propia página web.

2.2.3 OrisLine

Según la página oficial de OrisLine, es un programa que desarrolla las siguientes funcionalidades (OrisLine, s.f.):

- **Gestión completa:** Incluye la organización de actividades clínicas y contables. Además permite realizar análisis cefalométricos.
- **Análisis estadístico:** Permite controlar la evolución de las actividades que se desempeñan en la clínica y también controlar el alcance de los objetivos propuestos.
- **Fidelizar a los pacientes:** Permite el envío de comunicaciones específicas a los pacientes que están incluidos en la base de datos.

2.2.4 Gesden G5

Según la página oficial de Infomed Software, empresa desarrolladora de Gesden, es una aplicación de gestión clínica que cuenta con las siguientes funcionalidades divididas en módulos (Infomed Software Sanitario, s.f.):

- **Administración:** Permite la gestión de facturas, recibos y comprobantes. También controla la gestión de pedidos, movimientos de inventario y control de gastos. En este módulo también se puede acceder a estudios comparativos a partir de los datos introducidos en el programa.
- **Informes:** Desde la aplicación se pueden generar distintos listados como por ejemplo: pacientes, historiales, pacientes (deudas), tarifas, …

- Sistemas: Tiene la posibilidad de integración con Microsoft Outlook para el envío de correos a pacientes y acceso a un listín telefónico.
- Pacientes: Gestión de los datos de los pacientes e historial clínico, con odontograma y la creación de recetes médicas.
- Agenda: Permite tener una agenda para gestionar varios gabinetes con vista semanal y diaria y controlar la lista de espera.
- SMS e emailing: Comunicación por SMS o correo con los pacientes, con la posibilidad de enviar comunicaciones de forma masiva.

2.3 Propuesta de innovación

Ya se ha comentado que es imprescindible que exista una adecuada gestión de la clínica para que el negocio funciona de manera correcta. Sin embargo, existen algunos factores que pueden favorecer también en la gestión eficiente de la empresa y ninguna de las aplicaciones existentes en el mercado, vistas en el punto anterior, tiene en cuenta.

Ninguna de los programas ofrece al paciente la posibilidad de acceder a su propio perfil para consultar sus citas, en caso de que no la recuerden, ni la opción de consultar su información sobre tratamientos y pagos realizados y pendientes.

Una solución interesante a esta cuestión es dar la opción de que los propios pacientes tengan acceso a su perfil individual en el que consultar su calendario de citas, desde el que acceder a su historial de tratamientos ya realizados y consultar también los próximos pagos que les tocará abonar.

Además de crear un perfil individual para los pacientes también es interesante crear un perfil para los colaboradores. De esta forma los distintos dentistas también podrán consultar su propio calendario de las citas que tienen en su agenda y acceder al historial de cada paciente al que tienen que atender.

Con esta nueva idea que no está desarrollada en ninguna de las aplicaciones existentes se consigue además de una mejora en la eficiencia de la clínica dental una mejor experiencia tanto del paciente como de los colaboradores.

Capítulo 3 - Desarrollo de la aplicación web

3.1 Funcionalidades de la aplicación

A continuación se explicará de manera detallada las distintas funciones y capacidades que se ofrecen a los usuarios en la aplicación: Gestión de una clínica dental.

3.1.1 Gestión de usuarios

3.1.1.1 Tipos de perfil de usuario

En la aplicación existen tres tipos de perfil de usuario:

- Administrador: Los usuarios con perfil "admin" pueden acceder a toda la información existente en el programa: datos de pacientes, datos de colaboradores, calendario de citas, facturas, inventario, … Además pueden crear, editar o eliminar esta información.
- Paciente: Los usuarios con perfil "paciente" únicamente podrán acceder únicamente con permisos de visualización a la información relacionada con su usuario: datos personales, historial clínico, calendario de citas y pagos realizados y pendientes de abonar.
- Colaborador: Los usuarios con perfil "colaborador" podrán acceder únicamente con permisos de visualización a la información relacionada con sus pacientes, calendario de citas agendadas y sus facturas mensuales.

3.1.1.2 Registro de los usuarios

La creación de nuevos usuarios en la aplicación está exclusivamente a cargo la clínica dental desde el usuario con perfil "Administrador". Por lo tanto, los usuarios "Pacientes" y "Colaboradores" no pueden registrarse de manera independiente.

El registro de un nuevo usuario se realiza al dar de alta a un nuevo paciente o colaborador en la base de datos. En este momento además de rellenar los datos personales correspondientes,

se genera un ID de usuario y se asigna una contraseña. Estas credenciales permitirán a los pacientes y colaboradores acceder a su perfil individual en la aplicación.

Adicionalmente, en el caso de los usuarios "Pacientes" el registro consta de dos partes. En primer lugar, el formulario de datos personales mencionado anteriormente. Y en segundo lugar, un nuevo formulario para añadir las relaciones (padre, hijo, amigo, pareja, …) que pueda tener este paciente que se está añadiendo con otros pacientes que se encuentren en la base de datos previamente. La intención de este segundo formulario es mantener la cercanía con los pacientes, pudiendo conocer qué familiares o conocidos también acuden a la clínica y poder interesarse por ellos.

3.1.1.3 Inicio de sesión

El inicio de sesión se realiza mediante las credenciales ID usuario y contraseña. Tras comprobar que estas credenciales son correctas, se identificará el tipo de usuario que ha iniciado sesión para mostrar el inicio de la aplicación correspondiente: Menú Principal Administrador, Perfil Paciente o Menú Principal Colaborador. De esta forma se proporciona a cada tipo de usuario un acceso personalizado con la información correspondiente en cada caso.

3.1.2 Gestión de citas

3.1.2.1 Creación de citas

La creación de nuevas citas en la aplicación está exclusivamente a cargo la clínica dental desde el usuario con perfil "Administrador". Por lo tanto, los usuarios "Pacientes" y "Colaboradores" no pueden añadir nuevas citas desde sus perfiles individuales.

Para añadir una cita, se rellena un formulario con distintos campos: Tratamiento, Paciente, Colaborador, Fecha, Precio y Observaciones. Además, en este formulario se selecciona el material que se va a consumir y cuanta cantidad de dicho producto al realizar el tratamiento.

Esta información mencionada, relacionada con los materiales utilizados, se registra para poder actualizar el inventario, ajustando las existencias actuales en función de los productos utilizados. Asimismo, se calculan los costes asociados a los materiales consumidos, tomando como

referencia el precio unitario registrado en el inventario. Posteriormente, se repercuten estos costes en la factura del colaborador en una proporción del 50%. El procedimiento detallado para la generación de facturas se explica en el punto “3.1.4 Gestión de facturación”.

3.1.2.2 Visualización de citas

Para visualizar el calendario de citas existen en la aplicación tres opciones de visualización para ofrecer información distinta dependiendo del contexto.

- Visualización de todas las citas: En el calendario aparecerán todas las citas de todos los pacientes y colaboradores en un periodo determinado. A esta visualización solo tendrá acceso el usuario con perfil “Administrador”.
- Visualización de las citas de un colaborador: En el calendario aparecerán todas las citas que tenga que atender el colaborador en concreto en un periodo determinado. A esta visualización tendrán acceso el usuario con perfil “Administrador” y los usuarios con perfil “Colaborador”.
- Visualización de las citas de un paciente: En el calendario aparecerán todas las citas de ese paciente en un periodo determinado. A esta visualización tendrán acceso el usuario con perfil “Administrador” y los usuarios con perfil “Pacientes”.

En las tres opciones de visualización aparece en la vista un calendario mensual con las citas correspondientes a cada opción. Esta vista se puede cambiar a diaria, semanal o agenda.

3.1.3 Gestión de inventario

3.1.3.1 Entrada de existencias en el inventario

Los productos del inventario se actualizan al introducir facturas de proveedores en la aplicación. Al añadir una factura, se especifica el producto adquirido, la cantidad y el precio total. Para obtener más información sobre estas facturas ver el apartado “3.1.4.3 Facturas de proveedores”.

Con los datos obtenidos a partir de la factura agregada, se actualiza la cantidad del producto específico en el inventario, así como el precio unitario de ese producto. Para calcular el precio unitario del producto actualizado, se utiliza la técnica de Precio Medio Ponderado (PMP).

El cálculo del PMP es una técnica para la valoración de inventarios, para su cálculo se utiliza una media ponderada. Consiste en sumar el coste total de las entradas en el inventario y dividir entre la cantidad total de dicho material (Flamarique, 2018). Obteniendo así un precio unitario aplicable a todas las existencias que hay en el almacén de un determinado producto.

3.1.3.2 Salida de existencias del inventario

Las existencias de un producto se actualizan cuando se crea una cita de un paciente. Al crear una cita se especifica el material que se va a consumir y la cantidad correspondiente de dicho producto. Estos datos se utilizan para actualizar las existencias del inventario, reflejando así la salida de los productos utilizados en el tratamiento.

3.1.4 Gestión de facturación

Para gestionar la facturación hay tres tipos de documentos:

3.1.4.1 Tickets de clientes

Los tickets de los clientes hacen referencia a cada uno de los pagos que deberá abonar un paciente por el tratamiento realizado en la clínica, es decir, con cada cita programada se genera automáticamente un ticket asociado.

El ticket se crea con el DNI del paciente, el tratamiento realizado, la fecha y el importe a abonar. Además se registra un campo denominado “cobrado” que inicialmente tiene como valor “No”, indicando que el paciente aún no realizado el pago correspondiente.

El valor de este campo “cobrado” se actualiza cuando el paciente acude a la clínica y realiza el pago del tratamiento que se haya realizado.

3.1.4.2 Facturas de colaboradores

Las facturas de los colaboradores se generan de manera mensual para cada colaborador. Estas facturas están formadas por tickets de colaboradores que se generan automáticamente cada vez que un paciente abona el pago de su ticket de paciente (ver apartado "3.1.4.1 Tickets de clientes").

Para generar cada ticket de colaborador, se divide el importe abonado por el paciente y los costes asignados al tratamiento entre dos. De esta manera, se asigna la mitad de los ingresos, teniendo en cuenta los costes de materiales, al colaborador, mientras que la otra mitad corresponde a la clínica.

Con cada ticket de colaborador que se genera, se consulta la fecha de este y se asigna a la factura del colaborador y mes correspondiente. Así, cada factura tiene un importe total que es la suma de todos los tickets incluidos en ese mes.

Cada factura de colaborador tiene un campo denominado "pagado" que inicialmente tiene valor "No". Cuando se realiza el pago de la factura al colaborador, el administrador puede acceder a las facturas pendientes de pago y editar este campo.

3.1.4.3 Facturas de proveedores

Las facturas de los proveedores se añaden de manera manual a través de un formulario. En este formulario se indica el proveedor al que corresponde la factura, la fecha, el producto comprado, la cantidad y el precio total.

Al añadir una nueva factura se actualizan los productos del inventario (ver apartado "3.1.3.1 Entrada de existencias en el inventario").

Estas facturas también tienen un campo denominado "pagado" para llevar un control de las pendientes de pago a los proveedores.

3.2 Tecnologías utilizadas

3.2.1 Desarrollo Front-end

Para el desarrollo del front-end se ha utilizado principalmente React, una biblioteca de JavaScript para crear las interfaces del usuario. Además, se han empleado otras bibliotecas de JavaScript necesarias para complementar el desarrollo del lado del cliente que se explicarán a continuación con la documentación obtenida de la página web oficial de Node Package Manager (NPM), un gestor de paquetes muy utilizado en JavaScript.

3.2.1.1 React

Para el desarrollo del lado del cliente se ha utilizado react, una biblioteca de JavaScript. Y la sintaxis que utiliza es JavaScript XML (JSX), que es una combinación de HTML y JavaScript. Según la página oficial de React: "React te permite construir interfaces de usuario a partir de piezas individuales llamadas componentes" (Meta Open Source, s.f.).

Según Narayn (2022), un componente es una clase de JavaScript que, a partir de unas entradas opcionales, genera un elemento de React. La idea de React es dividir en piezas cada pantalla y que cada una de estas piezas sea un componente. Narayn (2022) también explica que para crear una página completa se pueden combinar distintos componentes individuales.

Para utilizar el mismo componente en distintas páginas, es necesario conseguir una personalización de este, que permita adaptar el componente a las distintas necesidades que puedan existir en cada caso.

Esta personalización se consigue gracias a las entradas opcionales mencionadas antes, estas entradas se conocen como "propiedades", comúnmente llamadas "props" y permiten establecer una configuración determinada para cada componente que se crea. De esta manera, los componentes se pueden reutilizar en distintas páginas permitiendo el desarrollo de una aplicación web eficiente y escalable.

A continuación, para una mejor comprensión del concepto "componente" se añade una imagen de una de las pantallas de la aplicación (perfil de un colaborador) y se identifican cada uno de los

componentes que la forman: MostrarDatos y Calendario (en el apartado “3.6.1Estructura del proyecto e implementación del front-end” se explica cada componente de manera detallada):

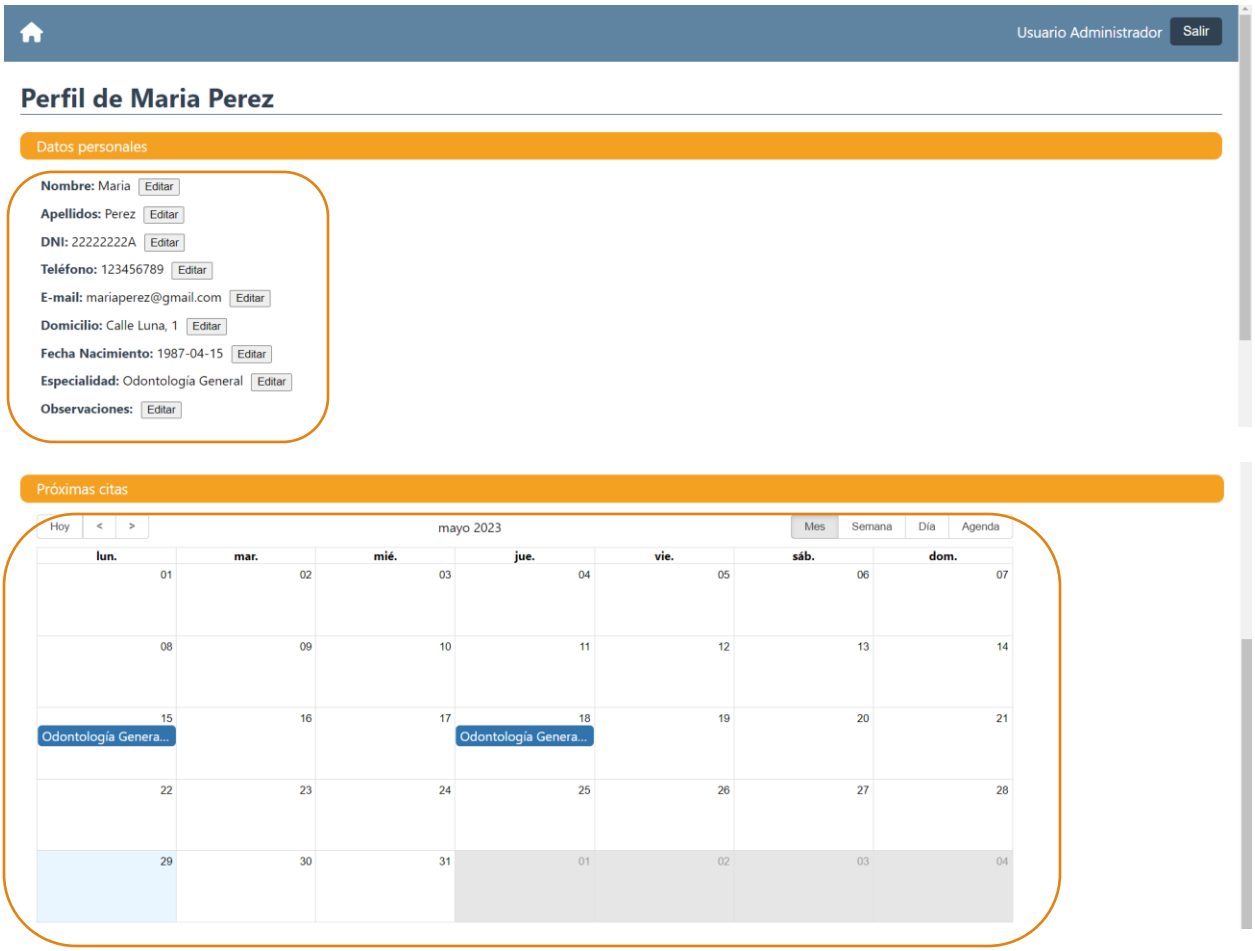


Figura 3.2—1. Perfil colaborador y los componentes que forman esta pantalla

¿Por qué React? React está siendo muy utilizado en el desarrollo de aplicaciones webs modernas y una de las principales razones es la eficiencia en la manipulación del Document Object Model (DOM). Cada navegador realiza una representación en forma de árbol del documento HTML que está leyendo, esto se denomina el DOM, y este se actualiza cuando se produce algún cambio en el código (Narayn, 2022):

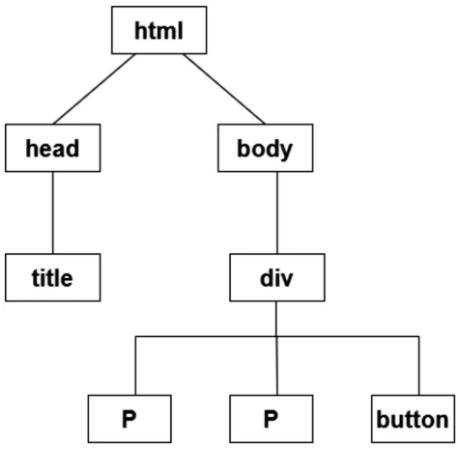


Figura 3.2—2. Representación del DOM

Narayn (2022) sostiene que React implementa el concepto Virtual DOM, que funciona de manera diferente a la manipulación directa del DOM. El autor, explica en que consiste este concepto, en lugar de modificar directamente el DOM del navegador, se crean dos copias del DOM en memoria. Cuando se realiza un cambio, React únicamente modifica una de las copias del DOM. Después se comparan ambas copias para identificar que ha cambiado, este proceso se conoce como diferenciación.

