

---

## **Especificación de requisitos de software**

**Proyecto: Desarrollo de una aplicación web para la gestión de citas del consultorio odontológico Bella Dent utilizando el framework Django con React**  
Revisión 1

# **Instrucciones para el uso de este formato**

*Este formato es una plantilla tipo para documentos de requisitos del software.*

*Está basado y es conforme con el estándar IEEE Std 830-1998.*

*Las secciones que no se consideren aplicables al sistema descrito podrán de forma justificada indicarse como no aplicables (NA).*

*Notas:*

*Los textos en color azul son indicaciones que deben eliminarse y, en su caso, sustituirse por los contenidos descritos en cada apartado.*

*Los textos entre corchetes del tipo “[Inserte aquí el texto]” permiten la inclusión directa de texto con el color y estilo adecuado a la sección, al pulsar sobre ellos con el puntero del ratón.*

*Los títulos y subtítulos de cada apartado están definidos como estilos de MS Word, de forma que su numeración consecutiva se genera automáticamente según se trate de estilos “Titulo1, Titulo2 y Titulo3”.*

*La sangría de los textos dentro de cada apartado se genera automáticamente al pulsar Intro al final de la línea de título. (Estilos Normal indentado1, Normal indentado 2 y Normal indentado 3).*

*El índice del documento es una tabla de contenido que MS Word actualiza tomando como criterio los títulos del documento.*

*Una vez terminada su redacción debe indicarse a Word que actualice todo su contenido para reflejar el contenido definitivo.*

# Ficha del documento

Fecha	Revisión	Autor	Verificado dep. calidad.
07/10/2025	1	Steve Tibán	

Documento validado por las partes en fecha:

Por el cliente	Por la empresa suministradora
Fdo. D./ Dña [Nombre]	Fdo. D./Dña [Nombre]

## Contenido

<b>FICHA DEL DOCUMENTO</b>	<b>3</b>
<b>CONTENIDO</b>	<b>4</b>
<b>1 INTRODUCCIÓN</b>	<b>6</b>
1.1 Propósito	6
1.2 Alcance	6
1.3 Personal involucrado	6
1.4 Definiciones, acrónimos y abreviaturas	7
1.5 Referencias	7
1.6 Resumen	7
<b>2 DESCRIPCIÓN GENERAL</b>	<b>8</b>
2.1 Perspectiva del producto	8
2.2 Funcionalidad del producto	8
2.3 Características de los usuarios	10
2.4 Restricciones	11
2.5 Suposiciones y dependencias	11
2.6 Evolución previsible del sistema	12
<b>3 REQUISITOS ESPECÍFICOS</b>	<b>12</b>
<b>3.1 Requisitos comunes de las interfaces</b>	<b>12</b>
3.1.1 Interfaces de usuario	13
3.1.2 Interfaces de hardware	14
3.1.3 Interfaces de software	15
3.1.4 Interfaces de comunicación	16
<b>3.2 Requisitos funcionales</b>	<b>17</b>
3.2.1 Requisito funcional 1	17
3.2.2 Requisito funcional 2	18
3.2.3 Requisito funcional 3	19
3.2.4 Requisito funcional 4	20
3.2.5 Requisito funcional 5	21
3.2.6 Requisito funcional 6	22
3.2.7 Requisito funcional 7	23
3.2.8 Requisito funcional 8	24
3.2.9 Requisito funcional 9	25
3.2.10 Requisito funcional 10	26
3.2.11 Requisito funcional 11	27

---

3.2.12	Requisito funcional 12	28
3.2.13	Requisito funcional 13	29
3.2.14	Requisito funcional 14	30
3.2.15	Requisito funcional 15	30
3.2.16	Requisito funcional 16	31
<b>3.3</b>	<b>Requisitos no funcionales</b>	<b>32</b>
3.3.1	Requisitos de rendimiento	32
3.3.2	Seguridad	33
3.3.3	Fiabilidad	33
3.3.4	Disponibilidad	34
3.3.5	Mantenibilidad	34
3.3.6	Portabilidad	34
<b>3.4</b>	<b>Otros requisitos</b>	<b>35</b>

# 1 Introducción

La presente Especificación de Requisitos de Software define de manera detallada las funcionalidades, restricciones, comportamientos y características de la aplicación web desarrollada como solución tecnológica para el consultorio odontológico Bella Dent.

Este documento tiene como objetivo proporcionar una descripción clara y verificable de los requisitos que el sistema debe cumplir, sirviendo como base para las etapas de diseño, desarrollo, validación y mantenimiento del software.

## 1.1 Propósito

El propósito de este documento es establecer, describir y detallar los requisitos de la aplicación web, que automatiza los procesos de gestión de citas, atención odontológica, control de pacientes, registro clínico y generación de reportes para el consultorio odontológico Bella Dent.

Este documento está destinado a los siguientes grupos:

- Desarrolladores y diseñadores de software, quienes usarán la SRS como guía técnica para implementar las funcionalidades.
- Usuarios finales (administradores, odontólogos y pacientes), como referencia para comprender el alcance de las funcionalidades que serán implementadas.
- Docentes evaluadores y revisores académicos, que validarán el cumplimiento metodológico y técnico del proyecto de titulación conforme a estándares de ingeniería de software.

## 1.2 Alcance

El producto desarrollado se denomina OralFlow: Sistema Web para la Gestión de Citas Odontológicas del Consultorio Bella Dent, una aplicación web multiplataforma desarrollada con Django (backend), React (frontend) y PostgreSQL (base de datos).

El sistema permitirá automatizar procesos relacionados con la administración de citas, control de pacientes, registro de fichas clínicas, agenda de odontólogos y comunicación con los usuarios mediante recordatorios automáticos vía WhatsApp.

## 1.3 Personal involucrado

Nombre	Aracely Cedeño
Rol	Product Owner
Categoría profesional	Odontóloga – Responsable del Consultorio Bella Dent
Responsabilidades	<ul style="list-style-type: none"><li>• Definir los requerimientos funcionales y validar las funcionalidades del sistema desde la perspectiva del usuario final.</li><li>• Proporcionar información sobre los procesos actuales de gestión de citas y atención al paciente.</li><li>• Aprobar los incrementos del producto desarrollados en cada sprint.</li></ul>
Información de contacto	belladentrio@gmail.com

Nombre	Julio Guallo
Rol	Scrum Master / Director de Tesis
Categoría profesional	Docente ESPOCH
Responsabilidades	<ul style="list-style-type: none"><li>• Facilitar el cumplimiento de la metodología SCRUM.</li><li>• Supervisar el avance del proyecto y garantizar la correcta aplicación de las prácticas de ingeniería de software.</li><li>• Validar los entregables académicos y técnicos del proyecto.</li></ul>
Información de contacto	julio.guallo@epoch.edu.ec

Nombre	Steve Tibán
Rol	Desarrollador / Tesista
Categoría profesional	Estudiante de Ingeniería en Software – ESPOCH
Responsabilidades	<ul style="list-style-type: none"> <li>Analizar, diseñar, desarrollar e implementar la aplicación web.</li> <li>Documentar los requisitos, modelos y casos de uso del sistema.</li> <li>Integrar los módulos backend (Django REST Framework) y frontend (React).</li> <li>Ejecutar pruebas, correcciones y despliegue final.</li> </ul>
Información de contacto	steve.tiban@epoch.edu.ec

## 1.4 Definiciones, acrónimos y abreviaturas

Término / Acrónimo	Definición
<b>SRS</b>	Software Requirements Specification (Especificación de Requisitos de Software): Documento formal que describe de manera completa y verificable las funcionalidades y restricciones del sistema a desarrollar.
<b>RF</b>	Requisito Funcional
<b>RNF</b>	Requisito No Funcional
<b>RUI</b>	Requisito de Interfaz de Usuario
<b>RHW</b>	Requisito de Interfaz de Hardware
<b>RSW</b>	Requisito de Interfaz de Software
<b>RCM</b>	Requisito de Interfaz de Comunicación
<b>CRUD</b>	Acrónimo de Create, Read, Update, Delete.
<b>BD</b>	Base de Datos.

## 1.5 Referencias

Referencia	Título	Ruta	Fecha	Autor
REF-01	Alcance del proyecto	Documento interno del proyecto OralFlow	23/09/2025	Steve Tibán
REF-02	Pila de tecnologías	Documento interno del proyecto OralFlow	30/09/2025	Steve Tibán
REF-03	Especificación de casos de uso	Documento interno del proyecto OralFlow	07/10/2025	Steve Tibán

## 1.6 Resumen

El presente documento reúne la especificación de requisitos del sistema OralFlow, una aplicación web para la gestión de citas y atención odontológica en el consultorio Bella Dent. En él se detallan los requisitos funcionales, no funcionales y legales que garantizan el correcto funcionamiento, seguridad, fiabilidad y cumplimiento normativo del sistema.

El documento describe inicialmente la perspectiva general del producto, los módulos que lo componen, los tipos de usuarios y las restricciones del sistema. OralFlow se desarrolla bajo una arquitectura cliente-servidor con backend en Django (Python) y frontend en React (JavaScript), permitiendo su ejecución en sistemas operativos Windows y Linux, y acceso desde los navegadores Chrome, Firefox, Edge y Safari.

