Diseño proyecto Clínica

# Objetivos

Será una **aplicación web**, que consta de un backend (API Rest) y la parte de front (SPA) desarrollada con tecnologías aprendidas en las últimas semanas.

Se pide implementar el **alcance mínimo** del proyecto que se detalla a continuación. Las **mejoras** propuestas son opcionales y se implementarán después de finalizar las funcionalidades básicas.

# Tecnologías y plazos

* **Backend**: Laravel.
* **Frontend**: Angular.

# Descripción funcional

La aplicación consiste en un sistema de gestión de citas médicas para un centro que realiza reconocimientos para empresas.

# Gestión de clientes

El administrador podrá gestionar los datos de los clientes, formados por los siguientes campos:

* Código
* Razón social
* CIF
* Dirección
* Municipio
* Provincia

**NOTA:** Al registrar el cliente se debe de asociar a un contrato

Disponer de un listado de consulta con filtrado por razón social y municipio.

Poder crear y editar los clientes.

Al crear un nuevo cliente, se genere un usuario para el cliente con rol tipo cliente. Las credenciales se le entregarán al usuario manualmente, de momento sin enviar email.

# Gestión de contratos de cliente

Al crear un cliente, se debe de generar un contrato. Los contratos tendrán duración de un año. Cuando finalice este, se deberá cerrar el contrato vigente y generar uno nuevo para el próximo año.

* El contrato constará de:
* Fecha de inicio del contrato
* Fecha de expiración del contrato
* Número de reconocimientos médicos contratos (los que se deben de realizar)

El administrador podrá consultar y gestionar estos contratos.

# Reserva de citas

El cliente o administrador podrá registrar el número de reconocimientos en una fecha determinada.

Cada reconocimiento tendrá una duración de 5 minutos.

Al realizar reserva, validar de que no puede haber más de una reserva a la misma hora.

Nota: Al registrar una cita, se debe identificar el contrato vigente del cliente para relacionarlo.

Además del registro de citas, se dispondrá de un listado para consultar las citas, filtrando por diversos criterios: fecha (desde, hasta) y cliente. Agrupando por cliente y contrato.

El cliente solo podrá consultar sus propias reservas, y el administrador las de todos.

# Realización de citas

El médico indicará en la cita el número real de reconocimientos médicos realizados, y el cliente y contrato vigente asociado a la revisión realizada.

# Autenticación y roles

Validación con usuario (email) y contraseña.

Roles:

* Cliente
* Médico (solo editar citas de día actual),
* Administración (Podrá acceder a cualquier sección de la app).

# Carga de datos de ejemplos

Se deberá alimentar automáticamente el sistema con datos de ejemplo:

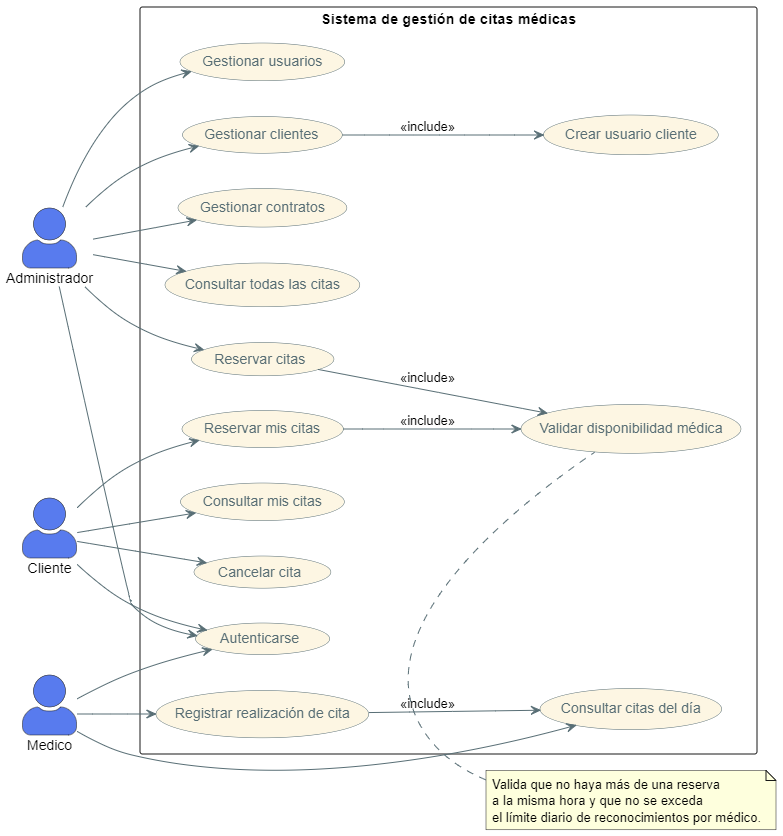
* 1000 clientes
* 30000 citas en varios años distribuidas de forma aleatoria en número para los clientes

# Infraestructura

Dockerizar la app back, contenedor php, nginx, mysql

El angular se compila y se usa con un docker-composer con nginx usando el build.