Narayn también señala que para conseguir controlar de esta manera los cambios que se producen, React encapsula cada sección de la interfaz en componentes y los cambios que se producen en los componentes se controlan a partir de su estado. Cuando se produce un cambio en el estado de un componente, esos cambios se aplican al Virtual DOM y se realiza una comparación para determinar qué partes del DOM deben actualizarse (Narayn, 2022):

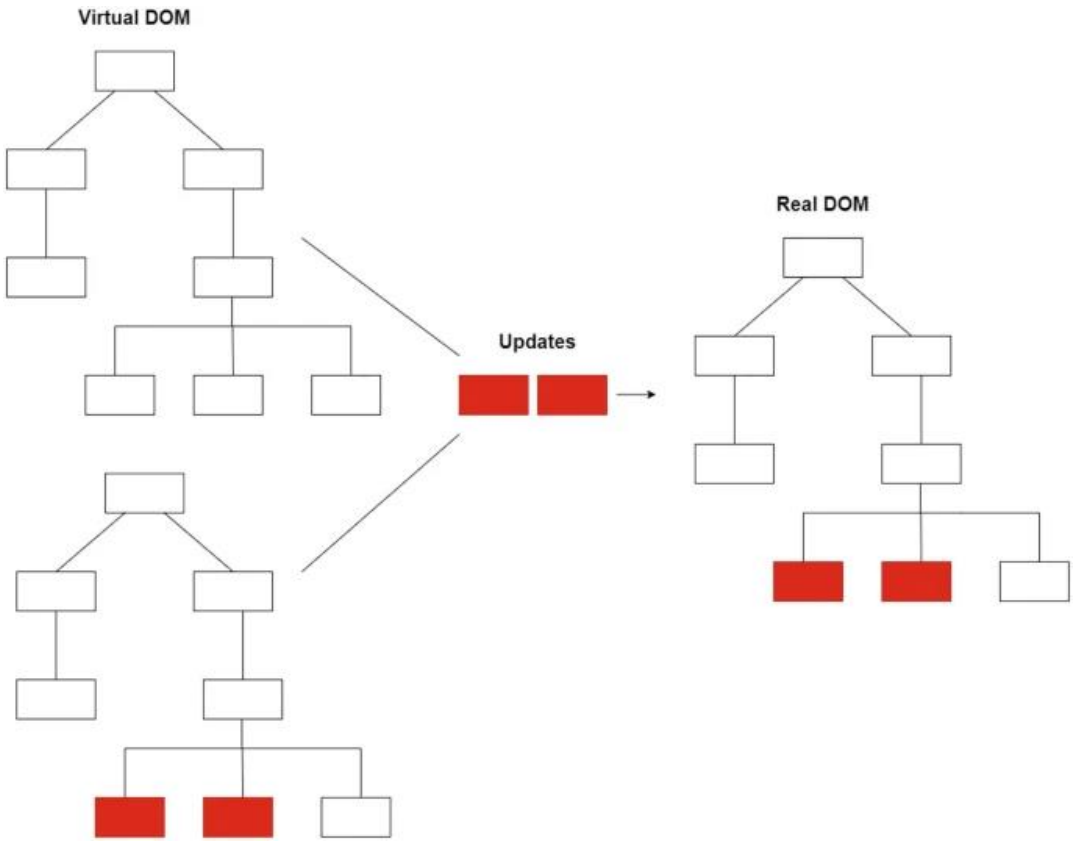


Figura 3.2—3. Funcionamiento del virtual DOM en React

A través de este mecanismo el DOM se manipula mucho menos, lo que permite mejorar el rendimiento.

3.2.1.2 react-dom

react-dom es un paquete complementario de la librearía de React, que permite realizar la funcionalidad descrita en el apartado anterior sobre el DOM Virtual (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 18.2.0.

3.2.1.3 react-router-dom

El paquete react-router-dom permite la navegación entre diferentes páginas en la aplicación de React (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 6.8.1.

3.2.1.4 axios

El paquete **axios** permite realizar solicitudes HTTP desde el lado del cliente y se utiliza para realizar peticiones al servidor (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 1.3.3.

3.2.1.5 jwt-decode

El paquete **jwt-decode** permite decodificar un JSON Web Token (JWT), estos tokens se utilizan para la autenticación de los usuarios en la aplicación (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 3.1.2. La creación del JWT se realiza desde el back-end con la biblioteca explicada en el apartado "3.2.2.6 jsonwebtoken".

3.2.1.6 moment

El paquete **moment** permite manipular, validar y formatear fechas y horas (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 2.29.4.

3.2.1.7 react-big-calendar

El paquete **react-big-calendar** proporciona un componente que crea un calendario sobre el que se puede navegar y visualizar los eventos en vista mensual, semanal o diaria (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 1.6.9.

3.2.2 Desarrollo Back-end

3.2.2.1 Node.js

Para el desarrollo del lado del servidor se ha utilizado Node.js. Según la página oficial de Node.js, se trata de un entorno de ejecución de JavaScript orientado a eventos asíncronos, con el objetivo para realizar el desarrollo de un servidor de manera escalable, esta escalabilidad se consigue gracias a la implementación de un modelo de operaciones I/O directamente, sin bloqueo (OpenJS Foundation, s.f.).

3.2.2.2 express

express es un framework de JavaScript, rápido y minimalista para facilitar el desarrollo de aplicaciones con Node.js (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 4.18.2.

3.2.2.3 mysql2

El paquete mysql2 permite la conexión y la interacción con una base de datos MySQL (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 3.1.2.

3.2.2.4 sequelize

sequelize es un ORM (Object-Relational Mapping) de Node.js que permite interactuar con bases de datos relacionales y proporciona los métodos necesarios para realizar consultas a la base de datos basado en promesas para MySQL (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 6.28.0.

3.2.2.5 bcrypt

El paquete bcrypt se utiliza para obtener el hash de una contraseña para almacenarla de forma segura en la base de datos, esta biblioteca también permite comparar contraseñas cifradas (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 5.1.0.

3.2.2.6 jsonwebtoken

El paquete jsonwebtoken permite crear y verificar JSON Web Tokens (JWT), esto son tokens de seguridad que se utilizan para la autenticación y transmisión segura de información entre las distintas partes de la aplicación (npm, Inc, s.f.). En el desarrollo de la aplicación se ha utilizado la versión 9.0.0.

3.2.3 Base de datos

3.2.3.1 MySQL

Para gestionar la base de datos se ha utilizado el sistema MySQL, que se basa en un modelo relacional. Esto significa que los datos se organizan en tablas con filas y columnas, y se pueden establecer relaciones entre tablas mediante claves primarias y foráneas.

En la interacción con MySQL, se utiliza el lenguaje SQL para la creación, actualización y consulta de datos. Y para la administración de la base de datos se ha utilizado la herramienta phpMyAdmin, que se explica en el siguiente apartado “3.2.4 Entorno de desarrollo”.

3.2.4 Entorno de desarrollo

3.2.4.1 Visual Strudio Code

Para el desarrollo del front-end y el back-end se ha utilizado la herramienta Visual Studio Code. Según la web oficial de Visual Studio Code, se trata de un entorno de desarrollo, donde se puede escribir y depurar código en distintos lenguajes de programación desde una interfaz intuitiva (Microsoft, s.f.).

3.2.4.2 phpMyAdmin

Según la web oficial de phpMyAdmin (s.f.), se trata de una herramienta que permite administrar bases de datos MySQL a través de una interfaz web sencilla de usar. Permite realizar operaciones crear tablas o gestionar las relaciones desde la interfaz de usuario y también es posible ejecutar directamente cualquier instrucción SQL (phpMyAdmin, s.f.).

3.3 Base de datos

3.3.1 Normalización

Durante el desarrollo de la base de datos se ha aplicado el proceso de normalización en el diseño de todas las tablas intentando cumplir hasta la tercera forma normal. Es decir, se han aplicado una serie de reglas para estructurar los datos de forma eficiente y eliminar la redundancia.

En cada tabla todos los atributos creados son atómicos, esto significa que cada atributo es indivisible. Además, se ha evitado la duplicidad de información organizando los datos en tablas separadas.

Un ejemplo de esto se encuentra en las tablas “colaboradores” y “tratamientos”, donde se incluye el campo “codigoEspecialidad”. En vez de añadir en ambas tablas también el campo “nombreEspecialidad” se ha creado una nueva tabla “especialidades”. Esta nueva tabla recoge ambos campos: “codigoEspecialidad” y “nombreEspecialidad”, evitando así la duplicación de datos:

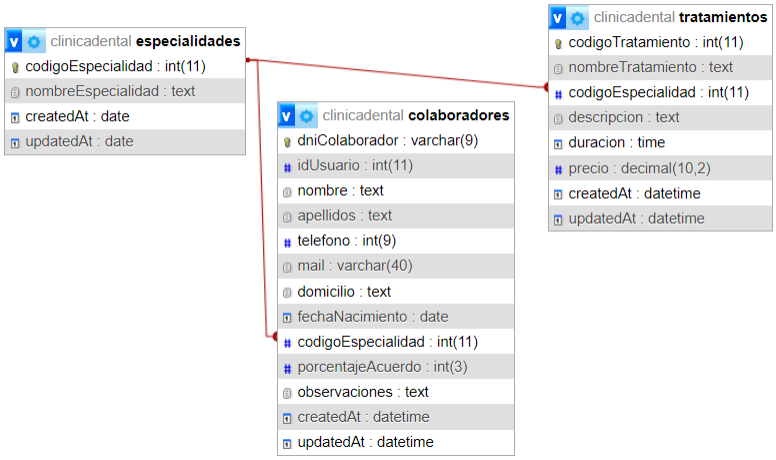


Figura 3.3—1. Ejemplo de tablas de la base de datos normalizadas

Asimismo, al crear nuevas tablas para evitar redundancia de datos, es necesario crear relaciones entre las tablas utilizando claves primarias y foráneas, para garantizar la consistencia de datos. En el siguiente apartado se muestran las claves y relaciones utilizadas en cada tabla.

3.3.2 Estructura de la base de datos y relaciones creadas

El siguiente diagrama entidad-relación refleja el conjunto de relaciones existentes entre las distintas tablas en la base de datos:

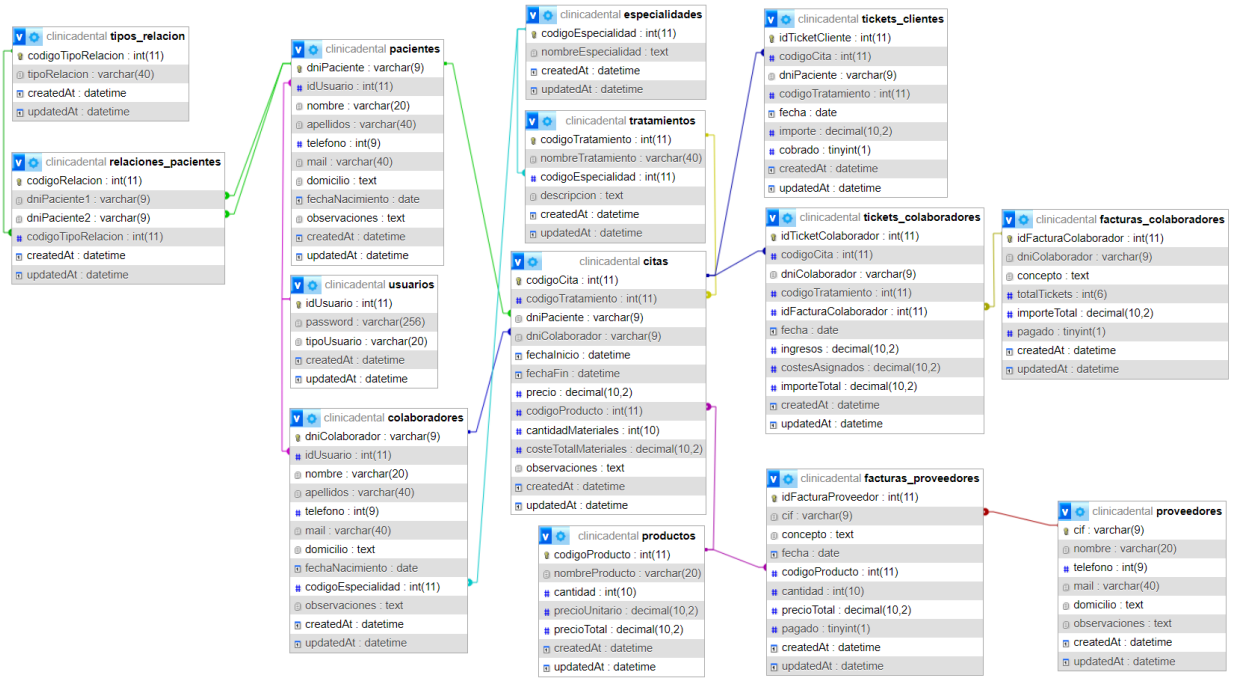


Figura 3.3—2. Diagrama entidad-relación de la base de datos

Debido al gran número de relaciones existentes entre las 14 tablas de la base de datos, en el “Apéndice A - Detalle de las tablas de la base de datos”, se indica una breve descripción del contenido de cada tabla, se explican de manera detallada las relaciones entre las distintas tablas y las claves primarias y foráneas correspondientes, para una mejor comprensión.

3.3.3 Reglas de integridad referencial

Para cada una de las relaciones creadas, con el objetivo de mantener la consistencia en la información de la base de datos, es necesario configurar qué acción se realizará sobre las claves foráneas de las tablas secundarias, en caso de modificar una clave primaria en la tabla principal.

En esta base de datos, todas las relaciones se implementan siguiendo la misma estructura:

- En el caso de “**on update**” la actualización de las claves externas en las tablas secundarias se realiza **en cascada.** Esto significa que si se modifica el valor de la clave

primaria, este cambio se verá reflejado automáticamente a todas las tablas secundarias a las que hace referencia.

- En el caso de "on delete" la actualización de las claves externas en las tablas secundarias que se realiza es **ninguna acción.** Esta opción se elige con el objetivo de mantener la integridad de los datos almacenados. Cuando se intente eliminar un registro en una tabla principal, si existen registros relacionados en tablas secundarias, esta acción no se podrá llevar a cabo. Con esta restricción tan estricta se evitan posibles eliminaciones de datos de manera accidental.

  Por ejemplo, no es interesante que se elimine un paciente de la base de datos y que, como consecuencia, se eliminen todas las citas relacionadas con dicho paciente, ya que es información que interesa almacenar, por ejemplo para hacer análisis de datos. Esta opción de "ninguna acción" en el caso "on delete" garantiza que todas las citas asociadas a un paciente no se eliminen al intentar eliminar al paciente de la base de datos.

3.4 Diseño de las pantallas

Antes de comenzar con el desarrollo de la aplicación web se realizó un diseño de las pantallas que formarían parte de la aplicación. Con el objetivo de estructurar cada uno de los elementos que se incluirían en la aplicación y conseguir una eficiencia durante el proceso de desarrollo al tener una guía clara que poder seguir.