Se definen los requisitos funcionales que describen las principales acciones del sistema, como la autenticación por roles, gestión de usuarios y citas, envío de notificaciones, registro clínico y generación de reportes. Los requisitos no funcionales aseguran su calidad mediante un tiempo de respuesta óptimo, uso de autenticación segura con JWT, arquitectura modular en Django, portabilidad entre entornos Windows y Linux y

## 2 Descripción general

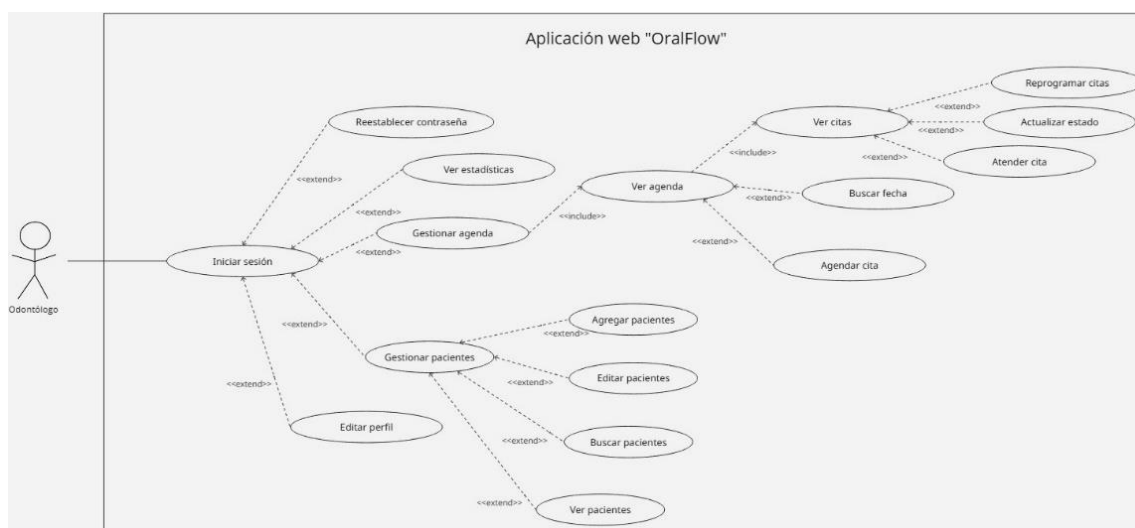
## 2.1 Perspectiva del producto

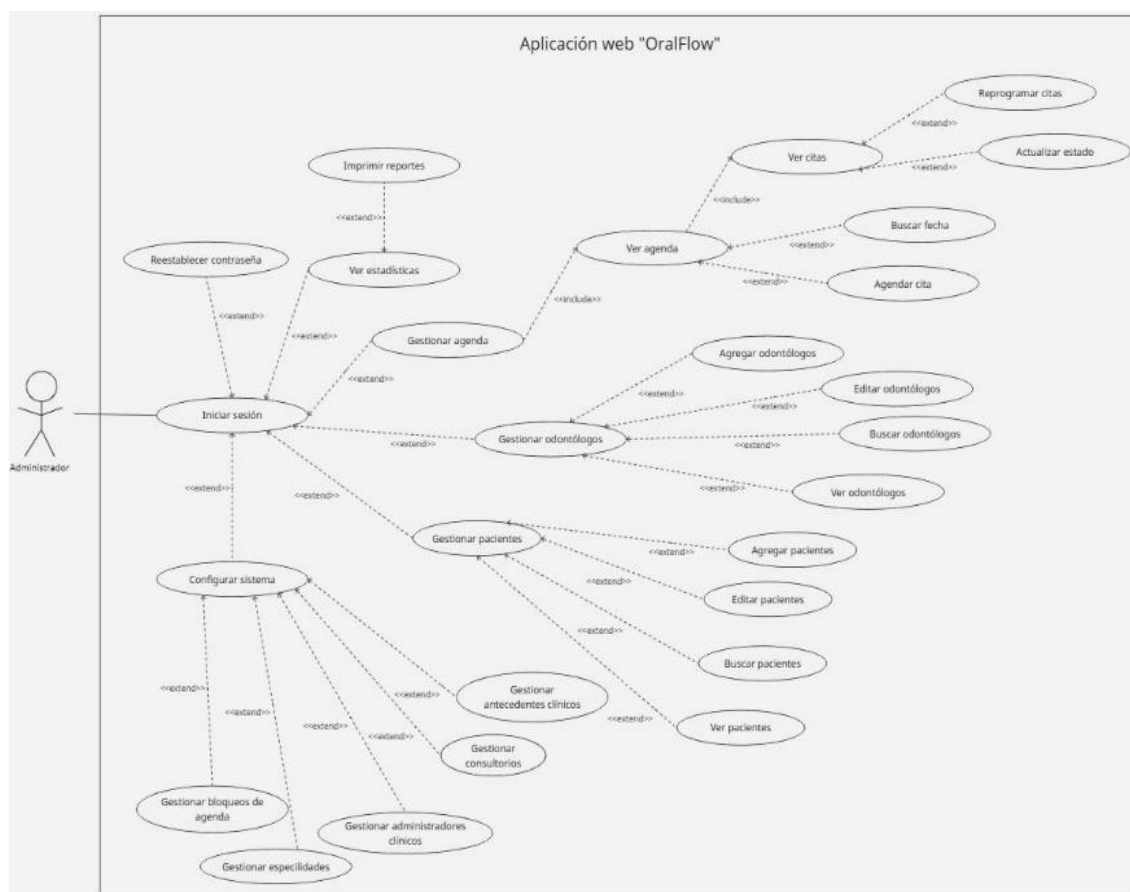
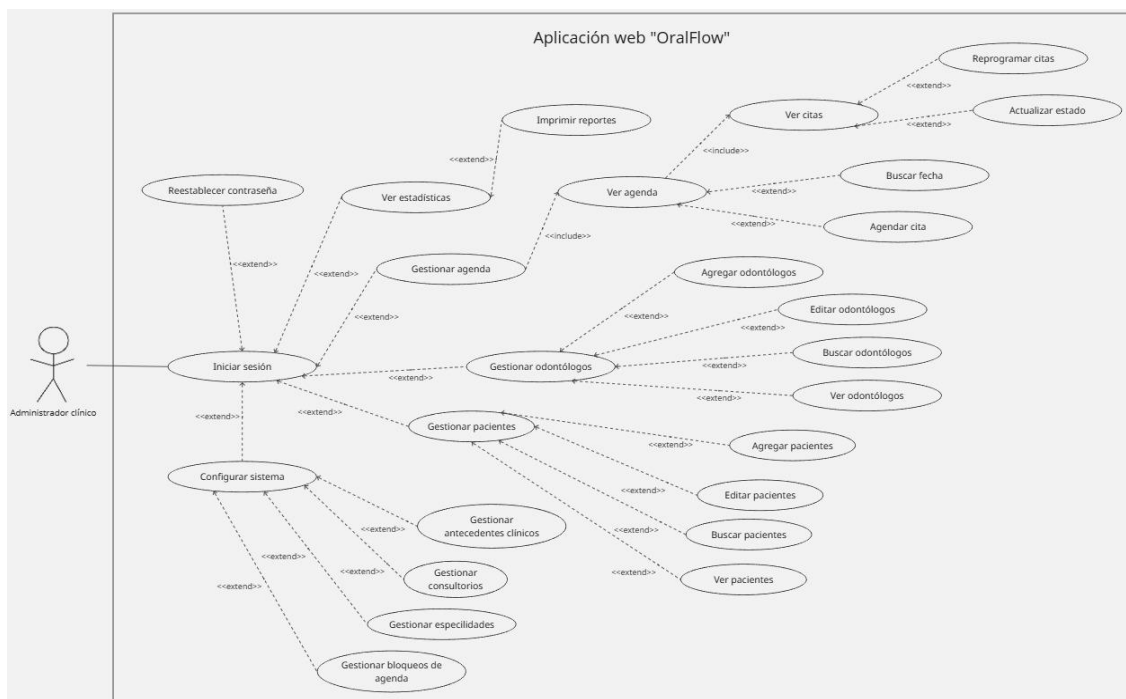
No forma parte de un sistema mayor, sino que constituye una solución integral que concentra la funcionalidad administrativa, clínica y operativa en una única aplicación web.

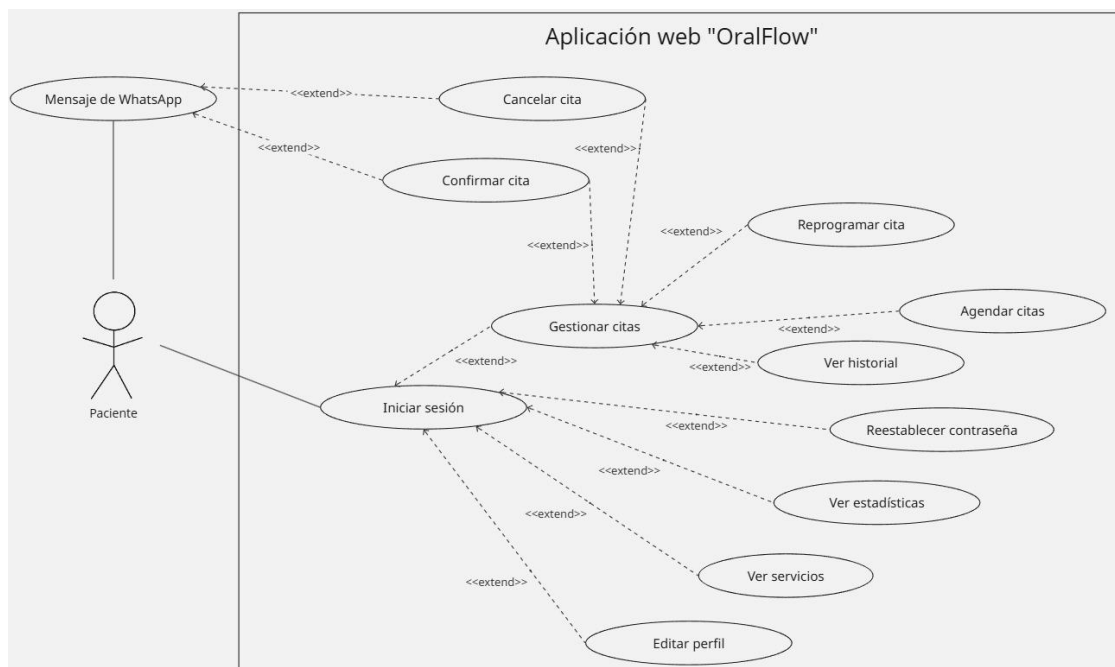
## 2.2 Funcionalidad del producto

A nivel general, las principales funcionalidades del producto son las siguientes:

- **Módulo de Autenticación y acceso por roles:** Controla el ingreso al sistema según el tipo de usuario (administrador, administrador clínico, odontólogo o paciente), garantizando la seguridad y personalización de la experiencia de uso.
- **Módulo de Gestión de Citas y Agenda:** Centraliza la programación, consulta, reprogramación y cancelación de citas, mostrando calendarios dinámicos que facilitan la organización de la atención odontológica.
- **Módulo de Notificaciones:** Automatiza la comunicación con los pacientes mediante el envío de recordatorios y confirmaciones de citas por WhatsApp, reduciendo ausencias y mejorando la coordinación.
- **Módulo de Gestión de Pacientes:** Permite registrar, actualizar y consultar la información personal y médica de los pacientes, asegurando la integridad y confidencialidad de los datos.
- **Módulo de Gestión de Odontólogos:** Administra los datos del personal odontológico, sus especialidades, horarios y disponibilidad, manteniendo la información actualizada para la correcta asignación de citas.
- **Módulo de Reportes y Estadísticas:** Genera reportes e indicadores que permiten analizar el desempeño del consultorio, la asistencia de pacientes y la productividad de los odontólogos.







## 2.3 Características de los usuarios

Tipo de usuario	Administrador
Formación	Formación técnica o administrativa en el área de salud y de sistemas. Experiencia en coordinación de agendas y atención a pacientes.
Habilidades	Conocimientos generales de uso de herramientas web. Habilidad para planificar y coordinar horarios, verificar disponibilidad y controlar agendas.
Actividades	Supervisa el funcionamiento del sistema, gestiona roles de usuario (administradores clínicos, odontólogos y pacientes), configura parámetros generales (consultorios, especialidades, bloqueos de agenda) y genera reportes globales del consultorio.

Tipo de usuario	Administrador Clínico
Formación	Formación técnica o administrativa en el área de salud. Experiencia en coordinación de agendas y atención a pacientes.
Habilidades	Manejo básico de sistemas informáticos. Habilidad para planificar y coordinar horarios, verificar disponibilidad y controlar agendas.
Actividades	Gestiona la agenda general del consultorio, crea y modifica citas, administra la información de pacientes y odontólogos, genera reportes operativos y apoya al administrador en el control de procesos.

Tipo de usuario	Odontólogo
Formación	Título profesional en odontología y experiencia en atención clínica.
Habilidades	Conocimientos básicos en el uso de herramientas digitales. Capacidad para registrar información clínica y manejar agendas en línea.

Actividades	Visualiza y gestiona su agenda personal, atiende citas confirmadas, registra fichas médicas (diagnósticos, tratamientos y observaciones), consulta el historial clínico de sus pacientes y analiza estadísticas de su desempeño.
-------------	--

Tipo de usuario	Paciente
Formación	Público general, sin necesidad de formación técnica.
Habilidades	Habilidad básica en el uso de computadoras o dispositivos móviles. Familiaridad con entornos web y aplicaciones de mensajería.
Actividades	Se registra en el sistema, inicia sesión, agenda y confirma citas, cancela o reprograma turnos, consulta su historial médico y recibe notificaciones automáticas de recordatorio por WhatsApp.

## 2.4 Restricciones

Durante el diseño y desarrollo del sistema OralFlow se identifican las siguientes restricciones técnicas, funcionales y metodológicas que delimitan su alcance actual:

### Alcance funcional limitado:

- El sistema no incluye integración con pasarelas de pago ni gestión de transacciones financieras.
- No se implementan funciones de inteligencia artificial (IA) para diagnóstico o predicción clínica en esta versión inicial.
- Las notificaciones se limitan al envío de mensajes de texto a través de la API de WhatsApp (Twilio), sin integración con otros canales de mensajería.

### Restricciones tecnológicas:

- El desarrollo se realiza exclusivamente con Django REST Framework (backend), React (frontend) y PostgreSQL (base de datos).
- Se debe mantener la compatibilidad con navegadores modernos (Google Chrome, Mozilla Firefox, Microsoft Edge).
- El sistema requiere conexión estable a internet para su funcionamiento completo, ya que depende de servicios en línea (API REST y notificaciones).

### Restricciones metodológicas:

- El proyecto se desarrolla bajo la metodología ágil SCRUM, con entregas parciales en sprints definidos, lo cual condiciona el ritmo y orden de implementación de los módulos.
- Se deben cumplir los estándares de codificación establecidos (camelCase para variables, PascalCase para clases, snake\_case para campos de base de datos).

### Restricciones de seguridad y datos:

- El acceso está limitado a usuarios registrados con autenticación por roles y tokens de seguridad.
- No se almacena información sensible como contraseñas en texto plano.
- Los datos clínicos se gestionan bajo criterios de confidencialidad y trazabilidad, conforme a las buenas prácticas de seguridad en sistemas de salud.

## 2.5 Suposiciones y dependencias

El desarrollo y correcto funcionamiento del sistema OralFlow dependen de una serie de factores técnicos, operativos y externos que se asumen disponibles o estables durante todo el ciclo de vida del software.

En caso de que alguna de estas condiciones cambie, será necesario revisar y posiblemente ajustar los requisitos establecidos en esta especificación.

**Disponibilidad de infraestructura tecnológica:** El sistema será implementado en un servidor Vercel (Frontend) y servidor Render (Backend) con capacidad suficiente para almacenar y procesar la información del consultorio.

**Dependencia de servicios externos:** El módulo de notificaciones depende del servicio Twilio API para WhatsApp, cuya disponibilidad o cambios en las políticas pueden afectar el envío de recordatorios.

**Compatibilidad del entorno de usuario:** Se asume que los usuarios accederán al sistema desde navegadores web modernos (Google Chrome, Mozilla Firefox o Microsoft Edge) con soporte para HTML5 y JavaScript.

**Suposiciones operativas y organizacionales:** Se asume que los usuarios (odontólogos, administradores y pacientes) contarán con conocimientos básicos en el uso de aplicaciones web.

## 2.6 Evolución previsible del sistema

El sistema OralFlow ha sido diseñado bajo una arquitectura modular y escalable, lo que permitirá incorporar mejoras y nuevas funcionalidades en versiones futuras sin afectar la estabilidad del núcleo del sistema.

Entre las posibles evoluciones planificadas o contempladas para fases posteriores del proyecto se incluyen:

### Implementación de un módulo de pagos electrónicos:

- Integración con pasarelas de pago en línea (como PayPhone, Stripe o PayPal) para permitir a los pacientes cancelar los servicios odontológicos de manera digital y segura.
- Control administrativo de ingresos, pagos pendientes y reportes financieros vinculados al sistema de gestión de citas.

### Ampliación del módulo de atención clínica:

- Incorporación de herramientas gráficas interactivas para el registro odontográfico (odontograma) y registro endodóntico (endorgrama), permitiendo documentar el estado dental y los procedimientos de cada pieza dentaria.
- Integración de funciones avanzadas para el seguimiento evolutivo del paciente, visualización histórica y comparación entre atenciones.

## 3 Requisitos específicos

### 3.1 Requisitos comunes de las interfaces

El sistema OralFlow dispone de una arquitectura cliente–servidor, en la cual los distintos componentes intercambian información mediante interfaces bien definidas.

Las interfaces del sistema permiten la comunicación fluida entre los usuarios, el backend, la base de datos y los servicios externos, garantizando la integridad y disponibilidad de los datos.

#### Entradas del sistema:

- Credenciales de acceso (correo electrónico y contraseña) para el inicio de sesión.
- Datos personales y clínicos de pacientes ingresados por el personal odontológico o administrativo.
- Información de registro y configuración de usuarios (odontólogos, especialidades, consultorios y horarios).
- Datos de citas: fecha, hora, motivo, odontólogo asignado, estado (pendiente, confirmada, cancelada, reprogramada).
- Información de fichas médicas: observaciones, diagnóstico, tratamiento y archivos adjuntos.
- Acciones del usuario en la interfaz: clics, formularios, filtros, confirmaciones y reprogramaciones.

#### Salidas del sistema:

- Visualización de datos en tiempo real mediante tablas, calendarios y gráficos estadísticos.
- Confirmaciones visuales (mensajes, notificaciones y alertas) sobre las operaciones realizadas.
- Generación de reportes en formato PDF con información de citas, pacientes y estadísticas de desempeño.

- Notificaciones automáticas a través de la API de WhatsApp (Twilio) para recordar citas y permitir confirmaciones.
- Actualizaciones dinámicas en la interfaz de usuario cuando se modifican o agregan registros.

### 3.1.1 Interfaces de usuario

El sistema OralFlow contará con una interfaz web moderna, intuitiva y responsiva, desarrollada con React.