# Diagrama de casos de uso.

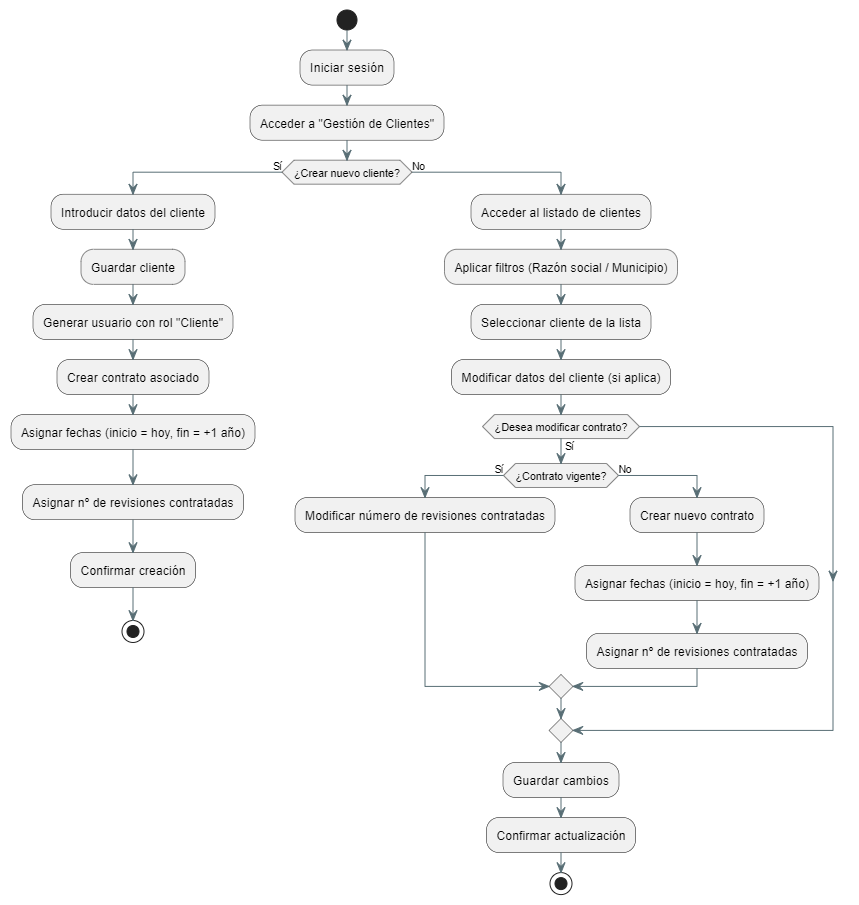
Se plantea los posibles usos que realizarán los usuarios de dentro de la aplicación.

La actividad común para todos los usuarios será la Autentificación, dependiendo de esta, cada usuario dispondrá de funcionalidades diferentes:

## Administradores

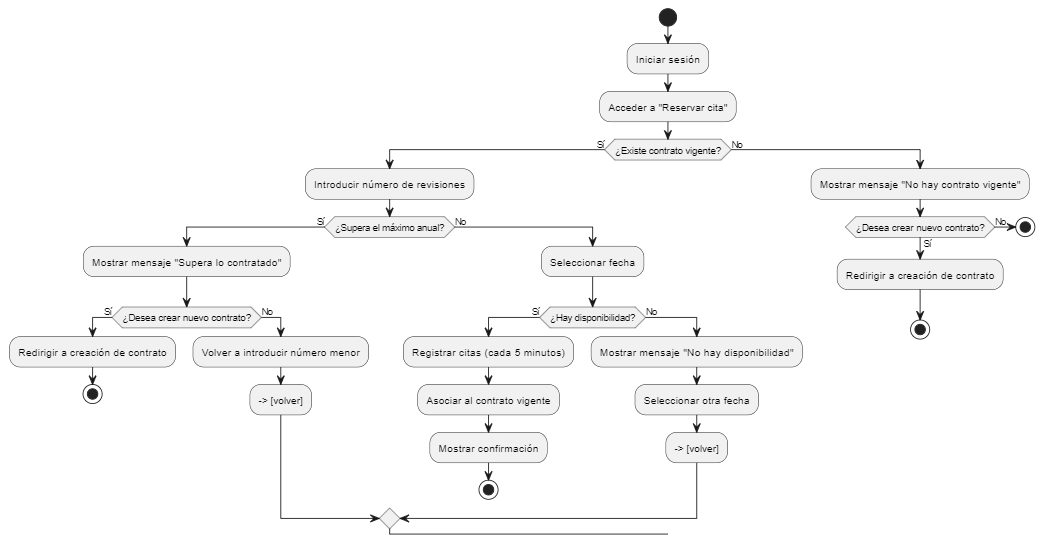
El administrador podrá:

* Gestionar los clientes (Crear y editar).
* Gestionar contratos (Al crear un cliente, se crea un nuevo contrato)
* Gestionar las citas
* Gestionar usuarios



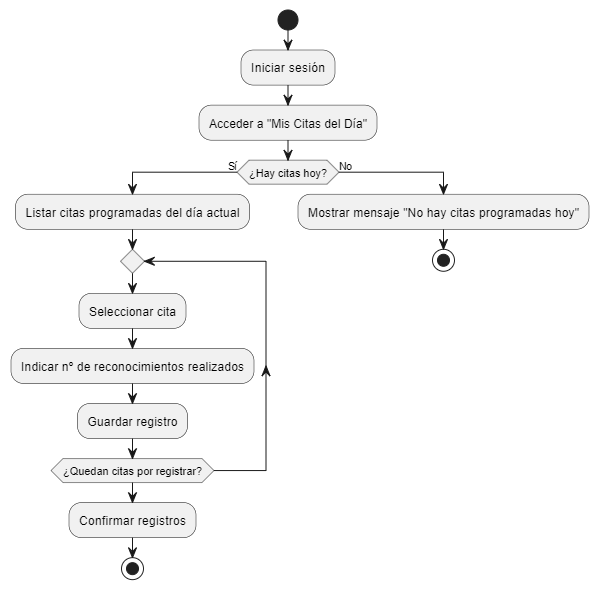
## Clientes

Los clientes podrán consultar y gestionar sus reconocimientos.