A continuación se exponen los bocetos de algunas de las pantallas principales a modo de ejemplo:

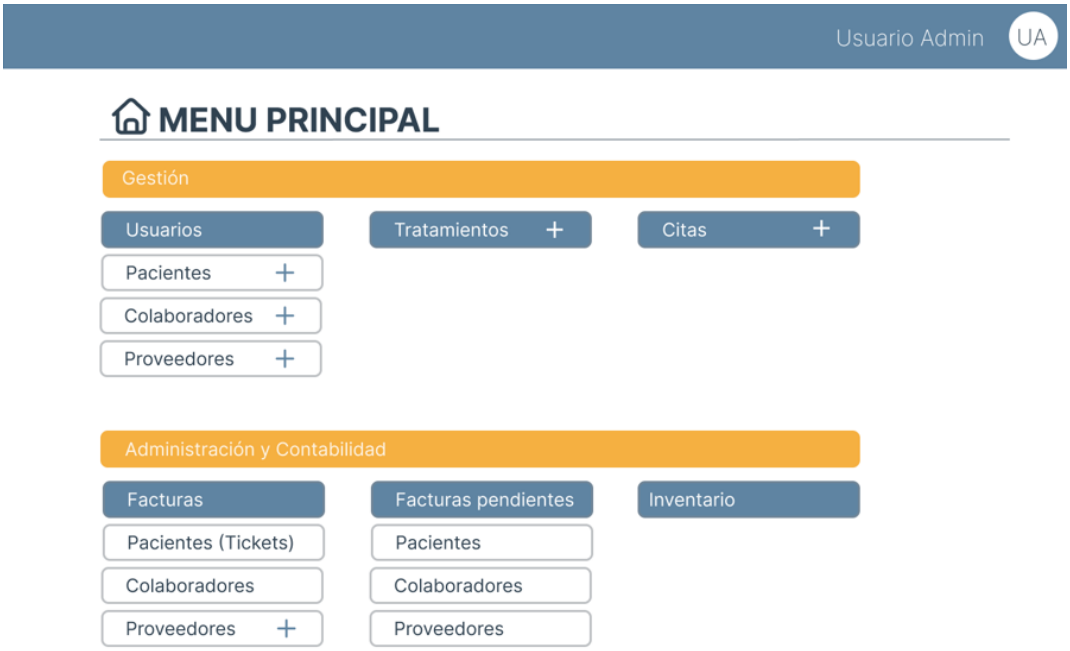


Figura 3.4—1. Boceto de la pantalla de menú principal del usuario administrador

Figura 3.4—2. . Boceto de la pantalla del listado completo de pacientes incluidos en la aplicación

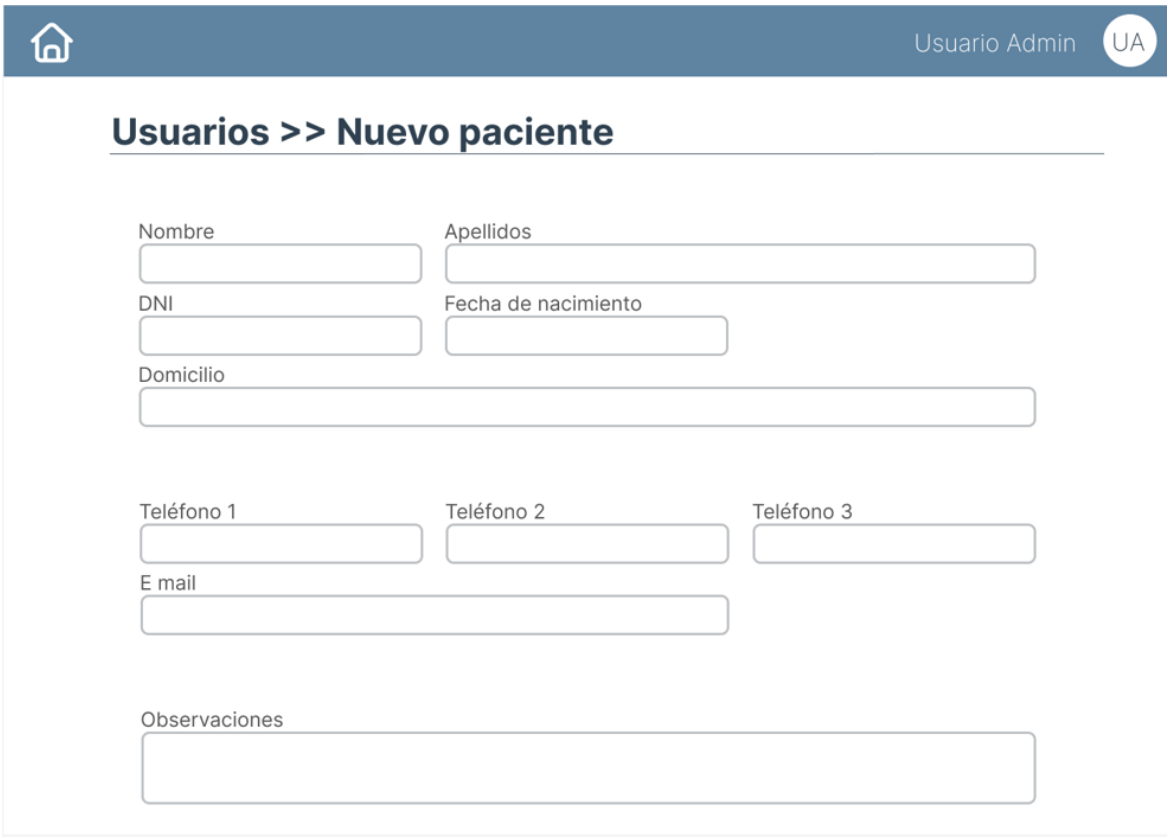


Figura 3.4—3. . Boceto de la pantalla del formulario para añadir un nuevo paciente

3.5 Casos de uso

Para especificar los requisitos funcionales para la creación de la aplicación web se han desarrollado distintos casos de uso que incluyen las distintas acciones que se podrán realizar en la aplicación.

A continuación se exponen cuatro diagramas de casos de uso, que representan de forma gráfica varios casos de uso agrupados por categorías: gestión de usuarios, de citas, de inventario y de facturación.

Después de cada diagrama se describirán únicamente los casos de uso más importantes y que tienen más detalle, necesarios para comprender como funciona la aplicación web. El resto de los casos de uso que son más sencillos están incluidos en el “Apéndice B - Detalle de los casos de uso”.

3.5.1 Diagrama de casos de uso 1: Gestión de usuarios

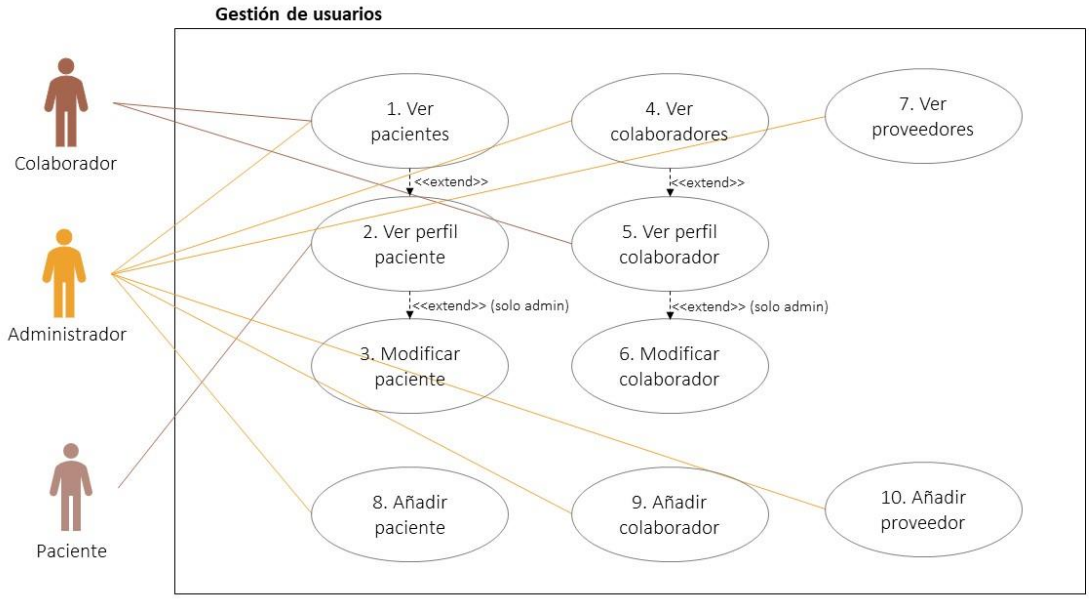


Figura 3.5—1. Diagrama de casos de uso 1: Gestión de usuarios

3.5.1.1 Caso de uso 8: Añadir paciente

Caso de uso 8	Añadir paciente
Objetivo	Permite añadir un nuevo paciente a la aplicación y crear su correspondiente usuario para acceder a su perfil en la aplicación.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin”. La conexión con la base de datos está establecida.

Caso de uso 8	Añadir paciente
Secuencia	1. El usuario accede a la opción “Añadir paciente” desde el menú principal. 2. El sistema solicita los campos para añadir un nuevo paciente (dni, nombre, apellidos, contraseña, teléfono, email, domicilio, fecha de nacimiento y observaciones). 3. El usuario rellena los campos y pulsa el botón “Añadir”. 4. El sistema realiza una consulta a la base de datos para crear un nuevo usuario con las siguientes credenciales: un id generado automáticamente y la contraseña proporcionada en el formulario. 5. El sistema realiza una consulta a la base de datos para crear un nuevo paciente con el id de usuario y el resto de los campos proporcionados en el formulario. 6. El sistema muestra un segundo formulario para añadir las posibles relaciones (padre, hijo, hermano, …) del paciente creado con otros pacientes que ya existen en la aplicación. 7. El usuario añade las posibles relaciones existentes y pulsa “Añadir” 8. El sistema realiza una consulta a la base de datos para añadir las relaciones con otros pacientes

Caso de uso 8	Añadir paciente
Postcondición	Se ha añadido un nuevo paciente a la base de datos y se ha creado un usuario para que el paciente pueda acceder a la aplicación con sus propias credenciales.

Tabla 3.5—1. Caso de uso 8: Añadir paciente

3.5.1.2 Caso de uso 9: Añadir colaborador

Este caso de uso es similar al "Caso de uso 8: Añadir paciente", a continuación se indican únicamente las diferencias:

- Objetivo: Permite añadir un nuevo colaborador a la aplicación y crear su correspondiente usuario para acceder a su perfil en la aplicación.
- Secuencia: El procedimiento es el mismo pero solo hasta el punto 5 (incluido), y se realiza desde la opción "Añadir colaborador" del menú principal. Los campos solicitados para añadir un nuevo colaborador son los mismos añadiendo la especialidad.

3.5.1.3 Caso de uso 10: Añadir proveedor

Este caso de uso es similar al "Caso de uso 8: Añadir paciente", a continuación se indican únicamente las diferencias:

- Objetivo: Permite añadir un nuevo proveedor a la aplicación, pero en este caso NO se crea un usuario para acceder a la aplicación
- Secuencia: El procedimiento es el mismo, pero solo los pasos 1-3, 5, y se realiza desde la opción "Añadir proveedor" del menú principal. Los campos solicitados para añadir un nuevo colaborador son: cif, nombre, teléfono, email, domicilio y observaciones.

3.5.2 Diagrama de casos de uso 2: Gestión de citas

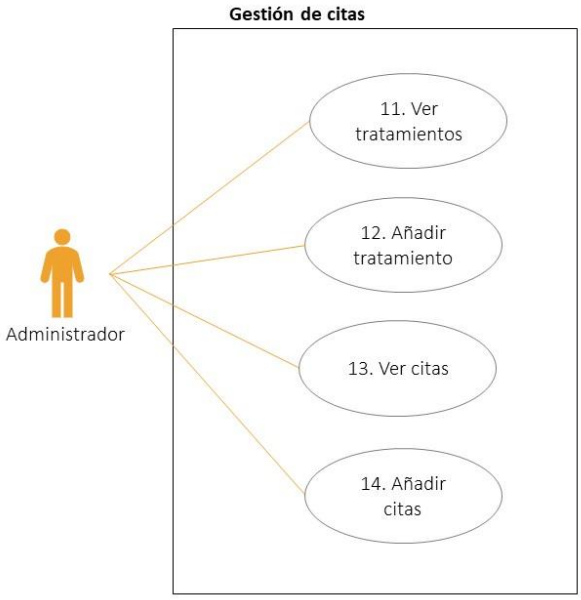


Figura 3.5—2. Diagrama de casos de uso 2: Gestión de citas

3.5.2.1 Caso de uso 14: Añadir cita

Caso de uso 14	Añadir cita
Objetivo	Permite añadir una nueva cita a la aplicación. Además se genera un ticket para el paciente asociado a la cita, para llevar un control de cobros. También se actualizan las existencias en el inventario de los productos que se consumen en la cita.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin”. La conexión con la base de datos está establecida.

Caso de uso 14	Añadir cita
Secuencia	1. El usuario accede a la opción “Añadir cita” desde el menú principal. 2. El sistema solicita los campos para añadir una nueva cita (tratamiento, paciente, colaborador, fecha y hora de inicio y fin, precio, observaciones, producto y cantidad de producto que se consumirán en la cita). 3. El usuario rellena los campos y pulsa el botón “Añadir”. 4. El sistema realiza una consulta a la base de datos para actualizar las existencias del inventario del producto consumido y obtiene el coste unitario de dicho producto para calcular los costes asociados a la cita (Coste Unitario x Cantidad Consumida). 5. El sistema realiza una consulta a la base de datos para crear una nueva cita con los campos proporcionados en el formulario y con el coste asociado, calculado a partir de la consulta anterior. 6. El sistema realiza una consulta a la base de datos para crear un nuevo ticket de paciente asociado a la cita con el importe que deberá ingresar el paciente tras realizarse el tratamiento (para llevar un control de cobros).

Caso de uso 14	Añadir cita
Postcondición	Se ha añadido una nueva cita a la base de datos, que se puede visualizar en el calendario. También se han actualizado las existencias en el inventario del producto consumido. Y se ha generado un nuevo ticket de paciente asociado a la cita creada.

Tabla 3.5—2. Caso de uso 14: Añadir cita

3.5.3 Diagrama de casos de uso 3: Gestión de inventario

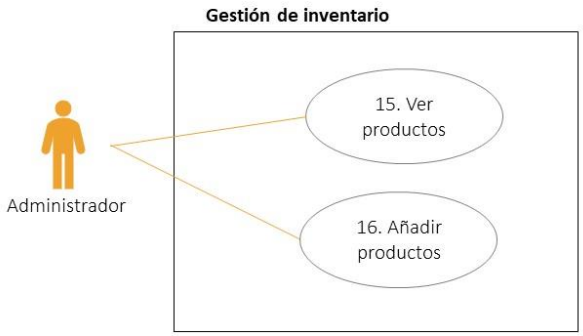


Figura 3.5—3. Diagrama de casos de uso 3: Gestión de inventario

Nota: En los casos de uso de esta sección "Gestión de inventario" solo se incluyen las acciones de "Ver los productos" y de "Añadir un nuevo producto" a la base de datos (con existencias iniciales 0). Las variaciones de existencias de inventario se producen en otros casos de uso:

- Las salidas de materiales en el "Caso de uso 14: Añadir cita".
- Las entradas de materiales en el "Caso de uso 24: Añadir factura proveedor".

3.5.4 Diagrama de casos de uso 4: Gestión de facturación

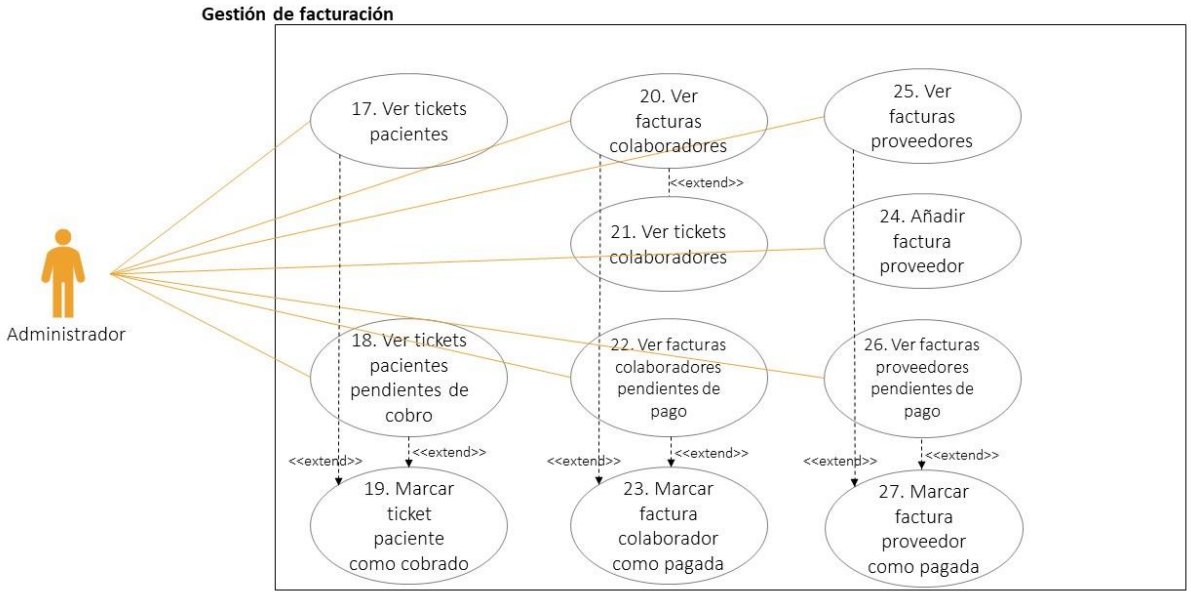


Figura 3.5—4. Diagrama de casos de uso 4: Gestión de facturación

3.5.4.1 Caso de uso 19: Marcar ticket paciente como cobrado

Caso de uso 19	Marcar ticket paciente como cobrado
Objetivo	Permite marcar un ticket de paciente como cobrado cuando se abona el importe del tratamiento realizado. Además, una vez se paga un ticket de paciente, se genera un ticket de colaborador, para contabilizar los ingresos correspondientes a ese dentista por el tratamiento realizado y se incluye en la factura del colaborador y mes correspondientes.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin”. La conexión con la base de datos está establecida.