El diseño prioriza la facilidad de uso, la accesibilidad, la coherencia visual y la personalización de acuerdo con el rol del usuario (Administrador, Administrador Clínico, Odontólogo y Paciente).

La interfaz sigue un enfoque minimalista y funcional, empleando colores institucionales azul (#0070B7) y blanco, tipografía sans-serif (Roboto), y componentes reutilizables que garantizan una experiencia homogénea en todo el sistema.

Número de requisito	RUI-01
Nombre de requisito	Diseño responsivo y adaptable
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Lineamientos de diseño del proyecto OralFlow
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	La interfaz debe ajustarse automáticamente al tamaño y resolución de la pantalla (PC, laptop, tablet o teléfono). Todos los componentes deben conservar su funcionalidad y legibilidad independientemente del dispositivo utilizado.

Número de requisito	RUI-02
Nombre de requisito	Componentes reutilizables e íconos estandarizados
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Buenas prácticas de desarrollo frontend
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial <input checked="" type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	Los botones, tarjetas, formularios y tablas deben implementarse como componentes reutilizables. Los íconos provendrán de la librería Lucide React, garantizando consistencia visual y semántica.

Número de requisito	RUI-03
Nombre de requisito	Notificaciones visuales y modales de confirmación
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Requisitos de interacción del usuario
Prioridad del requisito	<input type="checkbox"/> Alta/Esencial <input checked="" type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	El sistema debe mostrar mensajes visuales de confirmación, advertencia o error mediante toasts o modales, antes de ejecutar acciones críticas (como eliminar registros o cancelar citas).

Número de requisito	RUI-04
Nombre de requisito	Validación visual de formularios
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Requisitos de interacción del usuario
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	Los formularios deben mostrar indicadores visuales para campos requeridos, mensajes de error y confirmación al usuario.

	Los errores deben resaltarse en color rojo (#E74C3C) y las acciones exitosas en verde (#2ECC71).
Número de requisito	RUI-05
Nombre de requisito	Idioma e interfaz en español
Tipo	<input type="checkbox"/> Requisito <input checked="" type="checkbox"/> Restricción
Fuente del requisito	Requisitos de interacción del usuario
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	Toda la interfaz, mensajes y etiquetas deben estar redactados en español neutro, evitando tecnicismos o abreviaciones confusas para los usuarios finales.

### 3.1.2 Interfaces de hardware

El sistema OralFlow se ejecutará sobre infraestructura de tipo cliente–servidor y no requiere hardware especializado.

Sin embargo, se definen los requisitos mínimos de hardware tanto para el servidor como para los equipos cliente, con el fin de asegurar el correcto funcionamiento, desempeño y disponibilidad del sistema.

Número de requisito	RHW-01
Nombre de requisito	Requisitos de servidor
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Requisitos técnicos del proyecto
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	<p>El sistema debe alojarse en un servidor físico o virtual con las siguientes especificaciones mínimas:</p> <ul style="list-style-type: none"> <li>• <b>Procesador:</b> 4 núcleos (mínimo 2.5 GHz)</li> <li>• <b>Memoria RAM:</b> 8 GB o superior</li> <li>• <b>Almacenamiento:</b> 100 GB de espacio disponible</li> <li>• <b>Sistema operativo:</b> Ubuntu Server 22.04 LTS o equivalente compatible con Python 3.10+</li> </ul> <p>La configuración del servidor debe garantizar una alta disponibilidad (24/7) y permitir escalabilidad en caso de aumento de usuarios concurrentes.</p>

Número de requisito	RHW-02
Nombre de requisito	Requisitos del equipo del cliente
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Requisitos técnicos del proyecto
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Descripción	<p>Los usuarios accederán al sistema desde equipos personales con las siguientes condiciones mínimas:</p> <ul style="list-style-type: none"> <li>• <b>Procesador:</b> Doble núcleo o superior</li> <li>• <b>Memoria RAM:</b> 4 GB o superior</li> <li>• <b>Conectividad:</b> Acceso estable a internet (mínimo 2 Mbps)</li> <li>• <b>Resolución de pantalla mínima:</b> 1280 × 720 píxeles</li> <li>• <b>Navegador compatible:</b> Google Chrome, Mozilla Firefox o Microsoft Edge (últimas versiones)</li> </ul>

### 3.1.3 Interfaces de software

El sistema OralFlow se compone de varios elementos de software interconectados que permiten la ejecución coordinada de sus funcionalidades principales. Estas interfaces garantizan la interoperabilidad entre el frontend, el backend, la base de datos y los servicios externos empleados por el sistema.

Número de requisito	RSW-01
Nombre de requisito	Interfaz entre Frontend (React) y Backend (Django REST Framework)
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> Comunicación entre el cliente web y el servidor de aplicaciones mediante servicios RESTful.</li> <li>• <b>Propósito:</b> Permitir el intercambio estructurado de datos (peticiones y respuestas) entre la interfaz de usuario y la lógica de negocio.</li> <li>• <b>Definición del interfaz:</b> <ul style="list-style-type: none"> <li>– Protocolo: HTTP/HTTPS</li> <li>– Formato de datos: JSON</li> <li>– Métodos principales: GET (consultas), POST (registro), PUT/PATCH (modificación), DELETE (eliminación)</li> <li>– Autenticación: Tokens JWT (JSON Web Tokens) enviados en encabezados HTTP</li> </ul> </li> </ul>

Número de requisito	RSW-02
Nombre de requisito	Interfaz con base de datos PostgreSQL
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Diseño lógico de datos
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> Conexión directa entre Django ORM y la base de datos PostgreSQL para operaciones CRUD sobre las tablas del sistema.</li> <li>• <b>Propósito:</b> Almacenar, consultar y actualizar la información de usuarios, pacientes, odontólogos, citas, fichas clínicas y configuraciones.</li> <li>• <b>Definición del interfaz:</b> <ul style="list-style-type: none"> <li>– Gestor de base de datos: PostgreSQL 15+</li> <li>– Conexión: Controlada mediante archivo de configuración settings.py</li> <li>– Formato: Transacciones SQL gestionadas a través del ORM de Django</li> <li>– Integridad: Validada mediante claves primarias, foráneas y restricciones de integridad referencial</li> </ul> </li> </ul>

Número de requisito	RSW-03
Nombre de requisito	Interfaz con API externa Twilio (WhatsApp)
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de notificaciones
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> Integración con el servicio externo Twilio API for WhatsApp, que permite enviar mensajes automatizados a los pacientes.</li> </ul>

	<ul style="list-style-type: none"> <li>• <b>Propósito:</b> Facilitar el envío de recordatorios, confirmaciones o cancelaciones de citas de forma automática.</li> <li>• <b>Definición del interfaz:</b> <ul style="list-style-type: none"> <li>– Protocolo: HTTPS</li> <li>– Formato: JSON</li> <li>– Métodos: POST (envío de mensajes)</li> <li>– Autenticación: Mediante credenciales de API Key y Token de Twilio</li> <li>– Dependencia externa: Requiere disponibilidad y conectividad con el servicio Twilio</li> </ul> </li> </ul>
--	---

### 3.1.4 Interfaces de comunicación

Número de requisito	RCM-01
Nombre de requisito	Comunicación interna entre frontend y backend
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura cliente-servidor
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> El frontend desarrollado en React se comunica con el backend de Django REST Framework a través de solicitudes HTTP/HTTPS siguiendo el estándar API REST.</li> <li>• <b>Protocolo:</b> HTTPS (preferente), HTTP (solo en entorno de desarrollo).</li> <li>• <b>Formato de datos:</b> JSON.</li> <li>• <b>Autenticación:</b> Mediante tokens JWT (JSON Web Tokens) enviados en los encabezados de las solicitudes.</li> <li>• <b>Propósito:</b> Permitir el intercambio de información entre la interfaz de usuario y la lógica de negocio del sistema.</li> </ul>

Número de requisito	RCM-02
Nombre de requisito	Comunicación con base de datos PostgreSQL
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura cliente-servidor
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> El backend Django se comunica con la base de datos PostgreSQL mediante el ORM (Object-Relational Mapper) de Django.</li> <li>• <b>Protocolo:</b> Conexión TCP/IP local o remota segura.</li> <li>• <b>Formato de datos:</b> SQL estructurado a través de consultas ORM.</li> <li>• <b>Propósito:</b> Lectura, escritura, modificación y eliminación de registros (CRUD) de usuarios, citas, pacientes y fichas médicas.</li> </ul>

Número de requisito	RCM-03
Nombre de requisito	Comunicación externa con API Twilio (WhatsApp)
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de notificaciones
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional
Detalles	<ul style="list-style-type: none"> <li>• <b>Descripción:</b> Comunicación entre el sistema OralFlow y el servicio Twilio API for WhatsApp, utilizada para el envío automático de notificaciones y recordatorios de citas.</li> <li>• <b>Protocolo:</b> HTTPS.</li> </ul>

	<ul style="list-style-type: none"> <li>• <b>Formato de datos:</b> JSON.</li> <li>• <b>Métodos:</b> POST para envío de mensajes; manejo de respuestas de confirmación o error desde Twilio.</li> <li>• <b>Seguridad:</b> Autenticación mediante credenciales (API Key y Auth Token) configuradas en variables de entorno (.env).</li> <li>• <b>Propósito:</b> Automatizar la comunicación con pacientes para confirmar, cancelar o reprogramar citas odontológicas.</li> </ul>
--	---

## 3.2 Requisitos funcionales

### 3.2.1 Requisito funcional 1

Número de requisito	RF-01
Nombre de requisito	Autenticación por rol
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Análisis funcional del sistema OralFlow
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

#### Descripción del requisito

**Definición:** El sistema debe permitir a los usuarios iniciar sesión según su rol asignado (Administrador, Administrador Clínico, Odontólogo o Paciente). Cada rol determinará los permisos, vistas y funcionalidades disponibles dentro del sistema.