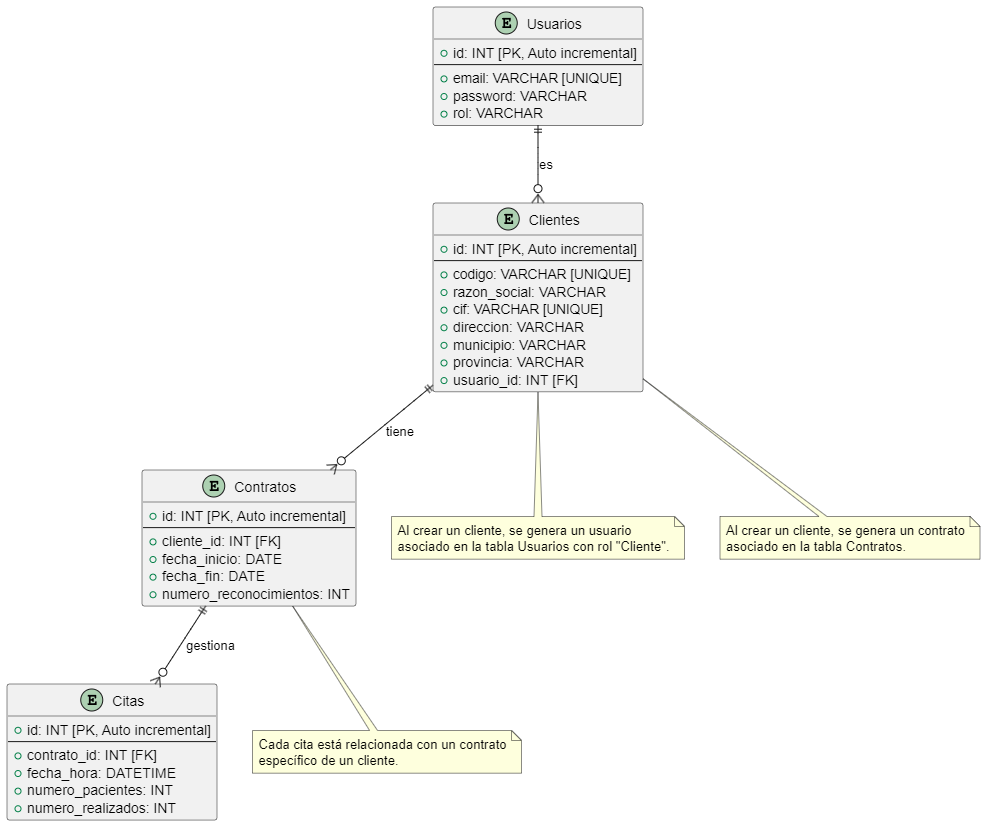


## Médico

Registra la realización de la cita y consulta las citas del día.



# Diagrama con la estructura de la base de datos

según los requerimientos, Médico, no genera tabla, ya que no necesitamos almacenar información particular sobre los médicos, más a allá de su identificación para iniciar sesión que es compartida con Usuarios. Lo mismo ocurre con Administrador.

Se observa la relación “tiene” entre Usuarios y Clientes. Es una relación 1:1 ya que algunos usuarios están relacionados con pacientes (0:1) y todos los pacientes serán usuarios (1:1).

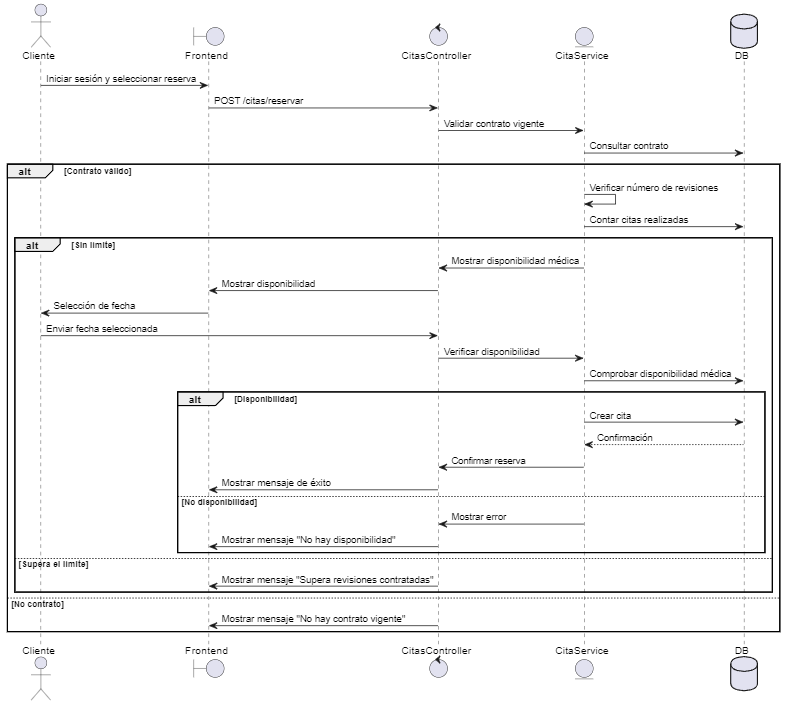
La relación entre Clientes y Contratos, “tiene” es 1:N ya que pueden existir más de un contrato asociado a un cliente a lo largo del tiempo y cada cliente estará asociado a un único contrato vigente.