Caso de uso 19	Marcar ticket paciente como cobrado
	El usuario se encuentra visualizando el listado de tickets (listado completo o listado de tickets pendientes de cobro).
Secuencia	1. El usuario marca un checkbox vacío correspondiente a un ticket concreto. 2. El sistema realiza una consulta a la base de datos para actualizar el campo "cobrado" del ticket de paciente correspondiente. 3. El sistema realiza una consulta a la base de datos para obtener el id de la factura correspondiente al mes y colaborador que ha realizado la cita. En caso de que la factura no exista, el sistema realiza una consulta a la base de datos para crearla. 4. El sistema realiza una consulta a la base de datos para crear un ticket de colaborador con los campos correspondientes (dni colaborador, tratamiento, id de la factura correspondiente, fecha, ingresos, costes e importe total). Para el cálculo de los ingresos y los costes se utiliza el precio pagado por el cliente y los costes de materiales, respectivamente, y dividido entre dos (la mitad del beneficio corresponde al colaborador y la otra mitad al negocio de la clínica). 5. El sistema actualiza la factura del mes y colaborador correspondiente, añadiendo 1 al campo "total tickets" y actualizando el valor de "importe total" sumando el valor del importe del

Caso de uso 19	Marcar ticket paciente como cobrado
	ticket creado. El campo “pagado” se inicializa en false.
Postcondición	Se ha actualizado el campo “cobrado” del ticket. Se ha generado el ticket del colaborador correspondiente que refleja los ingresos que le corresponden a ese dentista por el tratamiento realizado. También se ha incluido el ticket del colaborador en la factura correspondiente a ese mes y colaborador.

Tabla 3.5—3. Caso de uso 19: Marcar ticket paciente como cobrado

3.5.4.2 Caso de uso 24: Añadir factura proveedor

Caso de uso 24	Añadir factura proveedor
Objetivo	Permite añadir una nueva factura de un proveedor concreto especificando que producto se ha comprado, la cantidad y el importe. Con los datos de la factura actualizar en el inventario las existencias del material adquirido.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin”. La conexión con la base de datos está establecida.
Secuencia	1. El usuario accede a la opción “Añadir factura proveedor” desde el menú principal. 2. El sistema solicita los campos para añadir una nueva factura de proveedor (cif, concepto, fecha,

Caso de uso 24	Añadir factura proveedor
	producto comprado, cantidad, precio total y la opción de marcar como pagada o no). 3. El usuario rellena los campos y pulsa el botón "Añadir". 4. El sistema realiza una consulta a la base de datos para crear una nueva factura de proveedores con los campos proporcionados en el formulario. 5. El sistema realiza una consulta a la base de datos para actualizar las existencias del inventario del producto adquirido. Calcula el valor unitario actualizado del producto en el inventario con la técnica del PMP (explicado en el apartado "3.1.3.1 Entrada de existencias en el inventario").
Postcondición	Se ha creado una nueva factura de proveedor. Y se han actualizado las existencias en el inventario del nuevo producto comprado.

Tabla 3.5—4. Caso de uso 24: Añadir factura proveedor

3.6 Arquitectura y despliegue

En este apartado, se explicará la estructura del proyecto utilizada para el desarrollo de la aplicación web tanto para la parte del front-end y como del back-end. Indicando la organización de archivos y carpetas en cada una de las partes.

Además, en el apartado de la explicación del desarrollo del front-end se añadirán las pantallas correspondientes a cada parte de la aplicación

Con esta explicación se pretende lograr una comprensión clara sobre la arquitectura, el desarrollo y funcionamiento del proyecto.

3.6.1 Estructura del proyecto e implementación del front-end

Para el desarrollo del lado del cliente se ha utilizado la siguiente estructura de carpetas y ficheros:

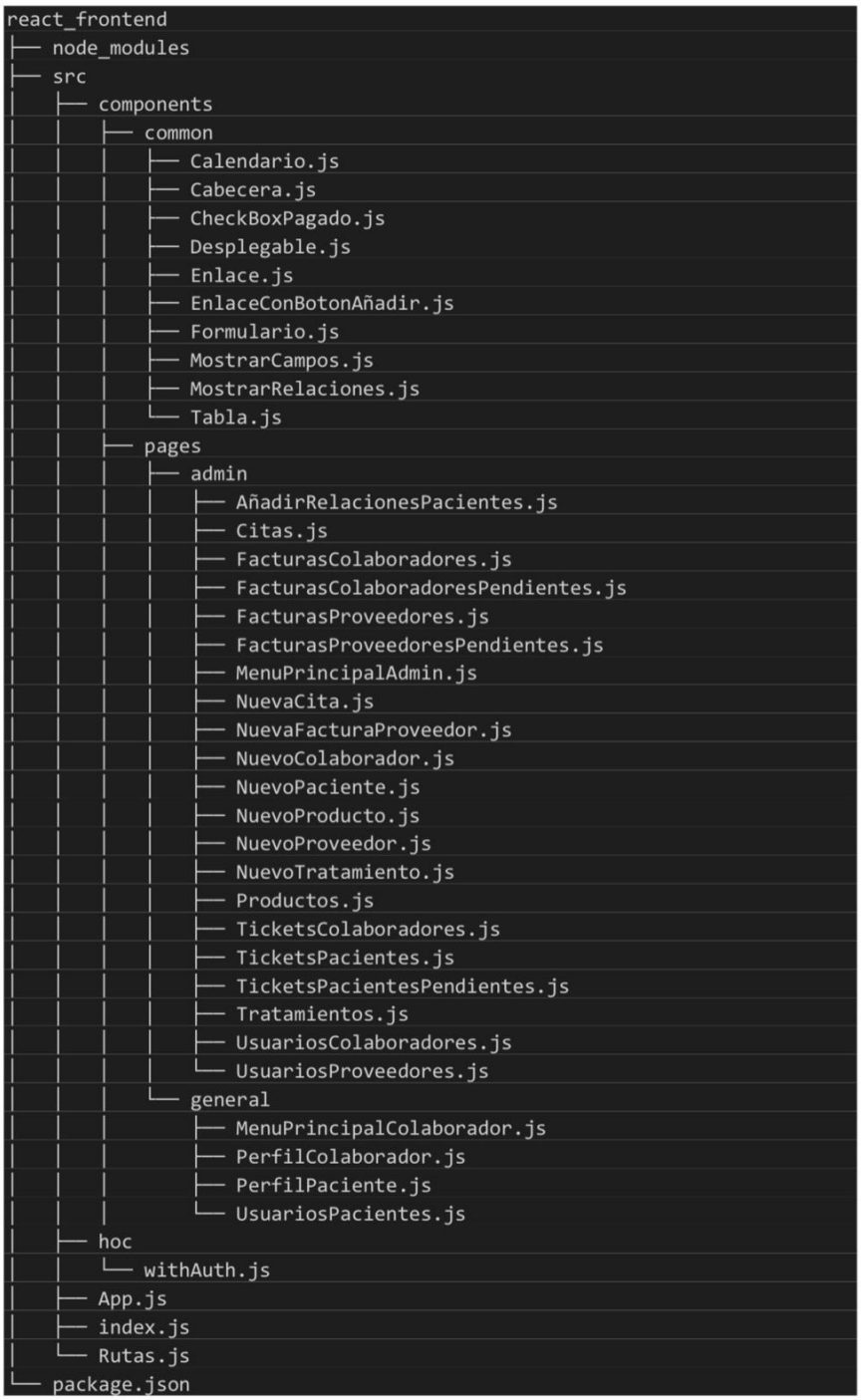


Figura 3.6—1. Diagrama de árbol de las carpetas y archivos utilizados para el front-end

A continuación se explicará el proceso de implementación y la utilidad de cada una de las principales carpetas y ficheros por niveles en la jerarquía:

- Primer nivel:
    - node_modules: Esta carpeta se crea en el directorio raíz, cuando se inicia el proyecto de React, en ella se incluyen todas las bibliotecas y paquetes necesarios para el correcto funcionamiento de la aplicación web (Manz, s.f.).
    - src: En esta carpeta se encuentran todos los ficheros de código implementados para el funcionamiento de la aplicación.
    - packaje.json: Es un fichero que se genera al iniciar un proyecto, en él se detalla automáticamente toda la información que se conoce sobre dicho proyecto como, por ejemplo, el nombre del proyecto o todas las dependencias instaladas.
- Segundo nivel:
    - components: En esta carpeta se incluyen todos los componentes creados que se utilizan para crear la aplicación de React. Para entender que es un componente consultar el apartado "3.2.1.1 React".
    - hoc: En esta carpeta se incluyen unos componentes concretos que existen en React que se llaman Componentes de Orden Superior (HOC). Según la página oficial de React "un componente de orden superior es una función que recibe un componente y devuelve un nuevo componente" (Meta Open Source, s.f.).
    - App.js: Es el componente principal de la aplicación de react, en este fichero se desarrolla la estructura general de la aplicación.
    - index.js: En este archivo se renderiza el componente App.js en el DOM.
    - Rutas.js: En este archivo se declaran todas las rutas utilizadas para que sea posible la navegación entre las distintas páginas de la aplicación, utilizando la biblioteca "react-router-dom".

- Tercer nivel:
    - common: En esta carpeta se incluyen los componentes genéricos que se reutilizan varias veces a lo largo del desarrollo de la aplicación. Por ejemplo, el componente Tabla.js.
    - pages: En esta carpeta se incluyen los componentes que representan las páginas específicas que forman la aplicación, y están compuestos por otros componentes genéricos.
    - withAuth.js: En este archivo se incluye un HOC que se encarga de comprobar que el resto de los componentes, que se utilizan para crear la interfaz, sean accedidos por usuarios que tengan los permisos necesarios en cada caso concreto. El listado del tipo de usuarios que tienen autorización se pasa al componente withAuth.js por "props".

      Para ello, este componente decodifica el token que está almacenado en "localStorage", utilizando la biblioteca "jwt-decode". Al decodificar el token se obtiene el tipo de usuario que ha iniciado sesión. Y pueden ocurrir dos cosas:

        a) Si el token no existe o el tipo de usuario no pertenece al listado proporcionado, withAuth.js devuelve el componente Login.js, para que el usuario inicie sesión.

        b) En caso contrario, si el tipo de usuario que ha iniciado sesión coincide con el listado de tipos de usuario autorizados, se devuelve el componente de la página que se estuviera intentando cargar.

      Por ejemplo, el componente PerfilColaborador.js solo puede ser accedido por usuarios de tipo "admin" o "colaborador", este listado se pasa por "props" a withAuth.js, y este HOC se encarga de verificar el tipo de usuario antes de cargar el componente PerfilColaborador.js:

```
156   export default withAuth(["admin", "colaborador"])(PerfilColaborador);
```

Figura 3.6—2. Código explicativo de llamada a un HOC en React

- Cuarto nivel. En este nivel se explican todos los componentes incluidos en la carpeta "common", para la explicación de los componentes se incluirá una breve descripción y se explicarán las "props" más relevantes. También se explican en este nivel las dos carpetas que forman la carpeta de componentes "pages":

    - Calendario.js: Es un componente genérico que muestra un calendario con las citas. Dependiendo de la página donde haya sido llamado se muestran todas las citas (opción "Ver citas" del menú principal), las citas de un paciente concreto (desde el perfil del paciente) o las citas que tiene que atender un determinado colaborador (desde el perfil del colaborador).

    - CheckboxPagado.js: Es un componente que se incluye dentro del componente Tabla.js, se utiliza para mostrar un checkbox que indica si una factura o un ticket ha sido pagado. Este componente también controla si el checkbox se marca, para indicar que un ticket o factura ha sido pagado y realiza las correspondientes consultas a la base de datos.

      Este componente recibe la siguiente "prop":

        - editable: especifica si el checkbox puede ser editado o debe tener únicamente permisos de visualización. De esta forma únicamente el administrador puede marcar un ticket o una factura como pagados y los usuarios con perfil "paciente" o "colaborador" pueden ver la situación de sus tickets o facturas, respectivamente, pero sin realizar ninguna modificación.

      Un ejemplo del componente Checkbox.js, incluido en el componente Tabla.js que muestra las facturas de los proveedores:

Figura 3.6—3. Ejemplo del componente Checkobox.js

- **Desplegable.js:** Es un componente que se incluye dentro del componente Formulario.js, se utiliza para mostrar un desplegable en un formulario.

  Para la personalización de este componente se reciben las siguientes "props":

  - tabla y columna: para realizar la consulta a la base de datos sobre la tabla y columna indicadas y mostrar en las opciones del desplegable una información específica.

  Por ejemplo, para añadir un nuevo colaborador a la base de datos hay que elegir en el formulario la especialidad de ese dentista. Para seleccionar la especialidad se muestra un desplegable, donde las opciones son las especialidades registradas en la base de datos en la "¡Error! No se encuentra el origen de la referencia.":

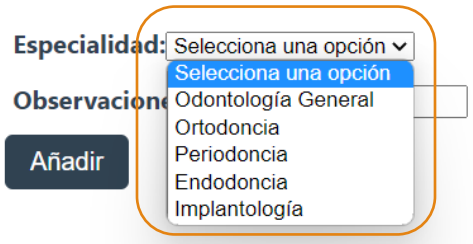


Figura 3.6—4. Ejemplo del componente Desplegable.js

- **Enlace.js**: Es un componente que muestra un texto como un enlace. Se utiliza principalmente en el componente MenuPrincipalAdmin.js y en el componente Tabla.js, cuando se quiere poner uno de los campos como un enlace.

  Para la personalización de este componente se reciben las siguientes "props":

- ❖ nombre: indica el valor que debe tener el texto del enlace
- ❖ enlace: indica la ruta a la que se tiene que llegar al pinchar en el enlace

Por ejemplo, en el caso de la tabla que muestra a los pacientes, cada valor en la columna "Nombre" es un enlace que lleva al perfil del paciente seleccionado. Este enlace en el nombre de cada paciente esta creado a partir del componente Enlace.js:

Figura 3.6—5. Ejemplo del componente Enlace.js

- **EnlaceConBotonAñadir.js:** Es un componente formado por dos componentes Enlace.js. Se utiliza en el menú principal del administrador para los casos en los que hay que mostrar unos datos pero también existe la posibilidad de añadir un dato nuevo a la base de datos.

  Por ejemplo, la opción "Ver pacientes" y "Añadir Pacientes", se representa con este componente, donde la opción "Ver pacientes" aparece como un enlace escrito y la opción "Añadir Pacientes" aparece como un enlace incluido en el botón de añadir (+):

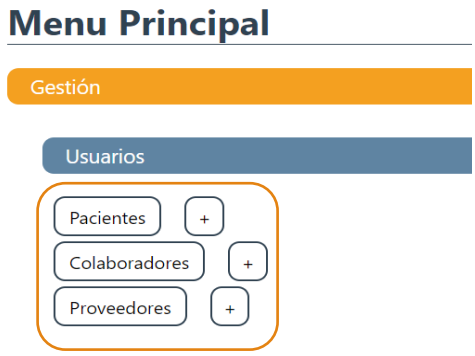


Figura 3.6—6. Ejemplo del componente EnlaceConBotonAñadir.js

- **Formulario.js:** Este componente muestra un formulario a partir de unos campos que recibe como props en forma de array.
- **MostrarCampos.js:** Este componente muestra un listado de los datos correspondientes a un paciente o un colaborador concreto en su perfil de usuario.

  Para la personalización del componente se reciben las siguientes "props":

  - tabla: indica la tabla de la base de datos donde realizar la consulta para obtener los datos que mostrar.
  - campos: es un array, como en el componente Formulario.js, que indica los campos concretos de la consulta realizada que se tienen que mostrar en el perfil del usuario, especificando también el título de cada uno de estos.
  - editar: indica si los campos mostrados pueden ser editados por el usuario o no. El valor de editar es "true" cuando el usuario es de tipo "admin" y false en caso contrario.

  Un ejemplo del componente MostrarCampos.js, en el caso del usuario "admin" y en el caso de un usuario tipo "paciente":

Figura 3.6—7. Ejemplo del componente MostrarCampos.js con la opción editar (usuario "admin") y sin la opción editar (usuario "paciente")

- **MostrarRelaciones.js:** Este componente muestra las relaciones en forma de lista que tiene un determinado paciente con otros pacientes. Se utiliza en el componente PerfilPaciente.js, para mostrar las relaciones que tiene ese paciente con otros y en el componente AñadirRelacionesPaciente.js, para poder visualizar las relaciones que ya están añadidas a la base de datos para ese paciente.

- **Tabla.js**: Este componente muestra en formato de tabla una información determinada.

  Para la personalización del componente se reciben las siguientes "props":

  - tabla: indica la tabla de la base de datos donde realizar la consulta para obtener los datos que mostrar.

  - campos: es un array, como en el componente Formulario.js y el componente MostrarCampos.js, que indica los campos concretos de la consulta realizada que se tienen que mostrar en la tabla, especificando también el encabezado de cada uno de estos.

- **admin**: En esta carpeta se incluyen los componentes que son utilizados para crear las páginas a las que exclusivamente tiene acceso el usuario de tipo "admin".

- **general**: En esta carpeta se incluyen los componentes que son utilizados para generar las pantallas a las que tienen acceso usuarios de cualquier tipo: "admin", "paciente" o "colaborador"

- Quinto nivel. Se explicarán los componentes que forman las distintas páginas de la aplicación. Se incluirá una breve descripción con la correspondiente **imagen de cada pantalla** y se listarán los componentes genéricos que se combinan para crear cada una de estas.

  Nota: Debido al elevado número de componentes utilizados en la aplicación, únicamente se explicarán algunos de ellos.