#### Entradas:

- Correo electrónico del usuario.
- Contraseña asociada a la cuenta.

#### Validación de entradas:

- Comprobación de que ambos campos (correo y contraseña) sean obligatorios.
- Validación de formato de correo electrónico.
- Verificación de existencia del usuario en la base de datos.
- Comparación de la contraseña ingresada con la contraseña almacenada cifrada con algoritmo pbkdf2\_sha256.
- Comprobación de que el usuario esté activo (is\_active = True).

#### Secuencia de operaciones:

1. El usuario accede a la pantalla de inicio de sesión.
2. Ingresa su correo y contraseña.
3. El sistema valida las credenciales contra la base de datos.
4. Si las credenciales son válidas, el sistema identifica el rol del usuario.
5. El sistema genera un token JWT con la información del usuario y su rol.
6. Se redirige al usuario a la interfaz correspondiente según su rol:
  - Administrador: Panel general de gestión.
  - Administrador Clínico: Panel operativo.
  - Odontólogo: Agenda personal.
  - Paciente: Portal de citas.

#### Parámetros involucrados:

- email (string, requerido)
- password (string, requerido)
- id\_rol (integer, asignado automáticamente al usuario)

#### Respuesta a situaciones anormales:

- Si las credenciales son incorrectas: "Correo o contraseña inválidos."
- Si la cuenta está inactiva: "Su usuario ha sido desactivado, contacte al administrador."

- Si el servidor o base de datos no responde: “Error de conexión, intente nuevamente más tarde.”

**Salidas generadas:**

- Token JWT válido con duración limitada (1 hora).
- Redirección a la interfaz correspondiente según el rol del usuario.
- Registro de inicio de sesión exitoso o fallido en la bitácora del sistema.

**Relación entre entradas y salidas:**

- Entradas válidas (email + contraseña): Token generado y acceso autorizado.
- Entradas inválidas: Mensaje de error sin generación de token.

**Información almacenada en la base de datos:**

- Tabla: usuario
  - id\_usuario (PK, entero)
  - email (texto, único)
  - password (texto cifrado)
  - id\_rol (FK, entero)
  - is\_active (booleano)
  - last\_login (datetime)

**Dependencias:**

- Django REST Framework para la autenticación.
- Biblioteca djangorestframework-simplejwt para generación y validación de tokens JWT.

### 3.2.2 Requisito funcional 2

Número de requisito	RF-02
Nombre de requisito	Recuperar cuenta de usuario
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Análisis funcional del sistema OralFlow
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los usuarios recuperar el acceso a su cuenta en caso de pérdida u olvido de credenciales, mediante un mecanismo de restablecimiento de contraseña enviado al correo electrónico registrado.

**Entradas:**

- Dirección de correo electrónico registrada por el usuario.

**Validación de entradas:**

- Verificación de que el campo de correo no esté vacío.
- Comprobación del formato válido del correo electrónico.
- Confirmación de que el correo exista en la base de datos de usuarios.

**Secuencia de operaciones:**

1. El usuario selecciona la opción “¿Olvidó su contraseña?” en la pantalla de inicio de sesión.
2. El sistema solicita el correo electrónico asociado a la cuenta.
3. El usuario ingresa su correo y envía la solicitud.
4. El sistema verifica si el correo está registrado y activo.
5. Si es válido, el sistema genera un token temporal de restablecimiento.
6. Se envía un correo automático al usuario con un enlace seguro para restablecer su contraseña.
7. El usuario accede al enlace y define una nueva contraseña
8. El sistema actualiza la contraseña cifrada en la base de datos y confirma el cambio.

**Parámetros involucrados:**

- email (string, requerido)
- new\_password (string, requerido)

**Respuesta a situaciones anormales:**

- Si el token ha expirado o es inválido: "El enlace de recuperación no es válido o ha caducado."

**Salidas generadas:**

- Envío de un correo electrónico con el enlace de recuperación.

**Relación entre entradas y salidas:**

- Entradas válidas (email): Token generado y correo enviado.
- Entradas inválidas (email no registrado): Correo no enviado.

**Información almacenada en la base de datos:**

- Tabla: usuario
  - id\_usuario (PK, entero)
  - email (texto, único)
  - password (texto cifrado)

**Dependencias:**

- Servicio SMTP para el envío de correos automáticos.
- Token JWT temporal para la validación del enlace de recuperación.

### 3.2.3 Requisito funcional 3

Número de requisito	RF-03
Nombre de requisito	Gestión de administradores clínicos
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de administradores
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir al Administrador principal crear, editar, consultar y desactivar administradores clínicos, quienes tienen acceso operativo al sistema para gestionar citas, pacientes, odontólogos y reportes, pero sin privilegios de administración global.

**Entradas:**

- Datos personales del administrador clínico: nombres, apellidos, cédula, correo, teléfono.
- Credenciales de acceso (correo electrónico y contraseña).

**Validación de entradas:**

- Todos los campos obligatorios deben completarse.
- Validación de formato de los campos obligatorios.

**Secuencia de operaciones:**

1. El Administrador accede al módulo "Gestión de Administradores Clínicos".
2. Selecciona la opción "Registrar nuevo administrador clínico".
3. Completa los datos requeridos en el formulario y guarda el registro.
4. El sistema valida los datos ingresados y crea el nuevo usuario.
5. El Administrador puede editar información o desactivar un registro existente cuando sea necesario.

**Parámetros involucrados:**

- Campos de información de la tabla usuario.

**Respuesta a situaciones anormales:**

- Si los campos obligatorios no se ingresan o no tienen el formato válido, muestra un mensaje de error resaltando los campos incorrectos.

**Salidas generadas:**

- Confirmación visual del sistema: "Administrador clínico creado correctamente."

**Relación entre entradas y salidas:**

- Entradas válidas: Creación o actualización exitosa del usuario.
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: usuario

- id\_usuario (PK)
- id\_rol (FK)
- primer\_nombre, segundo\_nombre, primer\_apellido, segundo\_apellido
- email
- password
- celular
- created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de usuarios.
- Django ORM para manejo de registros en base de datos.

### 3.2.4 Requisito funcional 4

Número de requisito	RF-04
Nombre de requisito	Gestión de odontólogos
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de odontólogos
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los Administradores (principal y clínico) registrar, modificar, desactivar y consultar odontólogos en la base de datos del consultorio.

**Entradas:**

- Datos personales: nombres, apellidos, cédula, sexo, correo, teléfono, fecha de nacimiento.
- Información profesional: especialidad y universidad.
- Datos de disponibilidad: horarios laborales, consultorio asignado.

**Validación de entradas:**

- Todos los campos obligatorios deben completarse.
- Validación de formato de los campos obligatorios.

**Secuencia de operaciones:**

1. El administrador accede al módulo "Gestión de Odontólogos".
2. Selecciona "Agregar nuevo odontólogo" y completa el formulario.
3. El sistema valida los datos y crea el registro en la base de datos.
4. El administrador puede editar los datos o desactivar al odontólogo.
5. El odontólogo desactivado deja de aparecer como disponible en la agenda.
6. Se actualiza automáticamente la lista de odontólogos activos.

**Parámetros involucrados:**

- Campos de información de la tabla usuario, odontólogo y odontólogo\_horario.

**Respuesta a situaciones anormales:**

- Si los campos obligatorios no se ingresan o no tienen el formato válido, muestra un mensaje de error resaltando los campos incorrectos.

**Salidas generadas:**

- Confirmación visual del sistema: "Odontólogo creado correctamente."

**Relación entre entradas y salidas:**

- Entradas válidas: Creación o actualización exitosa del usuario.
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: usuario
  - id\_usuario (PK)
  - id\_rol (FK)
  - primer\_nombre, segundo\_nombre, primer\_apellido, segundo\_apellido
  - email

- password
- celular
- created\_at, updated\_at
- Tabla: odontologo
  - id\_odontologo (PK)
  - id\_consultorio\_defecto (FK)
  - created\_at, updated\_at
- Tabla: odontologo\_horario
  - id\_odontologo (FK)
  - dia\_semana, hora\_inicio, hora\_fin
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de usuarios.
- Django ORM para manejo de registros en base de datos.

### 3.2.5 Requisito funcional 5

Número de requisito	RF-05
Nombre de requisito	Gestión de pacientes
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de pacientes
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los administradores, administradores clínicos y odontólogos registrar, modificar, desactivar y consultar pacientes dentro de la aplicación.

**Entradas:**

- Datos personales: nombres, apellidos, cédula, sexo, correo, teléfono, fecha de nacimiento.
- Información clínica: antecedentes médicos, alergias, enfermedades hereditarias, observaciones generales.

**Validación de entradas:**

- Todos los campos obligatorios deben completarse.
- Validación de formato de los campos obligatorios.

**Secuencia de operaciones:**

1. El actor (administrador, administrador clínico u odontólogo) accede al módulo "Gestión de Pacientes".
2. Selecciona la opción "Registrar nuevo paciente".
3. Completa los datos personales y clínicos del paciente.
4. El sistema valida la información ingresada.
5. Si es correcta, se registra el paciente y se genera su usuario.
6. El sistema permite editar la información o desactivar el registro cuando sea necesario.
7. Los cambios se reflejan inmediatamente en la lista de pacientes activos.