La relación “tiene” entre Contratos y Citas, relaciona los contratos vigentes con las citas que puede solicitar (1:n) y cada una de estas citas, está relacionada con un único contrato (1:1).

# Flujo de acciones, por parte del código

Podemos ver como interactúan los componentes en el registro de una cita por parte de un administrador o un cliente.

## Cliente.

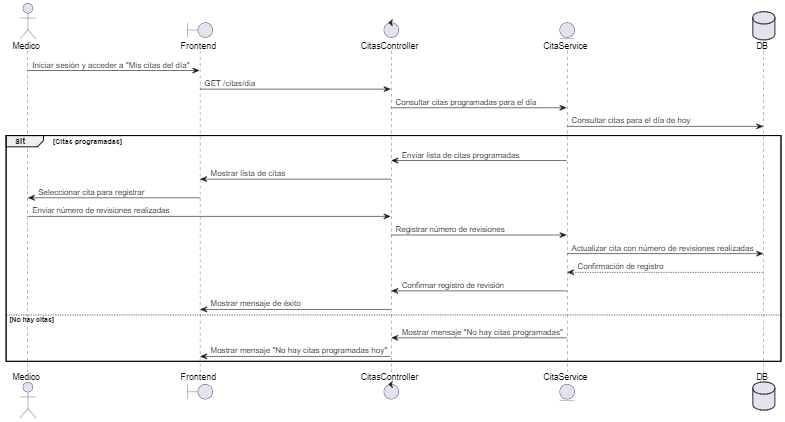


## Administrador



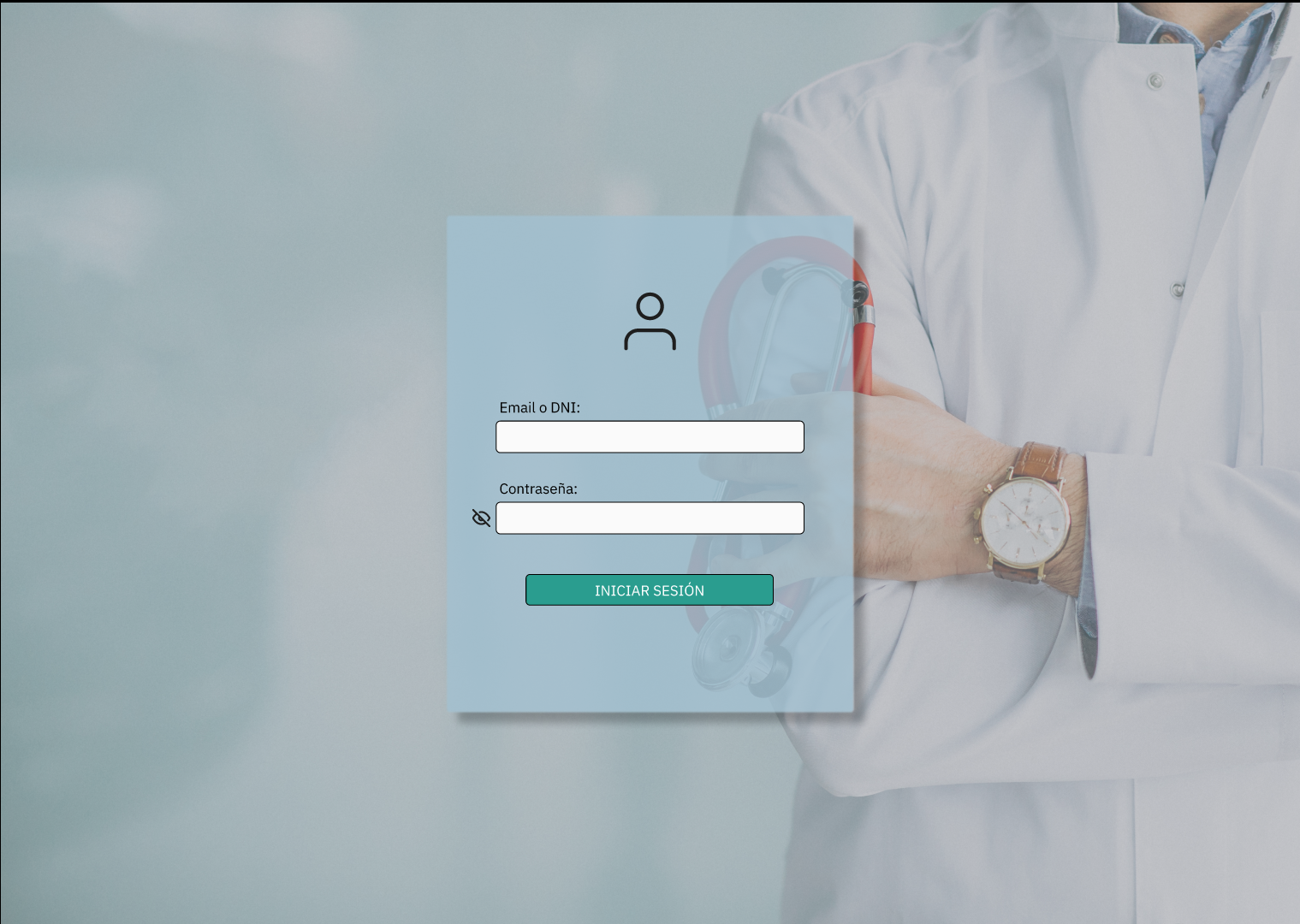
## Médico

En el caso del médico, la vista es mas sencilla, ya que solo hace una consulta.