En primer lugar se explicarán los componentes incluidos en la carpeta admin, a estos componentes solo tiene acceso el usuario de tipo “admin”:

- AñadirRelacionesPacientes.js: Este componente se utiliza después de haber añadido un nuevo paciente para añadir las posibles relaciones existentes de este con otro paciente que ya esté incluido en la base de datos.

  Está compuesto por el componente MostrarRelaciones.js, que muestra las relaciones ya establecidas y el componente Formulario.js que permite añadir nuevas relaciones.

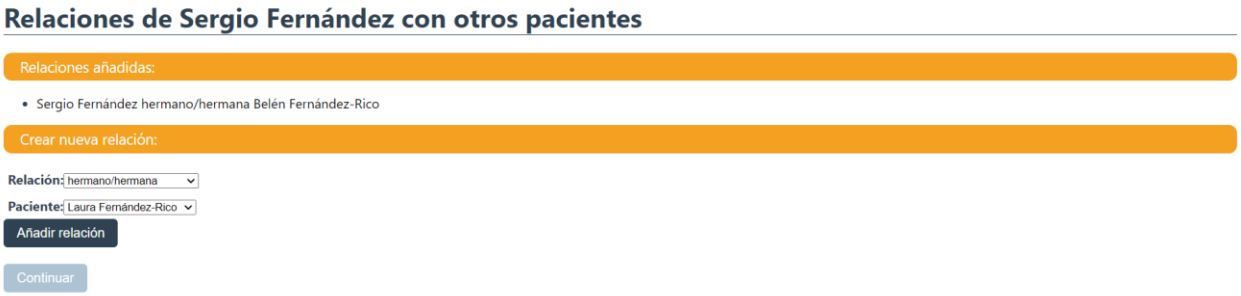


Figura 3.6—8. Pantalla "añadir relaciones paciente"

- **Citas.js:** Este componente se muestra al seleccionar la opción “Ver citas” desde el menú principal del administrador y permite ver todas las citas incluidas en la aplicación.

  Está compuesto por el componente Calendario.js.

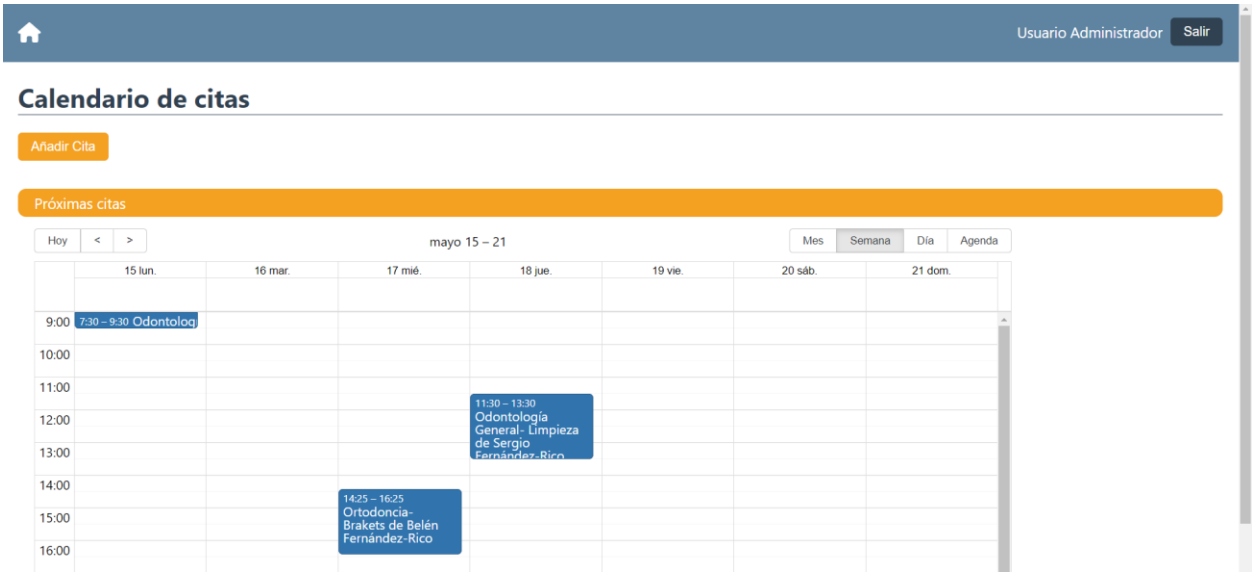


Figura 3.6—9. Pantalla "citas"

- **FacturasProveedores.js:** Este componente se muestra al seleccionar la opción “Facturas proveedores” desde el menú principal del administrador y permite ver todas las facturas de los proveedores incluidas en la aplicación.

  Está compuesto por el componente Tabla.js, que muestra todas las facturas incluidas en la tabla facturas proveedores.

Figura 3.6—10. Pantalla "facturas proveedores"

- FacturasProveedoresPendientes.js: Este componente se muestra al seleccionar la opción “Facturas proveedores pendientes” desde el menú principal del administrador y permite ver todas las facturas de los proveedores incluidas en la aplicación pendientes de pago.

  Está compuesto por el componente Tabla.js, que muestra todas las facturas incluidas en la tabla facturas proveedores, filtrando por el campo “pagado”.

Figura 3.6—10. Pantalla "facturas proveedores pendientes de pago"

- **MenuPrincipalAdmin.js:** Este componente es la primera pantalla que aparece cuando un usuario de tipo “admin” inicia sesión.

  Está compuesto por varios componentes Enlace.js y EnlaceConBotonAñadir.js, para mostrar cada una de las secciones que tiene la aplicación a las que puede acceder el administrador.

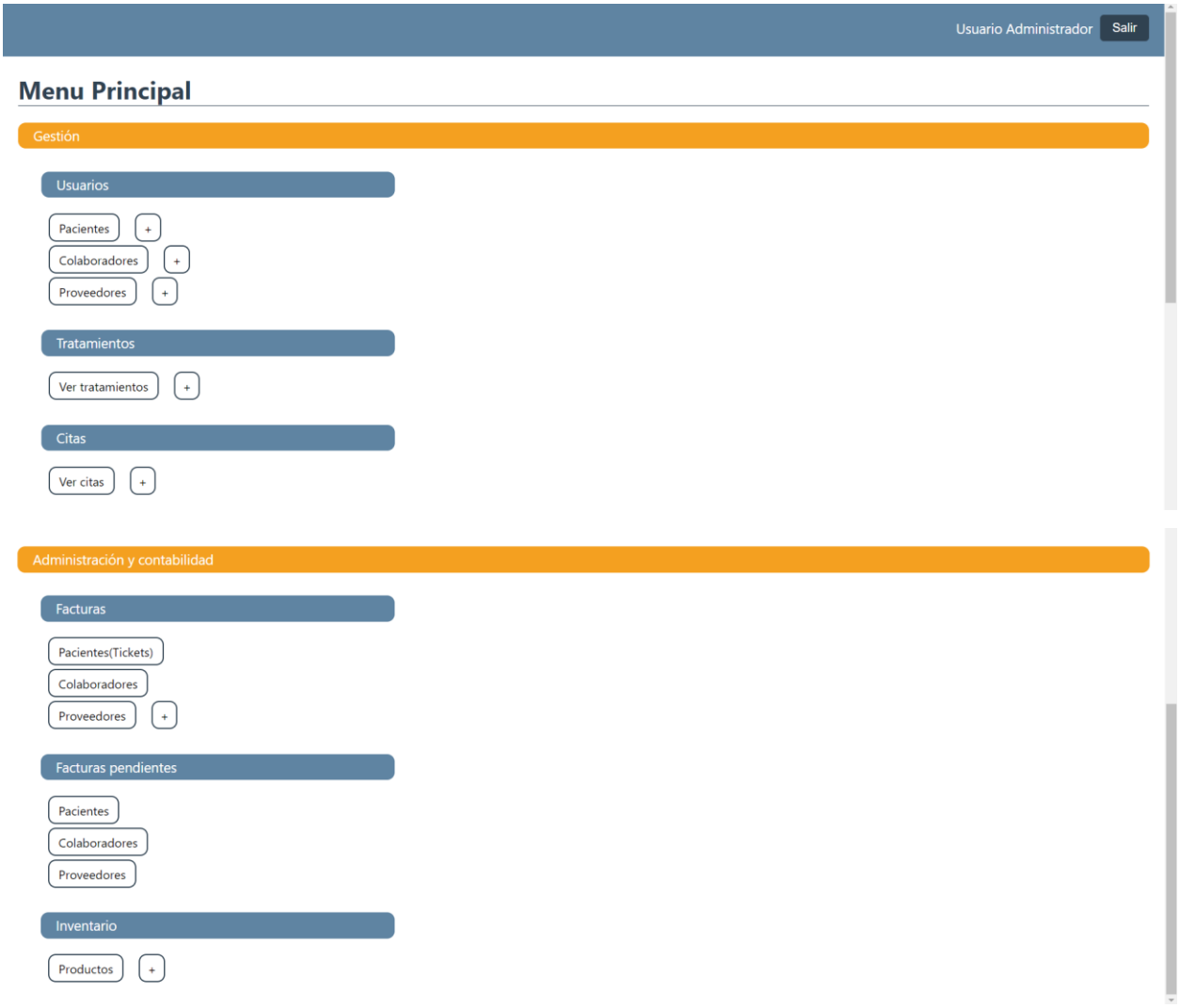


Figura 3.6—11. Pantalla "menu principal administrador"

- **NuevaCita.js:** Este componente se muestra al seleccionar la opción “Añadir cita” desde el menú principal del administrador y muestra un formulario para que el usuario lo rellene. Con los campos obtenidos del formulario realiza una consulta a la base de datos para que se añada una nueva cita a la tabla citas

  Está compuesto por el componente Formulario.js, que muestra los campos que se tendrán que rellenar para registrar una nueva cita. Estos campos son pasados como “props” al componente Formulario.js, desde el componente NuevaCita.js.

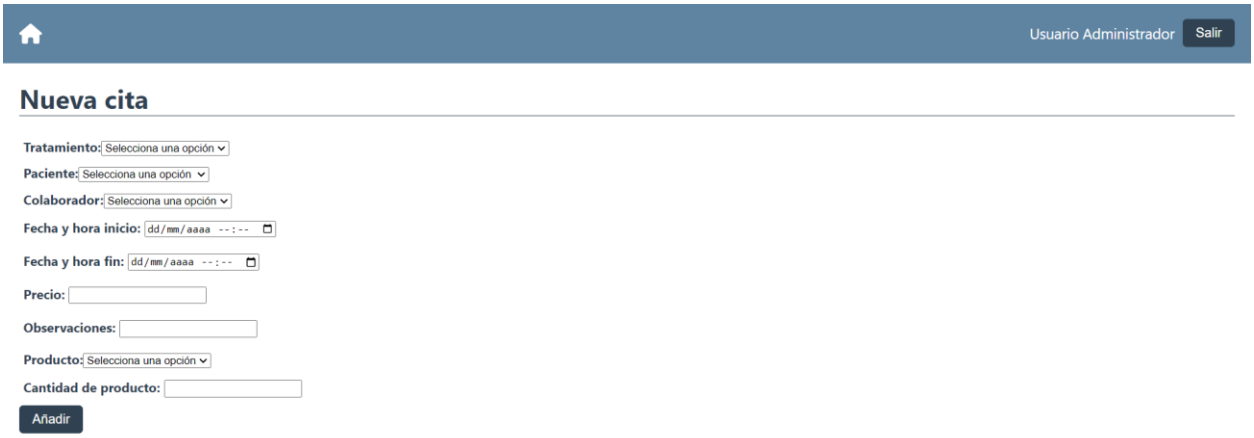


Figura 3.6—12. Pantalla "añadir citas"

- **Productos.js:** Este componente se muestra al seleccionar la opción “Productos” desde el menú principal del administrador y permite ver todos los productos del inventario incluidos en la aplicación, con sus correspondientes cantidades y precio unitario.

  Está compuesto por el componente Tabla.js, que muestra todos los productos incluidos en la “Tabla productos”.

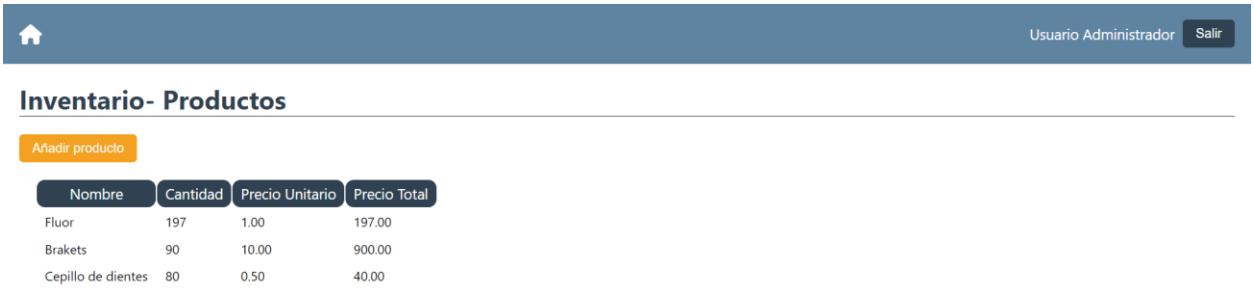


Figura 3.6—13. Pantalla "productos en el inventario"

- **TicketsPacientes.js:** Este componente se muestra al seleccionar la opción “Tickets pacientes” desde el menú principal del administrador y permite ver todas los tickets de los pacientes incluidos en la aplicación.

  Está compuesto por el componente Tabla.js, que muestra todos los tickets incluidos en la “Tabla Tickets Clientes”.

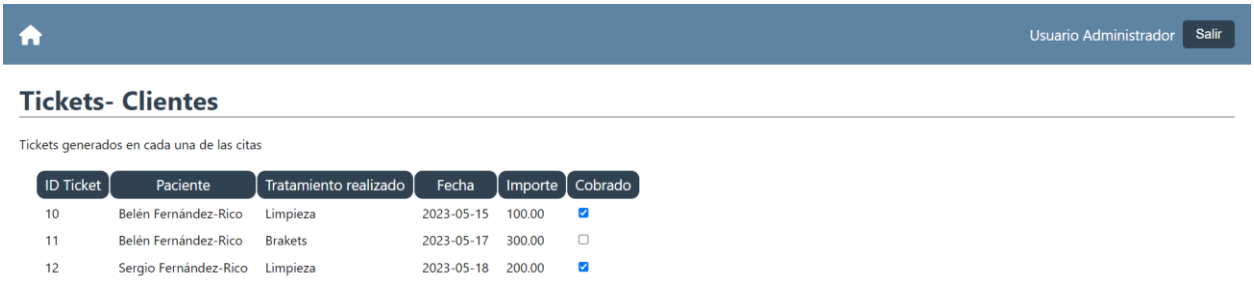


Figura 3.6—14. Pantalla "tickets clientes"

- **TicketsPacientesPendientes.js:** Este componente se muestra al seleccionar la opción “Tickets Pacientes Pendientes” desde el menú principal del administrador y permite ver todos los tickets de los pacientes incluidos en la aplicación pendientes de cobro.

  Está compuesto por el componente Tabla.js, que muestra todos los tickets incluidos en la “Tabla Tickets Clientes”, filtrando por el campo “cobrado”.

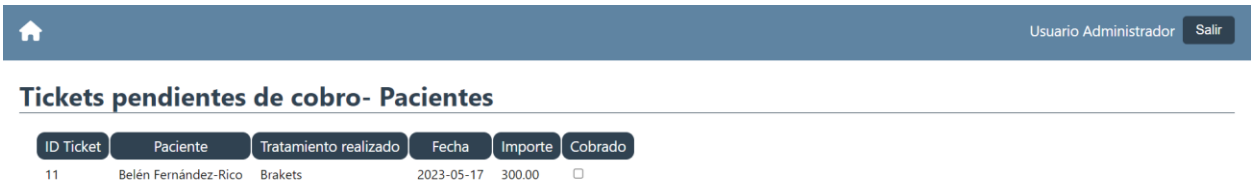


Figura 3.6—15. Pantalla "tickets pacientes pendientes de cobro"

- **UsuariosColaboradores.js:** Este componente se muestra al seleccionar la opción “Colaboradores” desde el menú principal del administrador y permite ver todos los colaboradores incluidos en la aplicación.

  Está compuesto por el componente Tabla.js, que muestra todos los colaboradores incluidos en la “Tabla Colaboradores”.

Figura 3.6—16. Pantalla "usuarios colaboradores"

- **UsuariosProveedores.js:** Este componente se muestra al seleccionar la opción "Proveedores" desde el menú principal del administrador y permite ver todos los proveedores incluidos en la aplicación.

  Está compuesto por el componente Tabla.js, que muestra todos los proveedores incluidos en la "Tabla Proveedores".

Figura 3.6—17. Pantalla "usuarios proveedores"

Por último se explican los componentes incluidos en la carpeta "general", que son aquellos componentes a los que acceden usuarios de cualquier tipo "admin", "paciente" o "colaborador".

- **PerfilColaborador:** Este componente se muestra al seleccionar un colaborador desde el listado completo de colaboradores al que tiene acceso el usuario de tipo "admin". Y en caso del usuario de tipo "colaborador" desde la opción "Ver mi perfil" del menú principal del colaborador. En este componente se muestra toda la información correspondiente al colaborador.