**Parámetros involucrados:**

- Campos de información de la tabla usuario y paciente.

**Respuesta a situaciones anormales:**

- Si los campos obligatorios no se ingresan o no tienen el formato válido, muestra un mensaje de error resaltando los campos incorrectos.

**Salidas generadas:**

- Confirmación visual del sistema: "Paciente creado correctamente."

**Relación entre entradas y salidas:**

- Entradas válidas: Creación o actualización exitosa del usuario.
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: usuario

- id\_usuario (PK)
- id\_rol (FK)
- primer\_nombre, segundo\_nombre, primer\_apellido, segundo\_apellido
- email
- password
- celular
- created\_at, updated\_at
- Tabla: paciente
  - id\_paciente (PK)
  - contacto\_emergencia\_nom
  - contacto\_emergencia\_par
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de usuarios.
- Django ORM para manejo de registros en base de datos.

### 3.2.6 Requisito funcional 6

Número de requisito	RF-06
Nombre de requisito	Gestión de especialidades y consultorios
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de configuración del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los administradores y administradores clínicos registrar, modificar, consultar y desactivar tanto especialidades odontológicas como consultorios físicos.

**Entradas:**

- Para especialidades: Nombre de la especialidad.
- Para consultorios: Número, descripción, estado (activo/inactivo).

**Validación de entradas:**

- Todos los campos obligatorios deben completarse.
- Validación de formato de los campos obligatorios.

**Secuencia de operaciones:**

1. El actor accede al módulo “Configuración del sistema”.
2. Selecciona la sección correspondiente: “Especialidades” o “Consultorios”.
3. Visualiza la lista de registros existentes.
4. Selecciona la opción “Agregar nuevo”, completa los datos y guarda.
5. El sistema valida los datos y crea el registro.
6. El usuario puede editar o desactivar un registro existente.

**Parámetros involucrados:**

- Campos de información de la tabla especialidades y consultorio.

**Respuesta a situaciones anormales:**

- Si el nombre o código ya existen: “La especialidad ya se encuentra registrada.”
- Si falta información: “Complete los campos obligatorios antes de guardar.”

**Salidas generadas:**

- Confirmación visual: “Registro creado correctamente.”

**Relación entre entradas y salidas:**

- Entradas válidas: Creación o actualización exitosa del registro.
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: especialidad
  - id\_especialidad (PK)
  - nombre

- created\_at, updated\_at
- Tabla: consultorio
  - id\_consultorio (PK)
  - numero
  - descripcion
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de consultorios y especialidades.
- Django ORM para manejo de registros en base de datos.

### 3.2.7 Requisito funcional 7

Número de requisito	RF-07
Nombre de requisito	Gestión de bloqueos de agenda
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de configuración del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los administradores y administradores clínicos registrar, modificar y eliminar bloqueos de agenda que impiden el agendamiento de citas en fechas específicas.

Los bloqueos pueden ser globales (afectan a todos los odontólogos) o individuales (afectan solo a un odontólogo determinado).

**Entradas:**

- Fecha del bloqueo.
- Indicador de bloqueo recurrente anual.
- Motivo del bloqueo.
- Odontólogo (en caso de ser un bloqueo específico)

**Validación de entradas:**

- La fecha es obligatoria y debe tener formato válido (YYYY-MM-DD).
- El campo motivo debe contener una descripción no vacía.
- Validación de formato de los campos obligatorios.

**Secuencia de operaciones:**

1. Los administradores acceden al módulo “Gestión de bloqueos de agenda”.
2. En el calendario interactivo seleccionan la fecha para el bloqueo.
3. Ingresa el motivo, tipo (global o por odontólogo) y marca si es recurrente anual.
4. El sistema valida la información y registra el bloqueo.
5. La fecha bloqueada se muestra visualmente en el calendario del sistema.
6. El usuario puede editar o eliminar bloqueos existentes.

**Parámetros involucrados:**

- Campos de información de la tabla bloqueo\_dia.

**Respuesta a situaciones anormales:**

- Si ya existe un bloqueo para esa fecha y odontólogo: “Ya existe un bloqueo para esta fecha.”
- Si se agrega un bloqueo para fechas con citas agendadas: “Existen citas por atender en el día seleccionado, debe modificar las citas para continuar”

**Salidas generadas:**

- Confirmación visual del registro o modificación: “Bloqueo registrado correctamente.”

**Relación entre entradas y salidas:**

- Entradas válidas: Registro o modificación exitosa.
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: bloqueo\_dia
  - id\_bloqueo (PK)
  - fecha
  - recurrente\_anual
  - motivo
  - id\_odontologo (FK)
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de bloqueos de agenda.
- Django ORM para manejo de registros en base de datos.

### 3.2.8 Requisito funcional 8

Número de requisito	RF-08
Nombre de requisito	Agendamiento de citas
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de citas
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir que pacientes, odontólogos o personal administrativo (administrador clínico) puedan agendar citas odontológicas en función de la disponibilidad del odontólogo, la configuración de su horario y los bloqueos existentes en la agenda.

**Entradas:**

- Fecha y hora solicitada.
- Motivo o tipo de atención.
- Odontólogo seleccionado.

**Validación de entradas:**

- La fecha y hora deben encontrarse dentro del horario laboral del odontólogo.
- No debe existir otra cita activa en la misma franja horaria.
- No puede existir bloqueo de agenda para esa fecha u hora.

**Secuencia de operaciones:**

1. El actor (paciente, odontólogo o administrador clínico) accede al módulo "Agenda / Citas".
2. Selecciona una fecha disponible en el calendario.
3. El sistema consulta la disponibilidad del odontólogo y muestra los horarios libres.
4. El actor selecciona la hora y el consultorio, e ingresa el motivo de la cita.
5. El sistema valida que la franja esté libre y que no exista bloqueo.
6. Se crea el registro de la cita con estado inicial "pendiente".

**Parámetros involucrados:**

- Campos de información de la tabla cita.

**Respuesta a situaciones anormales:**

- Si la hora no está disponible: "El horario seleccionado no se encuentra disponible."
- Si existe un bloqueo: "No se pueden agendar citas en fechas bloqueadas."
- Si falta algún campo obligatorio: "Complete todos los campos requeridos antes de continuar."

**Salidas generadas:**

- Confirmación visual: "Cita registrada correctamente."
- Actualización inmediata del calendario de citas.

**Relación entre entradas y salidas:**

- Entradas válidas: Cita creada con estado "pendiente"
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: cita
  - id\_cita (PK)
  - id\_usuario (FK)
  - id\_odontologo (FK)
  - id\_consultorio (FK)
  - fecha
  - hora
  - motivo
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de citas.
- Django ORM para manejo de registros en base de datos.

### 3.2.9 Requisito funcional 9

Número de requisito	RF-09
Nombre de requisito	Reprogramación y cancelación de citas
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de citas
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a pacientes, odontólogos y administradores clínicos reprogramar o cancelar citas odontológicas, garantizando el cumplimiento de las reglas de negocio establecidas.

La reprogramación debe conservar la trazabilidad de la cita original, y la cancelación debe actualizar el estado del registro y liberar la franja horaria correspondiente en la agenda.

**Entradas:**

- Fecha y hora solicitada.
- Motivo o tipo de atención.
- Odontólogo seleccionado.

**Validación de entradas:**

- Identificador de la cita a modificar.
- Nueva fecha y hora (en caso de reprogramación).
- Motivo de reprogramación o cancelación (opcional).
- Identificador del usuario que realiza la acción (paciente, odontólogo o administrador).

**Secuencia de operaciones:**

1. El usuario accede al módulo "Citas" y selecciona una cita existente.
2. El sistema muestra las opciones disponibles: Reprogramar o Cancelar.
3. Si elige Reprogramar, selecciona una nueva fecha y hora disponibles.
4. El sistema valida la disponibilidad y guarda los cambios, manteniendo un registro histórico.
5. Si elige Cancelar, se solicita confirmación del usuario.
6. El sistema cambia el estado de la cita a "cancelada", libera el espacio en la agenda y genera un registro de auditoría.

**Parámetros involucrados:**

- Campos de información de la tabla cita.

**Respuesta a situaciones anormales:**

- Si la nueva fecha está bloqueada: "No se puede reprogramar en una fecha bloqueada."
- Si hay otra cita en la misma hora: "El horario seleccionado no está disponible."

**Salidas generadas:**

- Confirmación visual.
- Actualización inmediata del calendario de citas.

**Relación entre entradas y salidas:**

- Entradas válidas: Cita actualizada correctamente
- Entradas inválidas: Mensaje de error y no se realiza la operación.

**Información almacenada en la base de datos:**

- Tabla: cita
  - id\_cita (PK)
  - id\_usuario (FK)
  - id\_odontologo (FK)
  - id\_consultorio (FK)
  - fecha
  - hora
  - motivo
  - estado
  - created\_at, updated\_at

**Dependencias:**

- Django REST Framework para gestión de citas.
- Django ORM para manejo de registros en base de datos.

### 3.2.10 Requisito funcional 10

Número de requisito	RF-10
Nombre de requisito	Confirmación de citas
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de citas
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los pacientes confirmar su asistencia a una cita odontológica tanto desde la aplicación web como mediante la notificación enviada por WhatsApp, garantizando la sincronización inmediata con la base de datos y el calendario del odontólogo.

**Entradas:**

- Identificador único de la cita (id\_cita).
- Acción del paciente (Confirmar / No asistir).
- Medio de confirmación (Aplicación Web o WhatsApp).