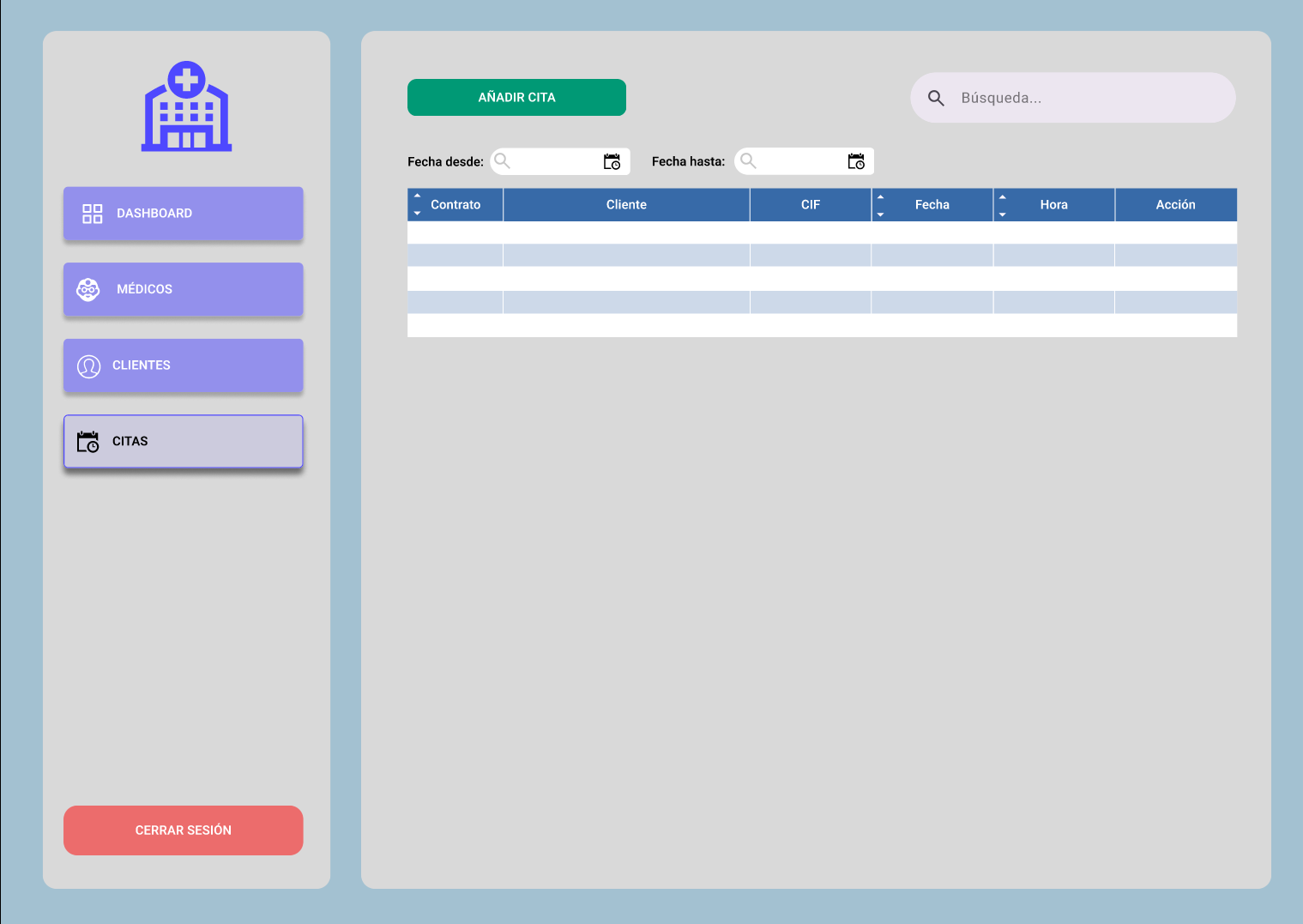
# Bocetos

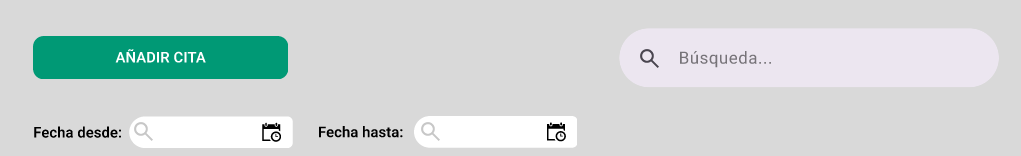
## Login:

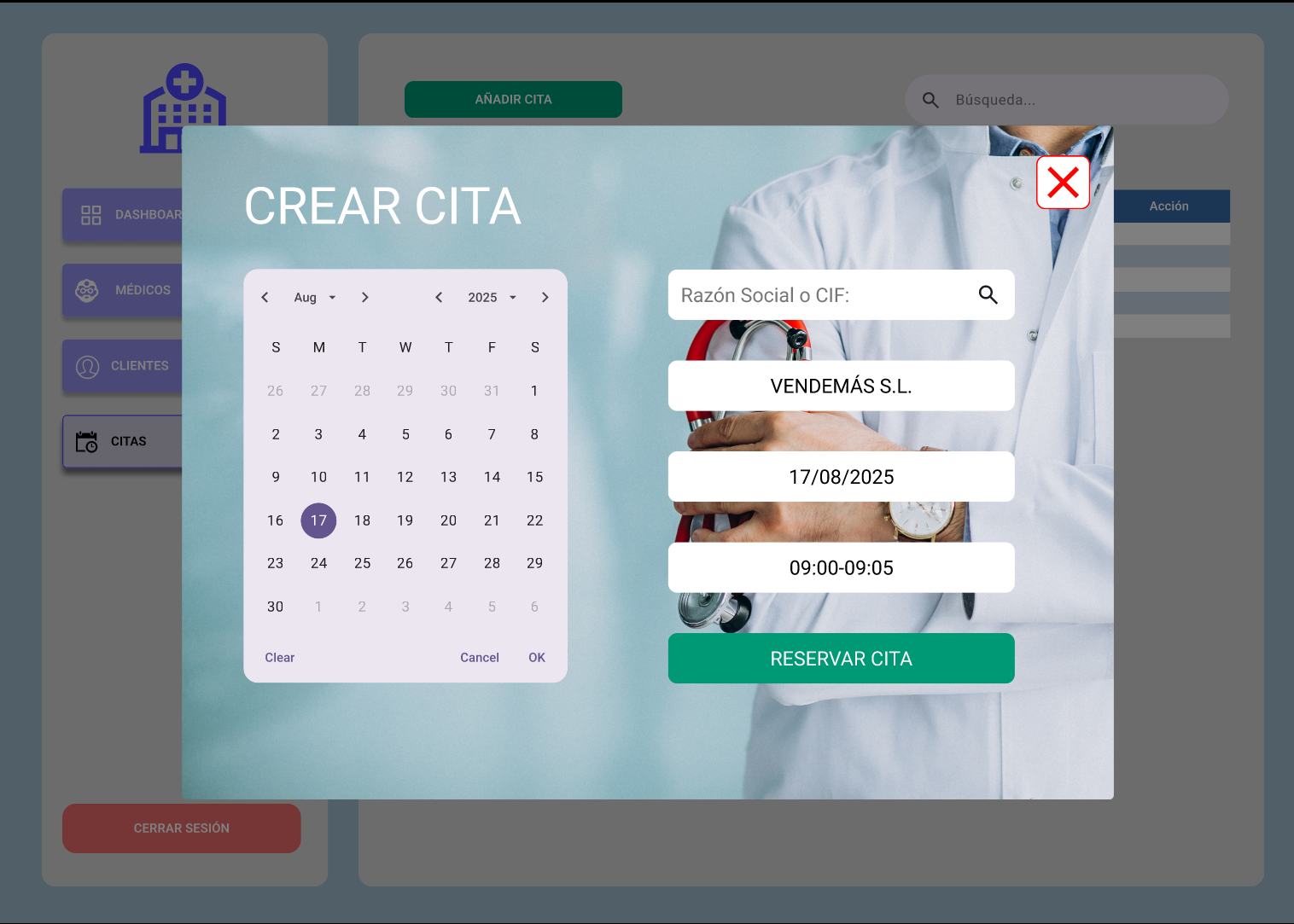


## Vistas Administrador:

* Citas:



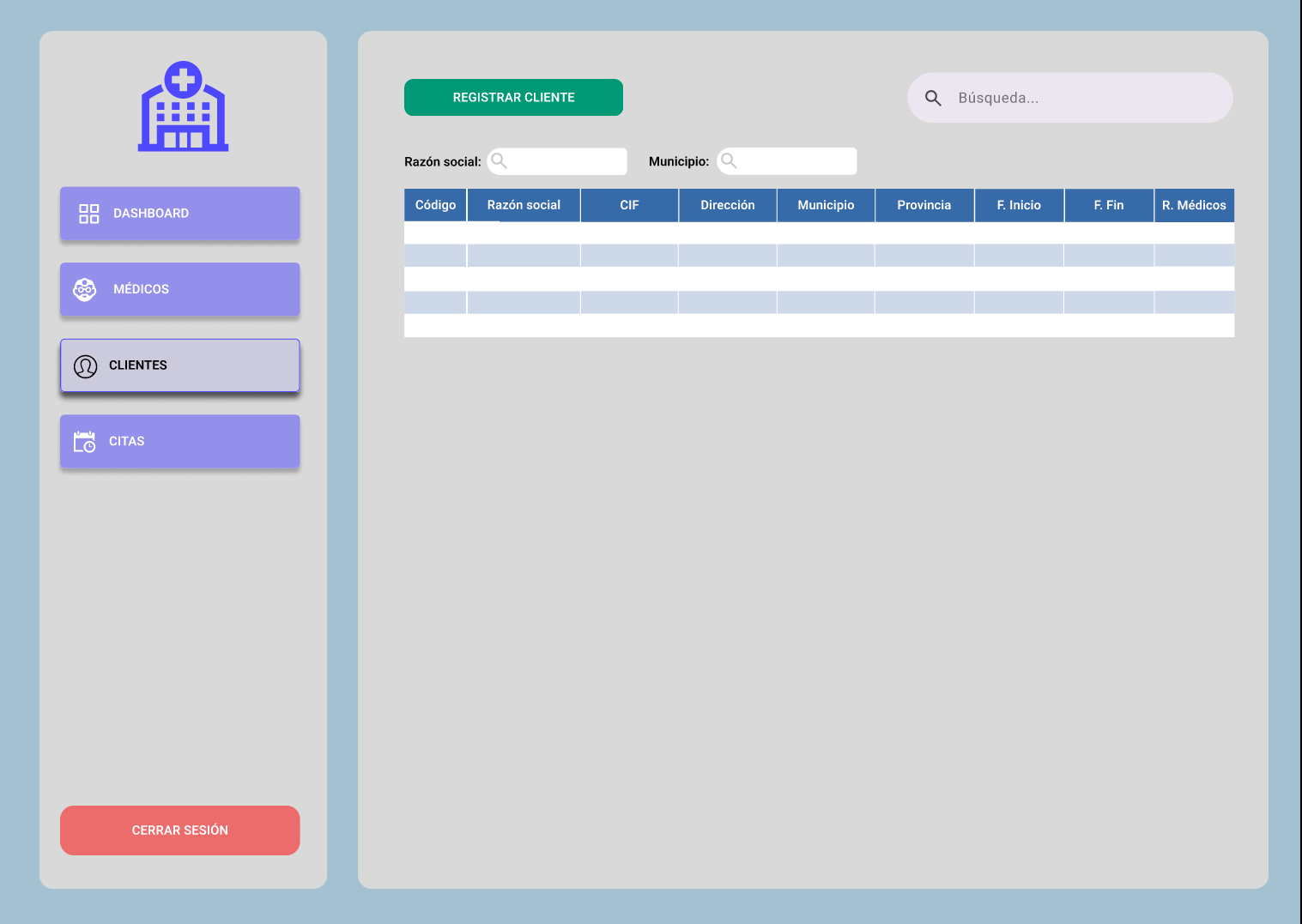


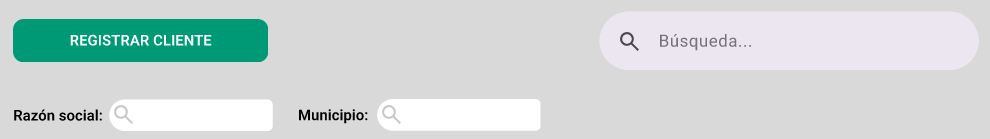


Modal para crear una cita

Unos de los requisitos en Citas era buscar por fecha desde y fecha hasta. También se puede filtrar por nombre, por CIF o contrato.

* Clientes:



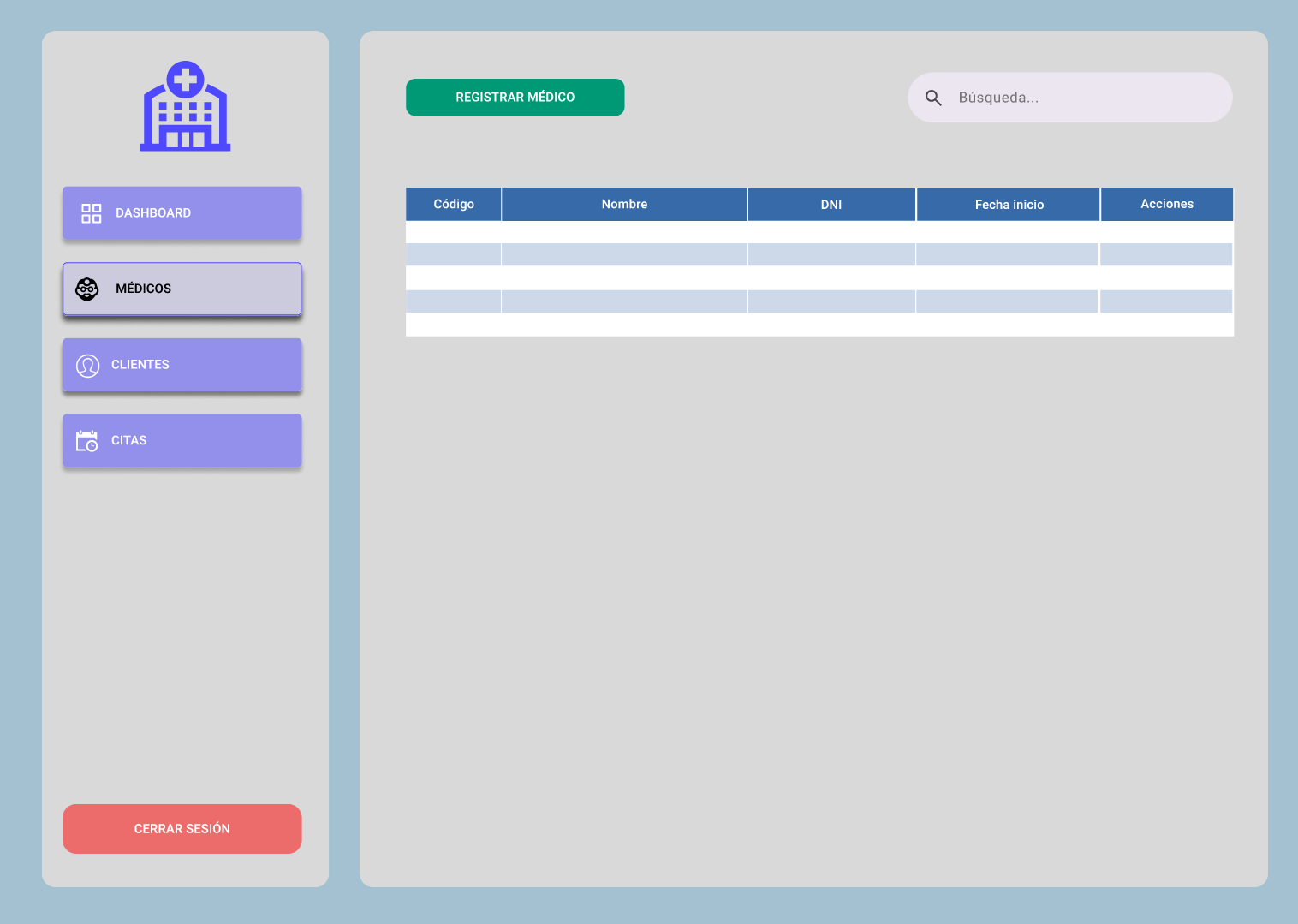


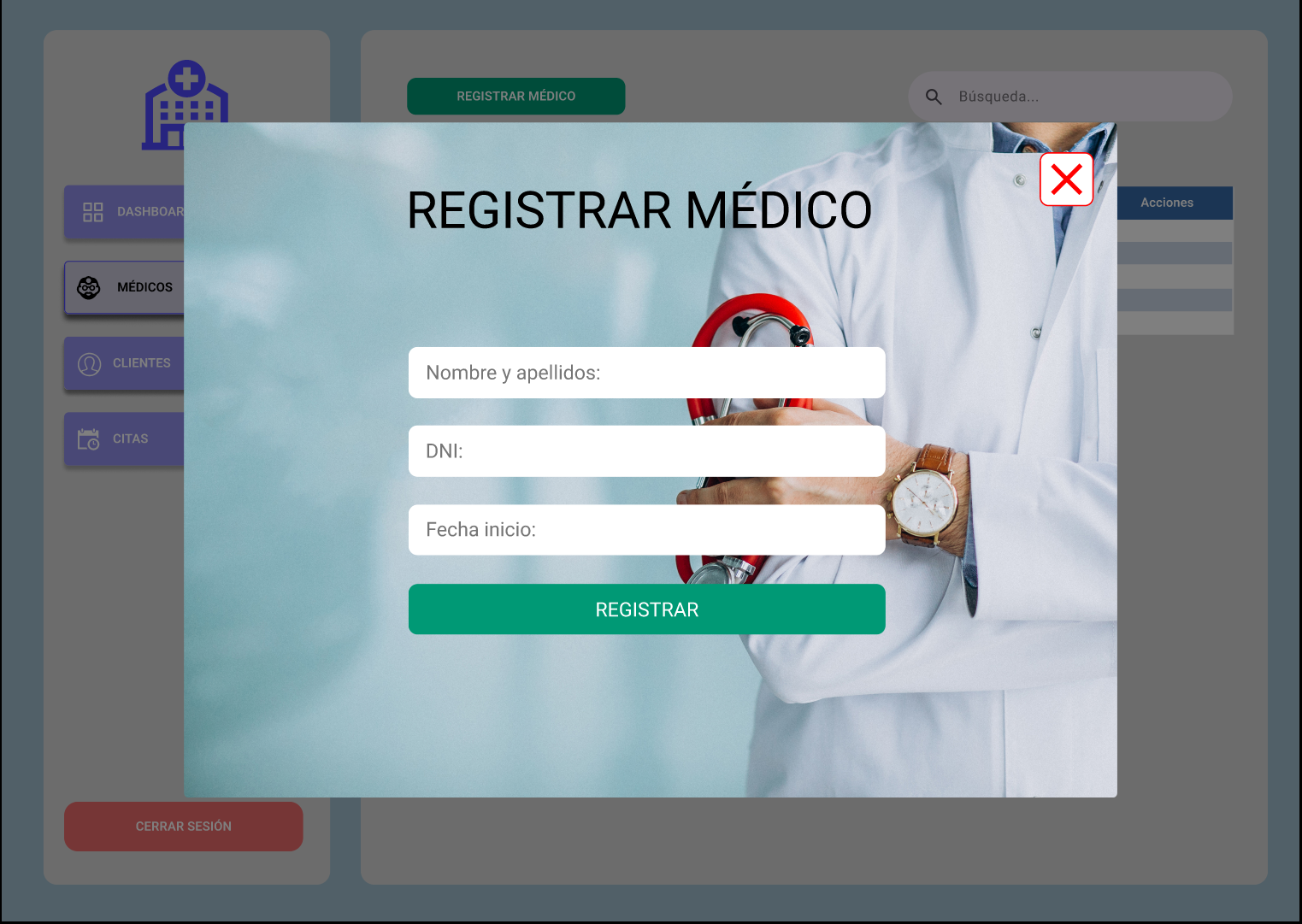
El requisito en clientes era la búsqueda por razón social o municipio.



Modal para registrar a un cliente.

* Médicos:





Modal con el formulario para registrar un médico

## Vista Cliente:



## Vista médico:

