

**Sistema Integral de Gestión Médica**

Medicare es una plataforma web diseñada para optimizar la administración y operación de centros médicos, clínicas y hospitales. El sistema permite gestionar usuarios, citas médicas, historiales clínicos, doctores, departamentos, recursos hospitalarios y más, ofreciendo módulos especializados según el rol del usuario: paciente, doctor o administrador.

**Versión del Sistema:** 1.1

**Fecha :** 06/2025

**Desarrollador del sistema:** Ing. Víctor Camacaro

**Contacto:** [victorcamacaro253@gmail.com](mailto:victorcamacaro253@gmail.com)

**INDICE**

Medicare es un sistema web integral desarrollado para mejorar la gestión operativa y clínica de centros médicos, hospitales y clínicas. El sistema permite centralizar la administración de pacientes, doctores, citas médicas, departamentos y recursos hospitalarios, facilitando una comunicación eficiente entre los distintos actores involucrados.

El sistema MediCare es una solución web integral para centros médicos y hospitales, compuesta por dos grandes áreas:

* Área pública/informativa: brinda información institucional del centro médico (historia, servicios, especialidades, noticias, seguros, instalaciones, doctores).
* Área privada/gestionada por roles: los usuarios registrados pueden agendar citas, consultar reportes médicos, ver prescripciones; los doctores gestionan pacientes y citas; y el administrador controla todos los datos del sistema.

**Objetivo**

El propósito principal de Medicare es brindar una solución digital intuitiva y funcional que permita a los centros médicos agilizar sus procesos internos, desde la programación de citas hasta la gestión de personal médico y recursos físicos, mejorando así la experiencia tanto del paciente como del personal sanitario.

**Objetivos específicos:**

* Informar sobre los servicios del centro médico.
* Gestionar citas médicas de forma digital.
* Ofrecer paneles diferenciados para paciente, doctor y administrador.
* Automatizar el control de pacientes, citas, departamentos y recursos clínicos.

**Características Principales**

* Sitio informativo con detalles sobre servicios médicos, especialidades, noticias e instalaciones.
* Registro e inicio de sesión de usuarios con rol de paciente.
* Módulo de doctor para gestionar pacientes y citas asignadas.
* Panel completo para el administrador con control total sobre usuarios, doctores, pacientes, departamentos, cuartos, camas, turnos, roles y más.
* Sistema seguro mediante autenticación JWT.
* Arquitectura escalable basada en NestJS (Backend) y React (Frontend), con base de datos MySQL.

**¿Para quién está pensado?**

* Centros médicos pequeños y medianos
* Hospitales públicos y privados
* Clínicas especializadas
* Profesionales de la salud que requieran una herramienta de gestión digital

**Arquitectura del Sistema**

+-------------------------------------+

| 👥 Usuario Final |

+-------------------------------------+

|

v

+-------------------------------------+

| 🖥️ Frontend (React + Vite) |

+-------------------------------------+

|

v

+-------------------------------------+

| ⚙️ Backend (NestJS + JWT) |

+-------------------------------------+

|

v

+-------------------------------------+

| 🗃️ Base de Datos (MySQL) |

+-------------------------------------+

|

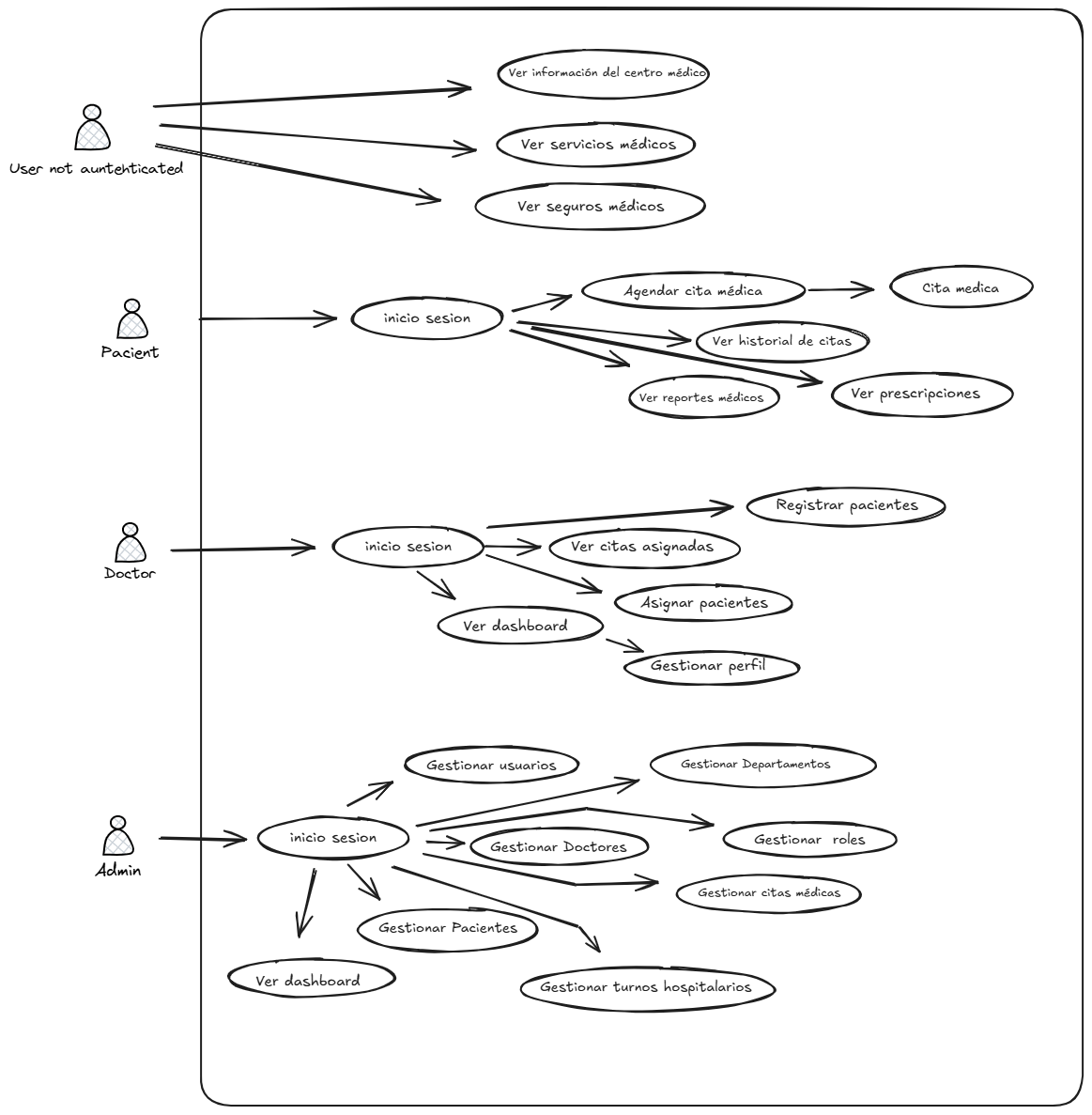
v

+-------------------------------------+

| 🧰 Herramientas & Servicios |

+-------------------------------------+

**Diagrama general del sistema**



[Inicio] → [Navegación pública]

↘ [¿Registrar/Iniciar sesión?]

→ [Registro] → [Dashboard Paciente]

→ [Login] → [Validar] → [¿Rol?]

→ [Doctor] → [Dashboard Doctor]

→ [Admin] → [Dashboard Admin]

→ [Pacientet] → [Dashboard Paciente]

← [Cerrar sesión]

← [Fin]

**🛠️ Tecnologías Utilizadas**

El sistema Medicare está desarrollado con tecnologías modernas y escalables, orientadas a brindar una experiencia eficiente tanto para el desarrollo como para el desempeño del sistema. A continuación, se detallan las principales herramientas y frameworks utilizados:

**Frontend**

* React.js: Biblioteca de JavaScript para construir interfaces de usuario dinámicas y reactivas.
* Vite: Herramienta de construcción rápida para proyectos frontend.
* TypeScript: Superset de JavaScript que añade tipado estático para mejorar la mantenibilidad del código.
* React Router: Para el manejo de rutas y navegación dentro de la aplicación.
* Tailwind CSS / Material UI: Frameworks de estilos para diseño responsivo y moderno.

**Backend**

* NestJS: Framework progresivo de Node.js para construir aplicaciones escalables y bien estructuradas.
* TypeScript: Usado en el backend para un mejor control del tipado y arquitectura limpia.
* JWT (JSON Web Token) : Para la autenticación y protección de rutas.
* Swagger/OpenAPI : Documentación interactiva de las APIs REST.
* TypeORM / Sequelize (según hayas usado): ORM para interactuar con la base de datos MySQL desde NestJS.

**Base de Datos**

* MySQL: Sistema de gestión de base de datos relacional utilizado para almacenar y gestionar toda la información del sistema.

**Otros**

* Node.js: Entorno de ejecución backend que permite correr JavaScript fuera del navegador.
* NPM / Yarn : Administrador de paquetes para la instalación y gestión de dependencias.
* Git / GitHub: Control de versiones del proyecto.

**Módulos Del Sistema y Funcionalidades**

El sistema Medicare está dividido en varios módulos que permiten una administración eficiente de los recursos médicos, usuarios y operaciones diarias en un centro médico o hospital. Cada módulo tiene funcionalidades específicas según el rol del usuario: Paciente, Doctor o Administrador.

1. **Módulo Público (Sitio Informativo)**

Este módulo es accesible sin necesidad de iniciar sesión y sirve como página principal del centro médico.

**Funcionalidades**:

* Visualización de información general del centro médico.
* Información sobre servicios médicos ofrecidos:
* Emergencias
* Hospitalización
* Cirugía
* Laboratorio
* Radiología e Imagenología
* Detalles sobre seguros médicos aceptados.
* Sección de noticias actualizadas del centro médico.
* Galería multimedia con fotos de las instalaciones.
* Historia del centro médico.
* Listado de especialidades médicas disponibles.
* Perfil profesional de los doctores registrados.

1. **Módulo de Usuarios (Pacientes)**

Los usuarios pueden registrarse y obtener automáticamente el rol de paciente, pudiendo luego acceder a su panel personal.

**Funcionalidades:**

* Registro e inicio de sesión seguro mediante autenticación JWT.
* Agendar citas médicas por especialidad y doctor disponible.
* Ver historial completo de sus citas médicas:
* Citas agendadas
* Citas completadas
* Citas canceladas
* Acceso a reportes médicos emitidos por los doctores.
* Visualización de prescripciones médicas.
* Gestionar su perfil:
* Actualizar datos personales
* Cambiar contraseña
* Ver información médica básica

1. **Módulo de Doctores**

Este módulo está diseñado exclusivamente para profesionales médicos cuyas credenciales son proporcionadas por el administrador.

**Funcionalidades:**

* Inicio de sesión con credenciales asignadas por el administrador.
* Dashboard con estadísticas generales:
* Total de pacientes asignados
* Total de citas programadas y realizadas
* Visualización de todas las citas médicas asignadas:
* Estado: pendiente, completada, cancelada
* Posibilidad de agregar nuevos pacientes al sistema.
* Asignar pacientes ya existentes a su lista personal de atención.
* Gestionar su perfil:

-Editar información personal y profesional

-Actualizar disponibilidad horaria (si aplica)

-Ver historial clínico de sus pacientes

1. **Módulo de Administrador**

El rol más completo del sistema, destinado al personal encargado de la gestión integral del centro médico.

**Funcionalidades:**

* Panel de control con indicadores clave del sistema:
* Total de pacientes
* Total de doctores
* Total de departamentos
* Estadísticas de citas médicas
* Gestión de Staff:
* Crear, leer, actualizar y eliminar empleados
* Gestión de Usuarios:
* Crear, leer, actualizar y eliminar usuarios (con rol de paciente)
* Gestión de Doctores:
* Crear, leer, actualizar y eliminar doctores
* Gestión de Pacientes:

Crear, leer, actualizar y eliminar pacientes

* Gestión de Departamentos:
* Registrar, editar, eliminar y listar los departamentos del centro médico
* Gestión de Cuartos:
* Registrar cuartos/habitaciones del hospital
* Asignar tipo de cuarto (individual, compartido, UCI, etc.)
* Gestión de Camas Clínicas:
* Registrar camas
* Asignar camas a cuartos
* Actualizar estado de ocupación
* Gestión de Puestos de Trabajo:
* Registrar puestos médicos y no médicos
* Gestión de Roles del Sistema:
* Crear, modificar y eliminar roles (paciente, doctor, admin, etc.)
* Gestión de Turnos Hospitalarios:
* Registrar turnos médicos y de enfermería
* Gestión de Especializaciones Médicas:
* Registrar, editar y eliminar especialidades médicas
* Gestión de Citas Médicas:
* Crear, modificar, eliminar y ver todas las citas del sistema

**🔒 Gestión de Autenticación y Roles**

La autenticación y autorización en el sistema Medicare está diseñada para garantizar que cada usuario tenga acceso únicamente a las funcionalidades permitidas según su rol dentro del sistema. Esto asegura la integridad, privacidad y seguridad de los datos médicos y operativos.

**Tipos de Roles**

El sistema define tres (3) roles principales:

|  |  |
| --- | --- |
| ROL | DESCRIPCION |
| Paciente | Usuario registrado que puede agendar citas, ver historial clínico y gestionar su perfil. Tiene acceso limitado a funcionalidades específicas. |
| Doctor | Profesional médico con acceso a su propio dashboard, donde puede ver sus pacientes asignados, citas programadas y gestionar información médica relacionada. |
| Administrador | Usuario con privilegios máximos. Puede gestionar usuarios, doctores, pacientes, departamentos, recursos hospitalarios y todas las configuraciones del sistema. |

Nota: El rol de usuario se asigna automáticamente al registrarse como paciente. Los roles de doctor y administrador son asignados manualmente por el administrador del sistema.

**Flujo de Autenticación**

El proceso de autenticación se basa en tokens JWT (JSON Web Token) , lo cual permite una comunicación segura entre cliente y servidor sin necesidad de mantener sesiones activas en el backend.

**Pasos del flujo:**

1. **Registro (solo para pacientes):**

* El usuario ingresa sus datos básicos (nombre, correo, contraseña).
* Se crea un registro en la tabla users con el rol de paciente.

1. **Inicio de sesión:**

* El usuario introduce su correo y contraseña.
* El backend verifica las credenciales contra la base de datos.
* Si es válido, se genera un token JWT firmado que contiene información del usuario (id, nombre, rol, etc.).
* Este token se devuelve al frontend y se almacena localmente (por ejemplo, en localStorage o cookies).
* Este token se envía al frontend mediante una cookie segura y HTTP-only , opcionalmente con atributos como Secure, HttpOnly, y SameSite.

1. **Acceso protegido:**

* Cada vez que el usuario hace una solicitud al backend, el token se incluye en el encabezado Authorization.
* El backend verifica las credenciales contra la base de datos.

1. **Cierre de sesión:**

* El backend elimina o expira la cookie que contiene el token JWT.
* Esto cierra la sesión del usuario de forma segura.

**Protección de Rutas**

Todas las rutas sensibles del sistema están protegidas mediante guards que verifican:

* Que el usuario tenga un token válido almacenado en la cookie.
* Que el rol del usuario tenga permiso para acceder a esa ruta o recurso.

**Por ejemplo:**

* Solo los usuarios con rol administrador pueden acceder a rutas como /api/doctors, /api/patients, /api/departments, etc.
* Solo los usuarios con rol doctor pueden acceder a rutas como /api/appointments/own.

**Seguridad Adicional**

* Contraseñas encriptadas: Las contraseñas de los usuarios se almacenan en la base de datos utilizando hash seguro (bcrypt).
* Token expiración: Los tokens JWT tienen un tiempo de vida limitado (ej. 24 horas), mejorando la seguridad del sistema.
* Cookies seguras: Se utilizan cookies con atributos como HttpOnly, Secure y SameSite=Strict/Lax para evitar accesos no autorizados.
* Validación de campos: Todos los endpoints validan los datos de entrada antes de procesarlos (usando DTOs y pipes de NestJS).
* Manejo de errores: Respuestas claras y controladas ante intentos de acceso no autorizado o credenciales inválidas.

🗃️ Modelo de Base de Datos (MySQL)

* users: id, nombre, email, contraseña, rol\_id, created\_at
* roles: id, nombre
* doctors: id, user\_id, especialidad\_id, ...
* patients: id, user\_id, fecha\_nacimiento, dirección, ...
* appointments: id, patient\_id, doctor\_id, fecha, estado
* departments: id, nombre, descripción
* specializations: id, nombre, descripción
* rooms: id, número, tipo, piso
* beds: id, room\_id, estado
* shifts: id, nombre, hora\_inicio, hora\_fin
* jobs: id, nombre, descripción
* news: id, título, contenido, imagen, fecha\_publicación
* medical\_records: id, appointment\_id, diagnóstico, prescripción
* login\_history
* bed\_assignments
* doctoravailability
* doctorpatients
* insurance
* pharmacy
* prescriptions
* procedures
* refresh\_tokens
* staff
* staff\_shifts