**Validación de entradas:**

- La cita debe encontrarse en estado "pendiente".
- La fecha de la cita debe ser futura (no se pueden confirmar citas pasadas).
- Validar que la confirmación proviene de un paciente asociado a esa cita.

**Secuencia de operaciones:**

1. El sistema envía una notificación de recordatorio al paciente vía WhatsApp o muestra la cita pendiente en su panel web.
2. El paciente selecciona "Confirmar asistencia" o "Cancelar".
3. Si confirma, el sistema cambia el estado de la cita a "confirmada".
4. Si selecciona "Cancelada", la cita se marca como "Cancelada" y la franja se libera.
5. Se registra la fecha, hora y medio de confirmación.

**Parámetros involucrados:**

- Campos de información de la tabla cita.

**Respuesta a situaciones anormales:**

- Si la cita ya fue confirmada: "Esta cita ya fue confirmada anteriormente."
- Si la cita ya fue cancelada: "No se puede confirmar una cita cancelada."

**Salidas generadas:**

- Confirmación visual o mensaje en WhatsApp: "Cita confirmada exitosamente."

**Relación entre entradas y salidas:**

- Entradas válidas: Cita actualizada a estado "confirmada" o "cancelada"

- Entradas inválidas: Mensaje de error y no se realiza la operación.

#### Información almacenada en la base de datos:

- Tabla: cita
  - id\_cita (PK)
  - id\_usuario (FK)
  - id\_odontologo (FK)
  - id\_consultorio (FK)
  - fecha
  - hora
  - motivo
  - estado
  - updated\_at

#### Dependencias:

- Django REST Framework para gestión de citas.
- Django ORM para manejo de registros en base de datos.
- Módulo notificaciones con Twilio API para WhatsApp.

### 3.2.11 Requisito funcional 11

Número de requisito	RF-11
Nombre de requisito	Notificaciones y recordatorios
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de notificaciones del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

#### Descripción del requisito

**Definición:** El sistema debe enviar recordatorios automáticos de citas a los pacientes mediante mensajes de WhatsApp, utilizando la API de Twilio. Los recordatorios se enviarán de forma programada con base en la fecha y hora de la cita, permitiendo al paciente confirmar o cancelar directamente desde el mensaje.

#### Entradas:

- Identificador único de la cita (id\_cita).
- Número de teléfono del paciente (en formato internacional).
- Plantilla de mensaje predefinida aprobada por WhatsApp Business.

#### Validación de entradas:

- La cita debe encontrarse en estado "pendiente".
- Verificar que el paciente tenga un número de teléfono válido y con código de país.
- Comprobar que la cita esté en estado "pendiente".

#### Secuencia de operaciones:

1. Un proceso automatizado (cron job o scheduler) revisa periódicamente las citas próximas.
2. Para cada cita, el sistema genera el mensaje de recordatorio utilizando una plantilla predefinida.
3. Se envía el mensaje mediante la API de Twilio, incluyendo botones o enlaces para confirmar o cancelar.
4. Twilio devuelve una respuesta HTTP indicando el éxito o error del envío.
5. El sistema actualiza el estado del recordatorio.

#### Parámetros involucrados:

- Campos de información de la tabla cita.

#### Respuesta a situaciones anormales:

- Si la cita ya fue confirmada: "Esta cita ya fue confirmada anteriormente."
- Si la cita ya fue cancelada: "No se puede confirmar una cita cancelada."

#### Salidas generadas:

- Confirmación del envío exitoso de mensaje.
- Registro en la bitácora con fecha, hora, destinatario y estado.
- Actualización del registro de la cita con la información del envío.

**Relación entre entradas y salidas:**

- Entradas válidas: Mensaje enviado y registrado.
- Entradas inválidas: Reintento y registro de error.

**Información almacenada en la base de datos:**

- Tabla: cita
  - id\_cita (PK)
  - id\_paciente (FK)
  - fecha, hora
  - estado
  - whatsapp\_message\_sid
  - batch\_id
  - recordatorio\_enviado\_at
  - confirmacion\_fuente
  - updated\_at

**Dependencias:**

- Twilio API for WhatsApp para el envío y recepción de mensajes.
- Django REST Framework para gestión de citas.
- Cron Jobs para ejecución programada de envíos automáticos.

### 3.2.12 Requisito funcional 12

Número de requisito	RF-12
Nombre de requisito	Visualización de agenda
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de gestión de citas del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los odontólogos, administradores y administradores clínicos visualizar la agenda de citas en formatos de vista diaria y mensual, mostrando información detallada sobre cada cita, su estado, paciente, consultorio y hora programada.

**Entradas:**

- Filtro de fecha

**Validación de entradas:**

- Validar que la fecha esté en formato correcto

**Secuencia de operaciones:**

1. El usuario accede al módulo "Agenda" desde su perfil.
2. El sistema carga por defecto la vista diaria con las citas del día actual.
3. El sistema consulta la base de datos y filtra las citas correspondientes al rango seleccionado.
4. El usuario puede hacer clic en una cita para ver sus detalles.

**Parámetros involucrados:**

- Campos de información de la tabla cita.

**Respuesta a situaciones anormales:**

- Si no existen citas en el rango: "No hay citas registradas en el período seleccionado."

**Salidas generadas:**

- Información por cita: hora, paciente, estado, motivo, y consultorio.
- Visualización gráfica de la agenda con citas ordenadas por hora y consultorio.

**Relación entre entradas y salidas:**

- Entradas válidas: Carga exitosa de citas.
- Entradas inválidas: Mensaje de error y agenda vacía.

**Información almacenada en la base de datos:**

- Tabla: cita
  - id\_cita (PK)

- id\_paciente (FK)
- fecha, hora, motivo, estado
- created\_at, updated\_at

**Dependencias:**

- Módulo de citas
- Django REST Framework para gestión de citas.

### 3.2.13 Requisito funcional 13

Número de requisito	RF-13
Nombre de requisito	Registro de fichas médicas
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de atención clínica del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir al odontólogo registrar una ficha médica digital asociada a cada cita confirmada, en la cual se consignent la observación clínica, diagnóstico, tratamiento y comentarios adicionales.

**Entradas:**

- Identificador de la cita (id\_cita).
- Texto de observación.
- Diagnóstico odontológico.
- Tratamiento propuesto o aplicado.

**Validación de entradas:**

- La cita asociada debe estar en estado “confirmada”.
- Los campos diagnóstico, observación y tratamiento son obligatorios.

**Secuencia de operaciones:**

1. El odontólogo accede al módulo “Atención Clínica” desde la cita confirmada.
2. El sistema carga los datos del paciente y muestra el formulario de ficha médica.
3. El odontólogo completa los campos de observación, diagnóstico y tratamiento.
4. El sistema valida la información ingresada.
5. Al guardar, la ficha médica se registra y se asocia automáticamente a la cita.
6. El sistema actualiza el estado de la cita a “realizada” y registra la fecha de atención.

**Parámetros involucrados:**

- Campos de información de la tabla ficha\_medica.

**Respuesta a situaciones anormales:**

- Si ya existe una ficha asociada a la cita: “La cita ya tiene una ficha médica registrada.”
- Si la cita no está confirmada: “Solo se pueden registrar fichas para citas confirmadas o realizadas.”
- Si faltan campos requeridos: “Complete todos los campos obligatorios antes de guardar.”

**Salidas generadas:**

- Entradas válidas: “Ficha médica registrada correctamente.”
- Entradas inválidas: del estado de la cita a “realizada”.

**Relación entre entradas y salidas:**

- Entradas válidas: Creación de ficha médica exitosa.
- Entradas inválidas: Mensaje de error y no se cancela la operación.

**Información almacenada en la base de datos:**

- Tabla: ficha\_medica
  - id\_ficha\_medica (PK)

- observación, diagnóstico, tratamiento, comentarios
- archivos\_adjuntos
- created\_at, updated\_at

**Dependencias:**

- Módulo de citas
- Django REST Framework para gestión de citas.

### 3.2.14 Requisito funcional 14

Número de requisito	RF-14
Nombre de requisito	Consulta de historial médico
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de atención clínica del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

**Descripción del requisito**

**Definición:** El sistema debe permitir a los odontólogos y pacientes consultar el historial clínico digital, que incluye las fichas médicas registradas.

**Entradas:**

- Rango de fechas (opcional).

**Validación de entradas:**

- Validar que el id\_paciente exista y esté activo.

**Secuencia de operaciones:**

1. El usuario accede al "Historial de citas".
2. El sistema carga las fichas médicas asociadas al paciente autenticado (si es paciente) o al paciente seleccionado (si es odontólogo).
3. El usuario puede aplicar filtros de búsqueda por fecha.
4. Se presentan los datos organizados cronológicamente, con los campos principales: observación, diagnóstico, tratamiento y comentarios.
5. El usuario puede visualizar los detalles de cada ficha, pero no modificarlos.

**Parámetros involucrados:**

- Campos de información de la tabla cita y ficha\_medica.

**Respuesta a situaciones anormales:**

- Si el paciente no tiene fichas registradas: "No existen fichas médicas registradas para este paciente."

**Salidas generadas:**

- Visualización estructurada del historial médico.

**Relación entre entradas y salidas:**

- Entradas válidas: Historial cargado exitosamente.
- Entradas inválidas: Mensaje de error e historial vacío.

**Información almacenada en la base de datos:**

- Tabla: ficha\_medica
- Tabla: cita
- Tabla: paciente

**Dependencias:**

- Módulo de fichas médicas
- Módulos de citas
- Django REST Framework

### 3.2.15 Requisito funcional 15

Número de requisito	RF-15
Nombre de requisito	Reportes generales
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de reportes y estadísticas del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

### Descripción del requisito

**Definición:** El sistema debe permitir a los administradores, odontólogos y pacientes visualizar estadísticas respecto a sus funciones, y además, los administradores puedan imprimir los reportes en PDF.