  Está compuesto por los componentes MostrarCampos.js, que muestra los datos personales del colaborador. En el caso del perfil del "admin" estos datos se pueden editar y en el caso del perfil "colaborador" los datos únicamente pueden visualizarse. También incluye el componente Calendario.js, que muestra las citas que tiene agendadas el colaborador correspondiente. Y por último el componente Tabla.js, que muestra todas las facturas correspondientes al colaborador incluidas en la tabla facturas colaborador filtrando por el campo "dniColaborador".

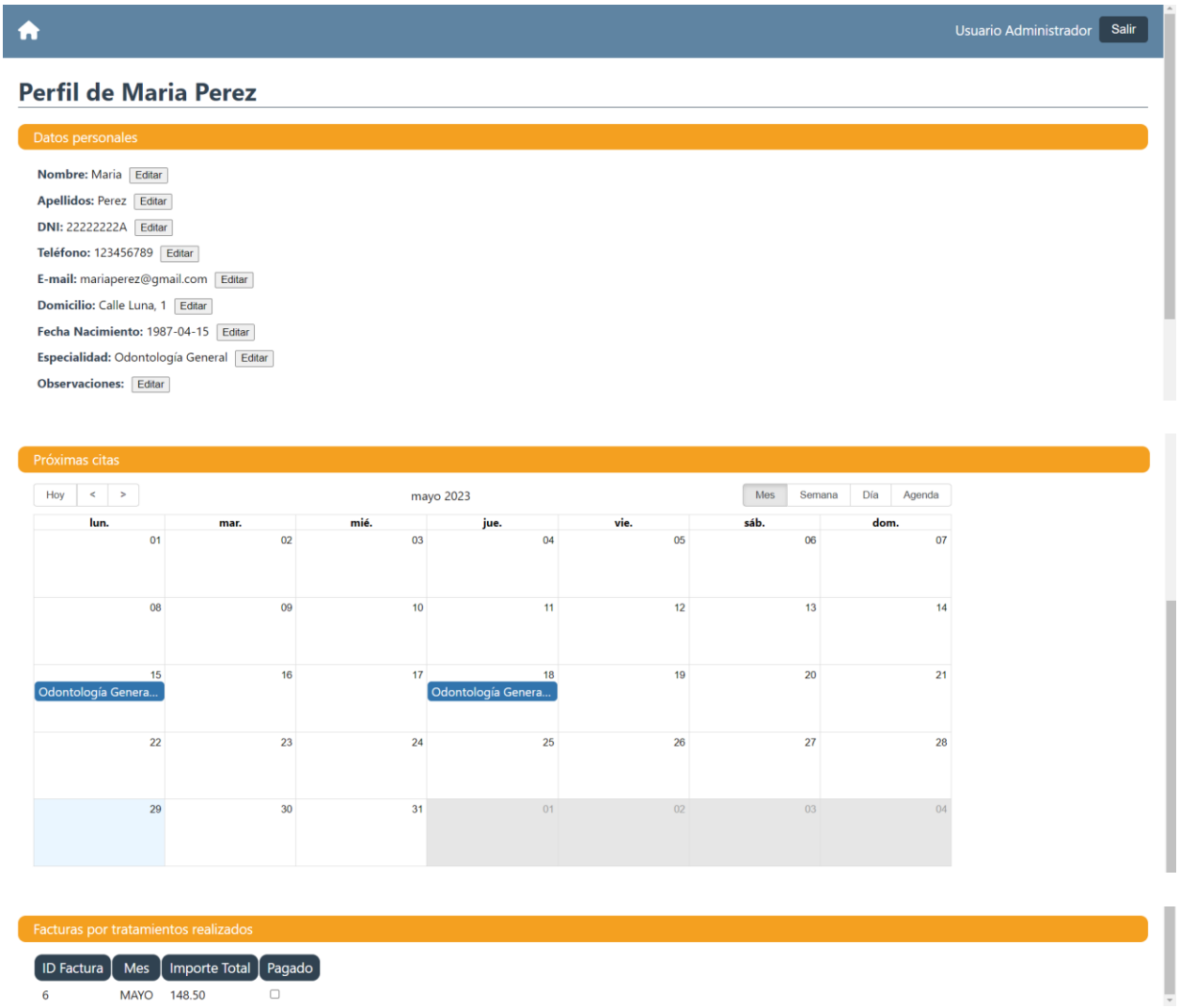


Figura 3.6—18. Pantalla "Perfil colaborador"

- **PerfilPaciente:** Este componente se muestra al seleccionar un paciente desde el listado completo de colaboradores al que tiene acceso el usuario de tipo "admin" y "colaborador". Y en el caso del usuario de tipo "paciente" se muestra el perfil al iniciar sesión en la aplicación.

  Está compuesto por los componentes MostrarCampos.js, que muestra los datos personales del paciente. En el caso del perfil del "admin" estos datos se pueden editar y en el caso del perfil "colaborador" y "paciente" los datos únicamente pueden visualizarse. El componente MostrarRelaciones.js, que refleja las relaciones del paciente con otros. También incluye el componente

Calendario.js, que muestra las citas del paciente correspondiente. Y por último el componente Tabla.js, que muestra todos los tickets asociados a cada una de las citas a las que acude el paciente obtenidos de la tabla tickets pacientes, filtrando por el campo “dniPaciente”.

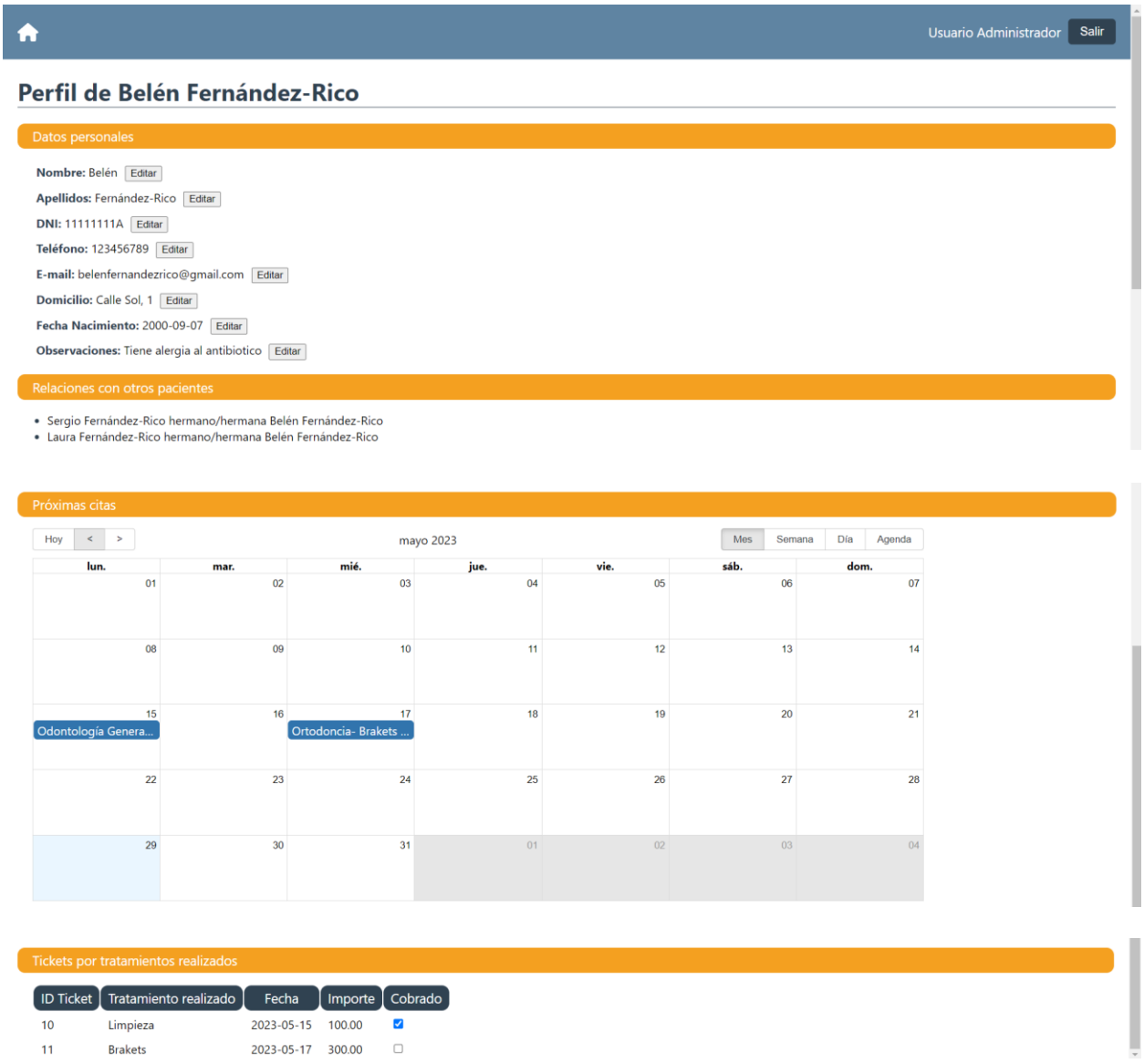


Figura 3.6—19. Pantalla "perfil paciente"

3.6.2 Estructura del proyecto e implementación del back-end

Para el desarrollo del lado del servidor se ha utilizado la siguiente estructura de carpetas y ficheros:

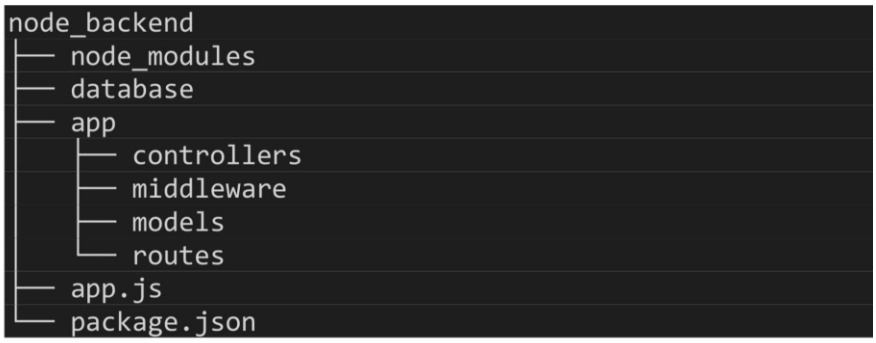


Figura 3.6—20. Diagrama de arbol de las carpetas y archivos utilizados para el back-end

- node_modules: Al igual que en la parte del front-end, esta carpeta se crea automáticamente y contiene las dependencias externas instaladas para el proyecto.
- database: En esta carpeta se incluyen los ficheros de configuración y conexión a la base de datos.
- app: Esta carpeta contiene la lógica de la aplicación.
    - controllers: En esta carpeta se incluye un fichero de controlador por cada tabla existente en la base de datos. Los controladores se encargan de realizar la lógica de la aplicación, para ello se reciben las solicitudes del cliente, se realizan las consultas correspondientes a la base de datos y se envían las respuestas de nuevo al lado del cliente.
    - middleware: En esta carpeta se incluye un middleware de autenticación. Que comprueba que la solicitud hecha desde el lado del cliente es realizada por un usuario autenticado en la aplicación, antes de enviar la respuesta de nuevo.
    - models: En esta carpeta se incluye un fichero de modelo por cada tabla existente en la base de datos. Los modelos definen la estructura de la tabla con

los campos, el tipo de cada uno de ellos y el nombre de la tabla. Un ejemplo de un modelo sería el siguiente, correspondiente a la tabla usuarios:

```
//Importamos la conexión a la Base de Datos
import db from "../../database/db.js";
import {DataTypes} from "sequelize";


const UsuarioModel = db.define("usuarios", {
  idUsuario: {
    type: DataTypes.INTEGER,
    primaryKey: true,
    autoIncrement: true
  },
  password: {
    type: DataTypes.STRING
  },
  tipoUsuario: {
    type: DataTypes.STRING
  },
  createdAt: {
    type: DataTypes.DATE
  },
  updatedAt: {
    type: DataTypes.DATE
  }
});

export default UsuarioModel;
```

Figura 3.6—21. Ejemplo de código de un fichero model en el back-end

- routes: En esta carpeta se incluye un fichero de rutas por cada tabla existente en la base de datos. En el fichero se incluyen las rutas que se puedan necesitar para realizar las llamadas al controlador, se han utilizado como máximo 4 rutas:

  a) "/obtener": Para obtener todos los registros de la tabla correspondiente

  b) "/obtenerPorId": Para obtener únicamente los registros identificados por un determinado id en la tabla correspondiente.

  c) "/crear": Para añadir un nuevo registro a la tabla correspondiente.

  d) "/actualizar": Para actualizar un registro de la tabla correspondiente.

  Un ejemplo de un fichero de rutas sería el siguiente, correspondiente a la tabla pacientes:

```
6   const PacienteRoutes = express.Router();
7
8   //Creamos las rutas para cada metodo del controlador
9   PacienteRoutes.get('/obtener', autenticacion, obtenerPacientes);
10  PacienteRoutes.post('/obtenerPorId', autenticacion, obtenerPacientePorId);
11  PacienteRoutes.post('/crear', autenticacion, crearPaciente);
12  PacienteRoutes.put('/actualizar', autenticacion, actualizarPaciente);
```

Figura 3.6—22. Ejemplo de un fichero de rutas en el back-end

- **app.js:** Es el fichero principal del proyecto del back-end donde se crea la conexión a la base de datos, utilizando los ficheros de la carpeta "database" y se definen las rutas para el manejo de las solicitudes:

```
34  //Configuración de rutas
35  app.use("/usuarios", UsuarioRoutes);
36  app.use("/pacientes", PacienteRoutes);
37  app.use("/colaboradores", ColaboradorRoutes);
38  app.use("/proveedores", ProveedorRoutes);
39  app.use("/tiposRelacion", TiposRelacionRoutes);
40  app.use("/relacionesPacientes", RelacionPacienteRoutes);
41  app.use("/especialidades", EspecialidadRoutes)
42  app.use("/citas", CitaRoutes);
43  app.use("/tratamientos", TratamientoRoutes);
44  app.use("/facturasProveedores", FacturaProveedorRoutes);
45  app.use("/ticketsClientes", TicketPacienteRoutes)
46  app.use("/facturasColaboradores", FacturaColaboradorRoutes)
47  app.use("/ticketsColaboradores", TicketColaboradorRoutes)
48  app.use("/productos", ProductoRoutes);
```

Figura 3.6—23. Rutas utilizadas para el manejo de solicitudes en el back-end

- **package.json:** Es un fichero que se genera al iniciar un proyecto, en él se detalla automáticamente toda la información que se conoce sobre dicho proyecto como por ejemplo el nombre del proyecto o todas las dependencias instaladas.

Capítulo 4 - Conclusiones y trabajo futuro

4.1.1 Conclusiones

La aplicación web para la gestión de una clínica dental se ha desarrollado con el objetivo de facilitar la administración de una clínica dental. Asimismo, se ha añadido como propuesta de innovación una personalización de la aplicación, dependiendo de cada tipo de usuario: Administrador, colaborador o paciente, para mejorar la calidad de atención.

Las funcionalidades implementadas: gestión de usuarios, de citas, de inventario y de facturación, ayudan con el control del negocio. Entre los beneficios que proporcionan estas funcionalidades se encuentran:

- En el perfil de cada paciente se incluyen las relaciones que tiene este paciente con otros que también acuden a la clínica. Así se puede ofrecer una atención personalizada a cada persona que acude a la clínica mostrando un interés por los pacientes y creando un ambiente familiar.
- El control de inventario digitalizado ayuda a la planificación de los pedidos que realizar a los proveedores para evitar que se produzcan roturas de stock.
- El control de la facturación permite llevar un control de gastos y conocer que pacientes no están abonando el importe de los tratamientos que se realizan.

Por otro lado, la personalización de la aplicación con la creación de perfiles individuales mejora la experiencia de los pacientes que acuden a la clínica. Y de igual manera, mejora el trato que pueden ofrecer los colaboradores pudiendo consultar desde su propio perfil el historial médico de los pacientes que tienen agendados.

Por último, el uso de tecnologías modernas como React y Node.js, ha permitido crear una aplicación rápida de desarrollar, escalable y fácil de mantener.

4.1.2 Trabajo futuro

La aplicación desarrollada cumple con el objetivo planteado en el apartado: "1.2 Objetivos" y ha aportado beneficios significativos a la clínica dental, mencionados en el apartado anterior. Sin embargo, existen posibilidades de desarrollo que podrían aportar un valor adicional a la aplicación. A continuación se exponen las áreas de mejora para el trabajo futuro:

- Añadir la posibilidad de enviar recordatorios de las citas de manera automática a los pacientes. También enviar promociones personalizadas a cada paciente a través de la aplicación web y que estos pudieran canjearlas desde su perfil.
- Añadir una opción de chat en la aplicación, con el objetivo de mejorar la comunicación entre la clínica dental y los pacientes. De esta forma, todo el flujo de información estaría centralizado en la aplicación y los pacientes podrían hablar directamente con sus dentistas en caso de sentir cualquier tipo de molestia.
- Desarrollar un nuevo módulo que utilizando técnicas de business intelligence, que pueda analizar los datos incluidos en la aplicación y ayude a la empresa a tomar decisiones a partir de estadísticas obtenidas como resultado del análisis de información.
- Por último, adaptar la aplicación web, para su uso en dispositivos móviles. Esta propuesta resulta muy interesante ya sería más accesible y cómoda para los usuarios, desde cualquier lugar.