### Entradas:

- Rango de fechas (obligatorio).
- Filtro por odontólogo.

### Validación de entradas:

- Validar que exista al menos una cita o ficha registrada en el rango solicitado.

### Secuencia de operaciones:

1. El usuario accede al módulo "Reportes".
2. Selecciona el tipo de reporte y el rango de fechas.
3. El sistema consulta las tablas cita, odontologo y paciente.
4. Se calculan indicadores agregados.
5. El sistema genera un informe visual en pantalla con gráficos y tablas.

### Parámetros involucrados:

- Campos de información de la tabla cita, odontologo y paciente.

### Respuesta a situaciones anormales:

- Si no hay datos disponibles en el rango: "No existen registros para el período seleccionado."

### Salidas generadas:

- Reporte visual en pantalla con tablas y gráficos.
- Archivo PDF descargable para el caso de los administradores.

### Relación entre entradas y salidas:

- Entradas válidas: Reporte PDF y gráficos estadísticos.
- Entradas inválidas: Mensaje de error y gráficos vacíos.

### Información almacenada en la base de datos:

- Tabla: cita
- Tabla: paciente
- Tabla: odontologo

### Dependencias:

- Módulo de odontólogos
- Módulo de pacientes
- Módulos de citas
- Django REST Framework

## 3.2.16 Requisito funcional 16

Número de requisito	RF-16
Nombre de requisito	Perfil de odontólogo
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de odontólogos del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

### Descripción del requisito

**Definición:** El sistema debe permitir a los pacientes visualizar el perfil profesional de los odontólogos registrados en el consultorio, incluyendo su nombre completo, especialidades, experiencia, horarios de atención y fotografía de perfil.

### Entradas:

- Odontólogo registrado en el sistema

### Validación de entradas:

- Verificar que el odontólogo esté activo.

### Secuencia de operaciones:

1. El paciente entra a la sección de "Agendar cita".
2. El paciente selecciona un odontólogo.
3. Se presenta el perfil e información profesional del odontólogo seleccionado

### Parámetros involucrados:

- Campos de información de la tabla odontólogo.
- Respuesta a situaciones anormales:**
- Si el odontólogo está inactivo: “El perfil no está disponible actualmente.”
- Salidas generadas:**
- Perfil visual del odontólogo con información profesional.
- Relación entre entradas y salidas:**
- Entradas válidas: Visualización del perfil del odontólogo.
  - Entradas inválidas: Mensaje de error y perfil vacío.
- Información almacenada en la base de datos:**
- Tabla: odontologo
- Dependencias:**
- Módulo de odontólogos
  - Django REST Framework

### 3.3 Requisitos no funcionales

#### 3.3.1 Requisitos de rendimiento

Número de requisito	RNF-01
Nombre de requisito	El 95% de las solicitudes HTTPS de deben completar en menos de 2 segundos
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

##### Descripción del requisito

**Definición:** El sistema debe garantizar que el 95% de las solicitudes HTTPS, incluyendo operaciones de inicio de sesión, consultas, registros y actualizaciones, se completen en menos de 2 segundos bajo condiciones normales de carga de uso.

##### Criterios de rendimiento medibles:

- Tiempo máximo de respuesta promedio: 2 segundos para el 95% de las peticiones HTTPS.
- Las operaciones de lectura (consultas de agenda, historial, odontólogos, etc.) deben completarse en un tiempo promedio de 1 segundo.
- Las operaciones de escritura (registro, actualización o eliminación de datos) no deben superar los 2 segundos.
- Las operaciones de autenticación y verificación de tokens no deben exceder 1.5 segundos.

##### Escenario operativo considerado:

- Consultorio odontológico con hasta 31 usuarios concurrentes (entre pacientes, odontólogos y personal administrativo).
- Volumen de operaciones típico:
  - 10 inicios de sesión por minuto.
  - 30 consultas de agenda o historial por minuto.
  - 10 operaciones de actualización o creación de citas por minuto.

**Validaciones técnicas:** Todas las solicitudes deben realizarse sobre protocolo HTTPS, garantizando integridad y seguridad de la conexión.

**Pruebas de rendimiento recomendadas:** Pruebas de carga mediante herramientas como Apache JMeter, simulando usuarios concurrentes.

### 3.3.2 Seguridad

Número de requisito	RNF-02
Nombre de requisito	El sistema debe garantizar la protección integral de la información clínica y administrativa, evitando accesos no autorizados, alteraciones o pérdidas de datos
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

#### Descripción del requisito

**Definición:** La aplicación implementará autenticación basada en roles, tokens JWT (JSON Web Token) para el control de sesiones y cifrado de contraseñas para salvaguardar las credenciales de los usuarios.

#### Mecanismos de seguridad implementados:

- **Autenticación y autorización por roles:** Cada usuario pertenece a un rol específico (Administrador, Administrador Clínico, Odontólogo o Paciente).
- **Cifrado de contraseñas:** Todas las contraseñas se almacenan en la base de datos encriptadas mediante el algoritmo PBKDF2 provisto por Django, con salt aleatorio por usuario.
- **Tokens de autenticación (JWT):** Las sesiones y solicitudes al backend usan JWT, asegurando comunicación cifrada mediante el protocolo HTTPS.
- **Comunicación segura:** Toda la comunicación entre frontend (React) y backend (Django) se realiza bajo el protocolo HTTPS, previniendo interceptaciones o manipulaciones de datos en tránsito.
- **Logs de actividad y auditoría:** Se registra la creación, modificación o eliminación de usuarios, citas y fichas médicas, incluyendo fecha, hora y responsable.

### 3.3.3 Fiabilidad

Número de requisito	RNF-03
Nombre de requisito	El sistema debe garantizar una operación continua y estable, minimizando la ocurrencia de fallos durante su ejecución.
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/Opcional

#### Criterios de fiabilidad medibles:

- **MTBF (Mean Time Between Failures):**

$$MTBF = \frac{A}{B}$$

Donde:

A = Tiempo total de operación (horas)

B = Número de fallos ocurridos.

Se espera un valor MTBF  $\geq 50$ , reflejando estabilidad operativa durante las jornadas laborales

- **Cobertura de pruebas (Test Coverage):**

$$TC = \frac{A}{B}$$

Donde:

A = Casos de prueba ejecutados

B = Casos de prueba planificados.

El sistema espera un valor TC  $\geq 90$ , asegurando ausencia de errores críticos antes del despliegue.

### 3.3.4 Disponibilidad

Número de requisito	RNF-04
Nombre de requisito	El sistema debe permanecer disponible para su uso al menos el 98 % del tiempo programado
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Módulo de odontólogos del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

- System Availability (SA):

$$SA = \frac{A}{B}$$

Donde:

A = Tiempo efectivamente disponible

B = Tiempo total programado.

Se espera un valor  $SA \geq 0.98$ , garantizando alta continuidad del servicio.

### 3.3.5 Mantenibilidad

Número de requisito	RNF-05
Nombre de requisito	El backend debe estar estructurado modularmente mediante Django Apps para facilitar el mantenimiento y corrección de errores
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

#### Descripción del requisito

**Definición:** La mantenibilidad se asegura mediante una arquitectura modular basada en aplicaciones independientes de Django, lo que permite realizar actualizaciones, correcciones o ampliaciones específicas sin comprometer la estabilidad global del sistema.

#### Responsables del mantenimiento:

- El mantenimiento correctivo, adaptativo y evolutivo será responsabilidad del desarrollador del sistema.
- Los usuarios administrativos pueden realizar tareas básicas de mantenimiento funcional desde la interfaz (configuración de horarios, bloqueo de agenda, gestión de usuarios.) sin acceso al código.

### 3.3.6 Portabilidad

Número de requisito	RNF-06
Nombre de requisito	El sistema debe poder ejecutarse y desplegarse en diferentes plataformas o entornos operativos sin requerir modificaciones sustanciales del código.
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Esencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

#### Descripción del requisito

**Definición:** El backend desarrollado en Django deberá ser totalmente compatible con los sistemas operativos Windows y Linux, mientras que el frontend desarrollado con React deberá poder ejecutarse correctamente en los principales navegadores web modernos Google Chrome, Mozilla Firefox, Microsoft Edge y Safari.

#### Atributos de portabilidad:

- El backend está desarrollado en Python (Django), lo que permite su ejecución tanto en Windows como en Linux sin modificaciones del código fuente.
- El frontend, desarrollado en React, debe ser completamente funcional y visualmente consistente en los navegadores Chrome, Firefox, Edge y Safari.
- El sistema puede instalarse en entornos locales, servidores institucionales o nubes públicas como Vercel y Render.

### 3.4 Otros requisitos

Número de requisito	RLEG-01
Nombre de requisito	Cumplimiento de normativa sanitaria nacional
Tipo	<input checked="" type="checkbox"/> Requisito <input type="checkbox"/> Restricción
Fuente del requisito	Arquitectura del sistema
Prioridad del requisito	<input checked="" type="checkbox"/> Alta/Eencial <input type="checkbox"/> Media/Deseado <input type="checkbox"/> Baja/ Opcional

#### Descripción del requisito

**Definición:** El sistema debe cumplir con las directrices y formatos establecidos por el Ministerio de Salud Pública del Ecuador (MSP) en el Formulario 033 para la Atención Odontológica, particularmente, los campos y estructuras de las fichas médicas.