Conclusions and future work

Conclusions

The web application for the management of a dental clinic has been developed with the aim of facilitating the administration of a dental clinic. Likewise, as a proposal for innovation, a personalisation of the application has been added, depending on each type of user: Administrator, collaborator or patient, in order to improve the quality of care.

The implemented functionalities: user, appointment, inventory and billing management, help with the control of the business. Among the benefits provided by these functionalities are the following:

- Each patient's profile includes the relationships that this patient has with others who also attend the clinic. This allows for personalised care to be offered to each person who comes to the clinic, showing an interest in the patients and creating a family atmosphere.
- The digitised inventory control helps to plan the orders to be placed with suppliers to avoid stock-outs.
- The invoicing control allows to keep track of expenses and to know which patients are not paying the amount of the treatments that are carried out.

On the other hand, the personalisation of the application with the creation of individual profiles improves the experience of patients who come to the clinic. Similarly, it improves the treatment offered by the collaborators, who can consult the medical history of the patients they have scheduled from their own profile.

Finally, the use of modern technologies such as React and Node.js has allowed us to create an application that is quick to develop, scalable and easy to maintain.

Future Work

The application developed meets the objective set out in section: "1.2 Objectives" and has brought significant benefits to the dental clinic, as mentioned in the previous section. However, there are possibilities for development that could bring additional value to the application. Areas for improvement for future work are outlined below:

- Adding the possibility to send appointment reminders automatically to patients. Also send personalised promotions to each patient through the web application and they could redeem them from their profile.
- Adding a chat option in the application, with the aim of improving communication between the dental clinic and patients. In this way, all the flow of information would be centralised in the application and patients would be able to talk directly to their dentists in case they feel any kind of discomfort.
- Develop a new module using business intelligence techniques that can analyse the data included in the application and help the company to make decisions based on statistics obtained as a result of the information analysis.
- Finally, adapt the web application for use on mobile devices. This proposal is very interesting as it would be more accessible and convenient for users, from anywhere.

BIBLIOGRAFÍA

Bookgy. (n.d.). *Software de Reservas y Gestión de clínicas Dentistas*. Retrieved from https://bookgy.com/es/software-gestion-reservas-online/dentistas/

Doctoralia Internet SL. (n.d.). *Software clínica Dental*. Retrieved from https://clinic-cloud.com/software-clinica-dental-programa-odontologico/

Flamarique, S. (2018). *Gestión de existencias en el almacen.* Marge Books.

Infomed Software Sanitario. (n.d.). *Gesden G5 Software de gestión dental*. Retrieved from https://www.grupoinfomed.es/software-sanitario/gesden-g5/

Manz. (n.d.). *NPM. Introducción a npm.* Retrieved from https://lenguajejs.com/npm/

Meta Open Source. (n.d.). *React. La biblioteca para interfaces de usuario web y nativas.* Retrieved from https://es.react.dev/

Microsoft. (n.d.). *VS Code. Code editing*. Retrieved from https://code.visualstudio.com/

Narayn, H. (2022). *Just React! Learn React the React Way.* Apress, Berkeley, CA.

npm, Inc. (n.d.). *npm. Search Packages*. Retrieved from https://www.npmjs.com/

OpenJS Foundation. (n.d.). *Acerca de Node.js*. Retrieved from https://nodejs.org/es/about

OrisLine. (n.d.). *Tu Partner de confianza para la gestión de la Clínica*. Retrieved from https://orisline.com/es/

phpMyAdmin. (n.d.). *Acerca de*. Retrieved from https://www.phpmyadmin.net/

APÉNDICES

Apéndice A - Detalle de las tablas de la base de datos

En esta sección se incluye una breve descripción de cada tabla y el detalle de las relaciones existentes, especificando en cada caso las claves primarias y foráneas utilizadas.

Tabla usuarios

En la tabla usuarios se recogen las credenciales, con las que los usuarios acceden a la aplicación (ID usuario y contraseña cifrada). También se incluye el tipo de usuario correspondiente (admin, paciente o colaborador).

La clave primaria en la tabla "usuarios" es "idUsuario". Esta tabla se relaciona con las tablas "pacientes" y "colaboradores", ya que son los dos tipos de usuarios que pueden acceder a la aplicación además del administrador.

- En la tabla "pacientes" la clave foránea es "idUsuario", para poder identificar a que paciente corresponde cada usuario.
- En la tabla "colaboradores" la clave foránea es "idUsuarios", para poder identificar a que colaborador corresponde cada usuario.

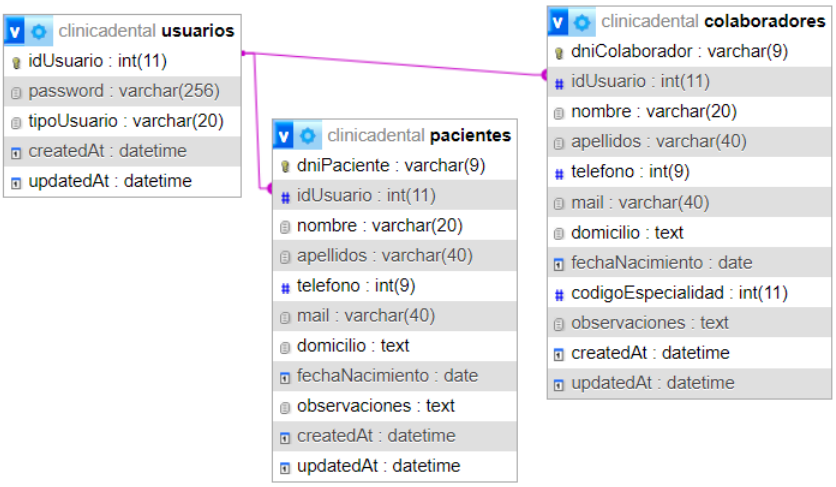


Figura Apéndice A—3.6—1. Relaciones de la tabla usuarios

Tabla pacientes

En la tabla **pacientes** se recoge toda la información de cada uno de los pacientes incluidos en la aplicación. También se incluye el id del usuario que utiliza cada paciente para acceder a su perfil.

La **clave primaria** en la tabla "pacientes" es "**dniPaciente**". Esta tabla se relaciona con las tablas "relaciones_pacientes" y "citas".

- En la tabla "**relaciones_pacientes**" hay dos claves foráneas y son "dniPaciente1" y "dniPaciente2", para poder identificar entre que dos pacientes se establece una relación.
- En la tabla "**citas**" la clave foránea es "dniPaciente", para poder identificar a que paciente corresponde la cita.

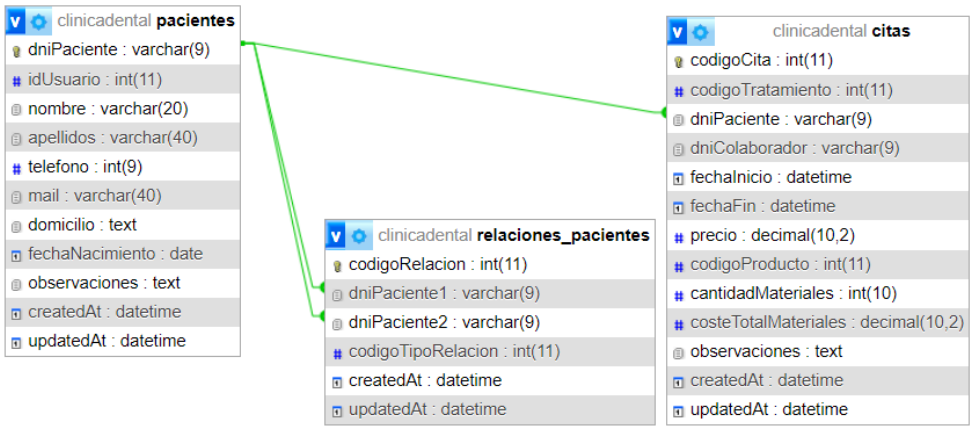


Figura Apéndice A —3.6—2. Relaciones de la tabla pacientes

Tabla colaboradores

En la tabla **colaboradores** se recoge toda la información de cada uno de los colaboradores incluidos en la aplicación. También se incluye el id del usuario que utiliza cada paciente para acceder a su perfil.

La **clave primaria** en la tabla "colaboradores" es "**dniColaborador**". Esta tabla se relaciona con la tabla "citas".

- En la tabla "**citas**" la clave foránea es "dniColaborador", para poder identificar a que colaborador corresponde la cita.

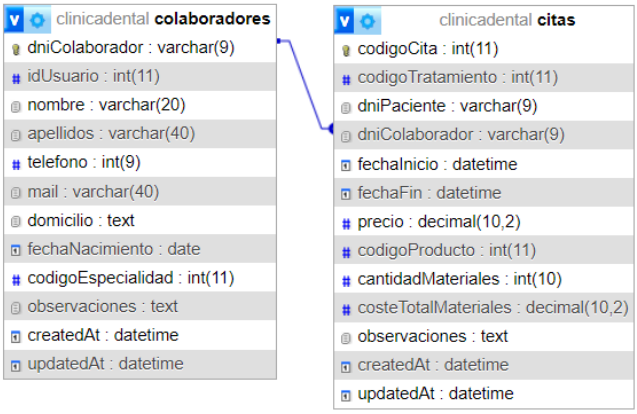


Figura Apéndice A —3.6—3. Relaciones de la tabla colaboradores

Tabla proveedores

En la tabla **proveedores** se recoge toda la información de cada uno de los proveedores incluidos en la aplicación.

La **clave primaria** en la tabla "proveedores" es "**cif**". Esta tabla se relaciona con la tabla "facturas_proveedores".

- En la tabla "**facturas_proveedores**" la clave foránea es "cif", para poder identificar a que proveedor corresponde la factura.

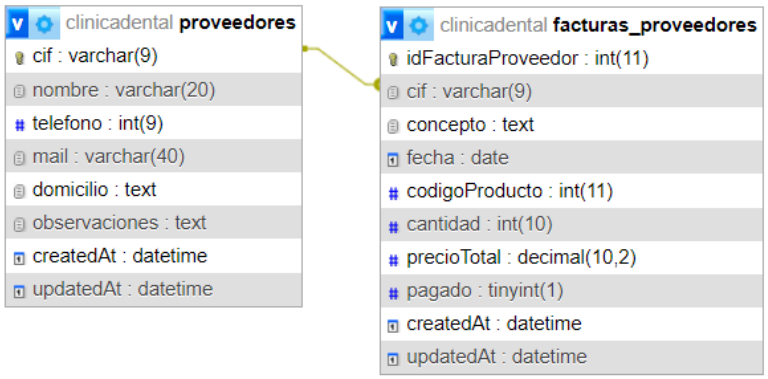


Figura Apéndice A —3.6—4. Relaciones de la tabla proveedores

Tabla tipos relación

En la tabla **tipos_relacion** se recogen las posibles relaciones que pueden existir entre pacientes.

La **clave primaria** en la tabla "tipos_relación" es "**codigoTipoRelacion**". Esta tabla se relaciona con la tabla "relaciones_pacientes".

- En la tabla "**relaciones_pacientes**" la clave foránea es "codigoTipoRelación", para poder identificar qué relación se establece entre dos pacientes.

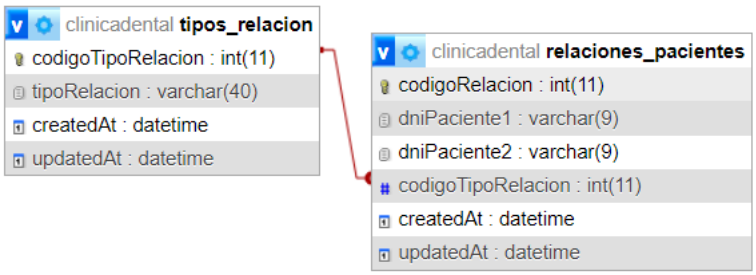


Figura Apéndice A —3.6—5. Relaciones de la tabla tipos relación

Tabla relaciones pacientes

En la tabla **relaciones_pacientes** se recogen todas las relaciones establecidas en la base de datos entre dos pacientes.

La **clave primaria** en la tabla "relaciones_pacientes" es "**codigoRelacion**". Esta tabla no se relaciona con ninguna otra tabla a través de su clave primaria.

Tabla especialidades

En la tabla **especialidades** se recogen las especialidades odontológicas que se desarrollan en la clínica dental.

La **clave primaria** en la tabla "especialidades" es "**codigoEspecialidad**". Esta tabla se relaciona con las tablas "colaboradores" y "tratamientos".

- En la tabla "**colaboradores**" la clave foránea es "codigoEspecialidad", para poder identificar en que área de la odontología está especializado un dentista.
- En la tabla "**tratamientos**" la clave foránea es "codigoEspecialidad", para poder identificar a que especialidad pertenece un tratamiento concreto.

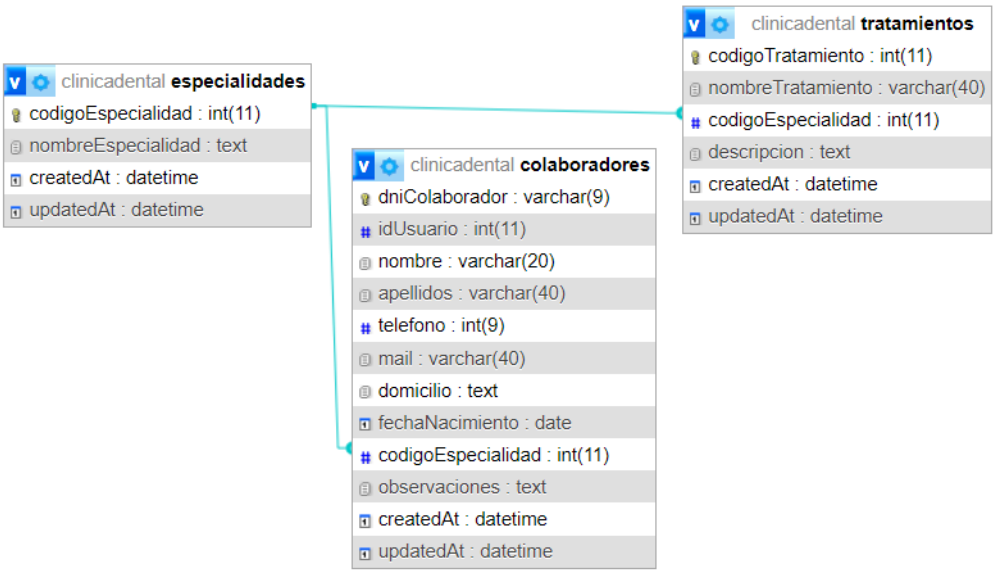


Figura Apéndice A —3.6—6. Relaciones de la tabla especialidades

Tabla tratamientos

En la tabla **tratamientos** se recogen los distintos tratamientos que se llevan a cabo en la clínica dental.

La **clave primaria** en la tabla "tratamientos" es "**codigoTratamiento**". Esta tabla se relaciona con la tabla "citas".

- En la tabla "**citas**" la clave foránea es "codigoTratamiento", para poder identificar el tratamiento que se realizará en la cita.

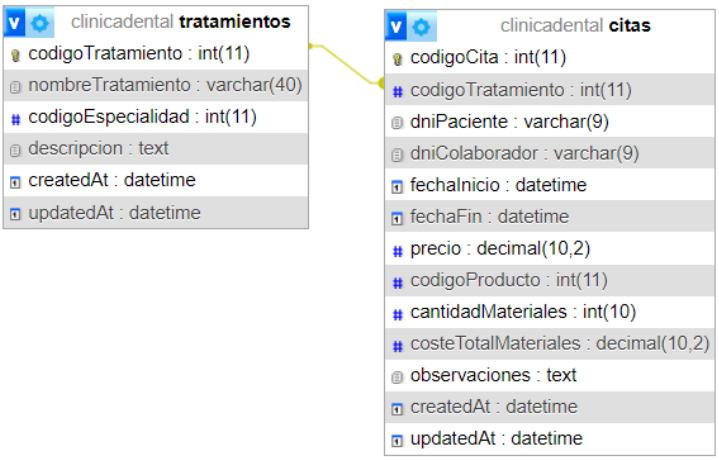


Figura Apéndice A —3.6—7. Relaciones de la tabla tratamientos

Tabla productos

En la tabla **productos** se recogen todos los productos que hay en el inventario, con su cantidad y valor correspondiente.

La **clave primaria** en la tabla "productos" es "**codigoProducto**". Esta tabla se relaciona con las tablas "citas" y "facturas_proveedores".

- En la tabla "**citas**" la clave foránea es "codigoProducto", para poder identificar que producto se ha utilizado en la realización del tratamiento, y poder disminuir el valor de las existencias en el inventario.
- En la tabla "**facturas_proveedores**" la clave foránea es "codigoProducto", para poder identificar a que producto se ha comprado, y poder aumentar el valor de las existencias en el inventario.

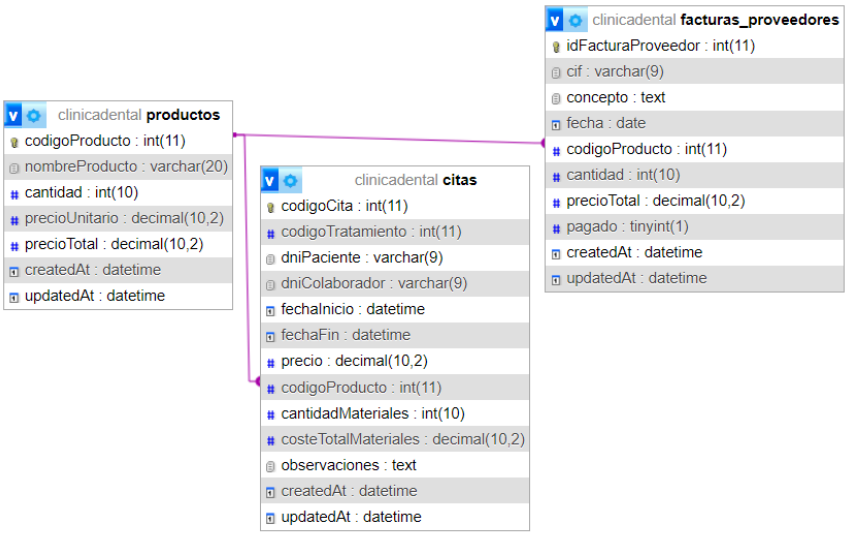


Figura Apéndice A —3.6—8. Relaciones de la tabla productos

Tabla citas

En la tabla **citas** se recogen todas las citas que se agendan con los pacientes que acuden a la clínica dental.

La **clave primaria** en la tabla "citas" es "**codigoCita**". Esta tabla se relaciona con las tablas "tickets_clientes" y "tickets_colaboradores".

- En la tabla "**tickets_clientes**" la clave foránea es "codigoCita", para poder identificar la cita a la que está asociado el ticket del paciente.
- En la tabla "**tickets_colaboradores**" la clave foránea es "codigoCita", para poder identificar la cita a la que está asociado el ticket del colaborador.

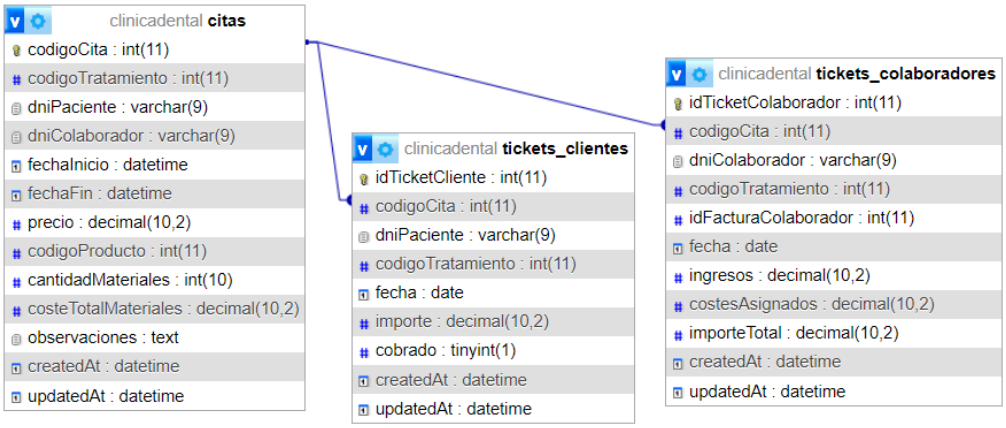


Figura Apéndice A —3.6—9. Relaciones de la tabla citas

Tabla tickets clientes

En la tabla **tickets_clientes** se recogen todos los tickets de los pacientes, que son generados de manera automática con cada cita, para llevar un control de los cobros que se llevan a cabo.

La **clave primaria** en la tabla "tickets_clientes" es "**idTicketCliente**". Esta tabla no se relaciona con ninguna otra tabla a través de su clave primaria.

Tabla tickets colaboradores

En la tabla **tickets_colaboradores** se recogen todos los tickets de los colaboradores, que son generados de manera automática una vez se marca como cobrado el ticket de un paciente.

La **clave primaria** en la tabla "tickets_colaboradores" es "**idTicketColaborador**". Esta tabla no se relaciona con ninguna otra tabla a través de su clave primaria.

Tabla facturas colaboradores

En la tabla **facturas_colaboradores** se recogen todos las facturas de cada mes y colaborador, estas facturas incluyen el valor total de los tickets correspondientes a cada colaborador a lo largo de todo el mes, para llevar un control de las cantidades que habrá que abonar a los colaboradores por la realización de los distintos tratamientos que llevan a cabo.

La **clave primaria** en la tabla "facturas_colaboradores" es "**idFacturaColaborador**". Esta tabla se relaciona con la tabla "tickets_colaboradores".

- En la tabla “**tickets_colaboradores**” la clave foránea es “idFacturaColaborador”, para poder identificar a que factura de colaborador esta asignado cada ticket.

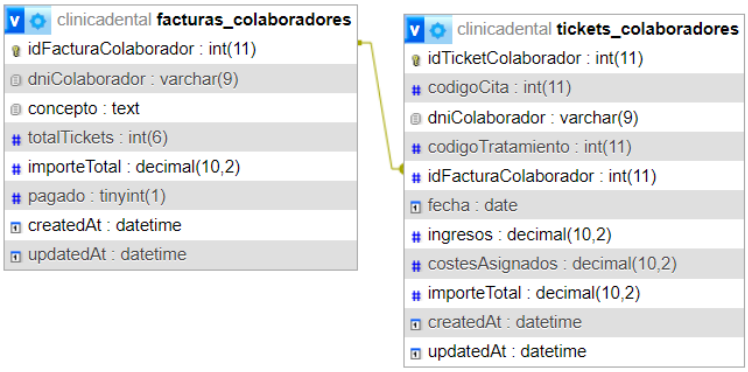


Figura Apéndice A —3.6—10. Relaciones de la tabla tickets colaboradores

Tabla facturas proveedores

En la tabla **facturas_proveedores** se recogen todos las facturas de compras de materiales de cada proveedor, para llevar un control de las existencias en el inventario y también las cantidades que habrá que abonar a los proveedores por la compra de productos.

La **clave primaria** en la tabla “facturas_proveedores” es “**idFacturaProveedor**”. Esta tabla no se relaciona con ninguna otra tabla a través de su clave primaria

Apéndice B - Detalle de los casos de uso

En esta sección se incluye el detalle de los casos de uso que no han sido explicados en el apartado “3.5 Casos de uso”. En este apartado 3.5, se encuentran explicados de manera detallada únicamente los casos de uso más complejos y relevantes, necesarios para comprender como funciona la aplicación web.

Caso de uso 1: Ver pacientes

Caso de uso 1	Ver pacientes
Objetivo	Permite visualizar a todos los pacientes, con información detallada, incluidos en la aplicación.
Actores	Administrador y Colaborador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin” o “colaborador”. La conexión con la base de datos está establecida.
Secuencia	1. El usuario accede a la opción “Pacientes” desde el menú principal. 2. El sistema realiza una consulta a la base de datos para obtener todos los pacientes y sus correspondientes datos (dni, nombre, apellidos, teléfono, email, domicilio, fecha de nacimiento y observaciones). 3. El sistema muestra en formato de tabla todos los pacientes con toda la información detallada de cada paciente.

Caso de uso 1	Ver pacientes
Postcondición	Se puede visualizar una tabla con los datos de todos los pacientes.

Tabla Apéndice B—1. Caso de uso 1: Ver pacientes

Caso de uso 2: Ver perfil paciente

Caso de uso 2	Ver perfil paciente
Objetivo	Permite visualizar toda la información incluida en el perfil de un paciente.
Actores	Administrador, Colaborador y Paciente.
Precondición	El usuario que ha iniciado sesión es de tipo "admin", "colaborador" o "paciente". La conexión con la base de datos está establecida.
Secuencia	1. El usuario accede al perfil de un paciente. a. En el caso de un usuario tipo "paciente" se accede al perfil del propio paciente al iniciar sesión de forma automática. b. En el caso de usuarios tipo "admin" o "colaborador" se accede al perfil de un paciente desde el listado de pacientes completo (caso de uso 1). 2. El sistema realiza varias consultas a la base de datos para obtener los datos de un paciente determinado (datos personales, relaciones con otros pacientes, calendario de citas y pagos realizados y pendientes). 3. El sistema muestra todos datos del paciente.
Postcondición	Se visualiza toda la información de un paciente

Tabla Apéndice B —2. Caso de uso 2: Ver perfil paciente

Caso de uso 3: Modificar paciente

Caso de uso 3	Modificar paciente
Objetivo	Permite editar cualquiera de los campos del perfil del paciente.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo “admin”. La conexión con la base de datos está establecida. El usuario se encuentra en el perfil de un paciente donde puede visualizar los datos personales.
Secuencia	1. El usuario pulsa el botón “Editar” que se encuentra al lado de cada campo del apartado “Datos personales”. 2. El sistema muestra el campo que se ha solicitado editar como un “input” (editable) y muestra a su lado dos botones: “Guardar” y “Cancelar”. 3. El usuario modifica el valor del campo. 4. El usuario pulsa “Guardar” 5. El sistema realiza una consulta a la base de datos para actualizar el valor del campo del paciente correspondiente.
Postcondición	Se muestran los datos personales actualizados en el perfil del paciente.

Tabla Apéndice B —3. Caso de uso 3: Modificar paciente

Caso de uso 4: Ver colaboradores

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todos los colaboradores, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Colaboradores" del menú principal y la información que se obtiene y se muestra es de los colaboradores (dni, nombre, apellidos, teléfono, email, domicilio, fecha de nacimiento, especialidad y observaciones).

Caso de uso 5: Ver perfil colaborador

Este caso de uso es similar al "Caso de uso 2: Ver perfil paciente", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar toda la información incluida en el perfil de un colaborador.
- Actores: Únicamente el Administrador y el Colaborador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin" o "colaborador".
- Secuencia: El procedimiento es el mismo pero, en el caso de un usuario de tipo "colaborador" se accede desde la opción "Mi perfil" del menú principal y en el caso de un usuario "admin" se accede desde el listado de colaboradores completo (caso de uso 4). Se muestran los datos de un colaborador determinado (datos personales, calendario de citas, y facturas).

Caso de uso 6: Modificar colaborador

Este caso de uso es similar al "Caso de uso 3: Modificar paciente", a continuación se indican únicamente las diferencias:

- Objetivo: Permite editar cualquiera de los campos del perfil del colaborador.
- Precondición: El usuario se encuentra en el perfil de un colaborador.
- Secuencia: El procedimiento es el mismo pero la información que se actualiza es de un colaborador.

Caso de uso 7: Ver proveedores

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todos los proveedores, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Proveedores" del menú principal y la información que se obtiene y se muestra es de los proveedores (cif, nombre, teléfono, email, domicilio y observaciones).

Caso de uso 11: Ver tratamientos

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar todos los tratamientos, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Tratamientos" del menú principal y la información que se obtiene y se muestra es de los tratamientos (nombre, especialidad a la que pertenece y descripción).

Caso de uso 12: Añadir tratamiento

Este caso de uso es similar al "Caso de uso 8: Añadir paciente", a continuación se indican únicamente las diferencias:

- Objetivo: Permite añadir un nuevo tratamiento a la aplicación.
- Secuencia: El procedimiento es el mismo, pero solo los pasos 1-3, 5, y se realiza desde la opción "Añadir tratamiento" del menú principal. Los campos solicitados para añadir un nuevo tratamiento son: nombre, especialidad a la que pertenece y descripción.

Caso de uso 13: Ver citas

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar todas las citas, con información detallada, incluidas en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Citas" del menú principal y la información que se obtiene y se muestra son las citas en formato de calendario.

Caso de uso 15: Ver productos

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todos los productos, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".

- Secuencia: El procedimiento es el mismo pero desde la opción “Productos” del menú principal y la información que se obtiene y se muestra es de los productos (nombre, cantidad, precio unitario y precio total).

Caso de uso 16: Añadir producto

Este caso de uso es similar al “Caso de uso 8: Añadir paciente”, a continuación se indican únicamente las diferencias:

- Objetivo: Permite añadir un nuevo producto a la aplicación (con existencias iniciales a 0) para que esté disponible para seleccionarlo al registrar una factura de materiales de un proveedor.
- Secuencia: El procedimiento es el mismo, pero solo los pasos 1-3, 5, y se realiza desde la opción “Añadir producto” del menú principal. El campo solicitado para añadir un nuevo producto es el nombre.

Caso de uso 17: Ver tickets pacientes

Este caso de uso es similar al “Caso de uso 1: ”Ver pacientes”, a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todos los tickets de los pacientes, con información detallada, incluidos en la aplicación. Estos tickets han sido generados al crear una cita, con el objetivo de llevar un control de cobros de los tratamientos realizados.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo “admin”.
- Secuencia: El procedimiento es el mismo pero desde la opción “Tickets Pacientes” del menú principal y la información que se obtiene y se muestra es de los tickets de los pacientes (id del ticket, nombre del paciente, tratamiento realizado, fecha, importe y un checkbox que indica si ha sido cobrado).

Caso de uso 18: Ver tickets pacientes pendientes de cobro

Este caso de uso es similar al "Caso de uso 17: Ver tickets pacientes", pero el sistema realiza la consulta a la base de datos filtrando por el campo "cobrado" para mostrar únicamente los tickets pendientes de cobro.

Caso de uso 20: Ver facturas colaboradores

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todas las facturas de los colaboradores, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Facturas colaboradores" del menú principal y la información que se obtiene y se muestra es de las facturas de los colaboradores (id de la factura, nombre del colaborador, mes, total tickets incluidos, importe total y un checkbox que indica si ha sido pagada).

Caso de uso 21: Ver tickets colaboradores

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todos los tickets incluidos en una factura de un colaborador y mes determinado, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin". Y se encuentra visualizando el listado completo de las facturas de los colaboradores o el listado de las facturas de colaboradores pendientes de pago (caso de uso 20 o 22).

- Secuencia: El procedimiento es el mismo pero se elige una factura del listado completo de facturas para visualizar los tickets correspondientes. Y la información que se obtiene y se muestra es de las facturas de los colaboradores (id del ticket, nombre del colaborador, tratamiento realizado, fecha, ingresos, costes asignados e importe total).

Caso de uso 22: Ver facturas colaboradores pendientes de pago

Este caso de uso es similar al "Caso de uso 20: Ver facturas colaboradores", pero el sistema realiza la consulta a la base de datos filtrando por el campo "pagado" para mostrar únicamente las facturas pendientes de pago.

Caso de uso 23: Marcar factura colaborador como pagada

Caso de uso 23	Marcar factura colaborador como pagada
Objetivo	Permite marcar una factura de un colaborador, como pagada cuando se realiza el pago del importe total de la factura al dentista correspondiente, para llevar un control de los pagos pendientes.
Actores	Administrador.
Precondición	El usuario que ha iniciado sesión es de tipo "admin". La conexión con la base de datos está establecida. El usuario se encuentra visualizando el listado de facturas de colaboradores (listado completo o listado de facturas pendientes de pago).
Secuencia	1. El usuario marca un checkbox vacío correspondiente a una factura en concreto.

Caso de uso 23	Marcar factura colaborador como pagada
	2. El sistema realiza una consulta a la base de datos para actualizar el campo "pagado" de la factura de colaborador correspondiente.
Postcondición	Se ha actualizado el campo "pagado" de la factura correspondiente.

Tabla Apéndice B —4. Caso de uso 23: Marcar factura colaborador como pagada

Caso de uso 25: Ver facturas proveedores

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite visualizar a todas las facturas de los proveedores, con información detallada, incluidos en la aplicación.
- Actores: Únicamente el Administrador
- Precondición: El usuario que ha iniciado sesión es de tipo "admin".
- Secuencia: El procedimiento es el mismo pero desde la opción "Facturas proveedores" del menú principal y la información que se obtiene y se muestra es de las facturas de los proveedores (id de la factura, cif del proveedor, nombre del proveedor, concepto, fecha, producto adquirido, cantidad adquirida, precio total y un checkbox que indica si ha sido pagada).

Caso de uso 26: Ver facturas proveedores pendientes de pago

Este caso de uso es similar al "Caso de uso 25: Ver facturas proveedores", pero el sistema realiza la consulta a la base de datos filtrando por el campo "pagado" para mostrar únicamente las facturas pendientes de pago.

Caso de uso 27: Marcar factura proveedor como pagada

Este caso de uso es similar al "Caso de uso 1: Ver pacientes", a continuación se indican únicamente las diferencias:

- Objetivo: Permite marcar una factura de un proveedor como pagada, cuando se realiza el pago del importe total de la factura a la empresa correspondiente, para llevar un control de los pagos pendientes.
- Precondición: El usuario se encuentra visualizando el listado de facturas de proveedores (listado completo o listado de facturas pendientes de pago).
- Secuencia: El procedimiento es el mismo pero la información que se actualiza es el campo "pagado" de la facturas de un proveedor concreto.

Apéndice C - Credenciales para acceder a la aplicación

A continuación se muestran las credenciales necesarias para acceder a la aplicación con cada uno de los tres tipos de usuarios disponibles:

- Perfil Administrador. Usuario: 1. Contraseña admin.
- Perfil Paciente. Usuario: 2. Contraseña 1234.
- Perfil Colaborador. Usuario: 3. Contraseña 1234.