

**CLINICAL ADMIN**

2019

Carlos Eduardo Arreola Cuin

Monarca Labs

16-5-2019

Manual Técnico

# **Introducción**

Este manual contiene las especificaciones técnicas más importantes del sistema desarrollado. Constituye una guía especializada para la realización de las operaciones de mantenimiento de la aplicación. Se encuentra dirigido fundamentalmente a la dirección de Tecnologías de la Información, al administrador del sistema, a otros desarrolladores, así como al departamento de calidad y auditoría de sistemas.



**CLINICAL ADMIN**

**Contenido**

[**Introducción** 2](#_Toc9014655)

[**Objetivos Generales y Específicos del Sistema** 3](#_Toc9014656)

[**Normas, Políticas y Procedimientos** 3](#_Toc9014657)

[**Fundamentación de la Tecnología Utilizada** 3](#_Toc9014658)

[**Descripción de los Actores del Sistema** 5](#_Toc9014659)

[**Modelo Lógico de Datos** 6](#_Toc9014660)

[**Diseño de Pantallas** 7](#_Toc9014661)

[**Vista de Implementación** 13](#_Toc9014662)

[**Diagrama de Navegación del Sistema** 14](#_Toc9014663)

[**Glosario de Términos** 14](#_Toc9014664)

# **Objetivos Generales y Específicos del Sistema**

* El objetivo general del es una aplicación web para llevar un control de la gestión de citas y de expedientes médicos de pacientes para acudan a una cita médica.
* Este sistema apoyará de gran medida a los usuarios finales (médicos y secretarias) ya que será sencillo, intuitivo y estandarizado para que cualquier médico de las diferentes ramas de la salud puedan hacer uso de éste, así como las secretarias puedan llevar una agenda más formal.
* Los pacientes podrán solicitar citas médicas mediante la plataforma web.

# **Normas, Políticas y Procedimientos**

* A los pacientes se les crea una cuenta directamente en el hospital o clínica al que acuden.
* Una vez que la cuenta del paciente es creada se le asigna un expediente.
* Los encargados de revisar las solicitudes de los pacientes son los encargados de decidir si esta es aprobada o rechazada.
* Si la solicitud es aceptada se le asigna una fecha y una hora además de el doctor que los atenderá.
* Cada cita realizada se agrega al historial clínico del paciente.
* El expediente de cada paciente contiene su información personal y todo su historial clínico

# **Fundamentación de la Tecnología Utilizada**

Tecnologías a utilizar:

**Front end**

En la parte del front end(cliente) se decidió utilizar las siguientes tecnologías:

* React
* Apollo Client
* JavaScript
* Sematic UI.

El utilizar Reac fue porque es de Facebook y ahora es código abierto, cada vez más empresas y desarrolladores han adoptado React. Una de las razones es porque la librería tiene soporte completo por parte de Facebook. Actualmente la librería se encuentra en la versión 16.4, esto gracias a que Facebook no ha dejado de invertir en React (y no lo dejará de hacer en mucho tiempo). Esto significa que todos los proyectos que utilizan React tienen soporte de largo plazo por defecto.

El utilizar Apollo Client es porque puede proporcionar un cliente en cualquier lugar de la jerarquía de componentes de React dándonos así una buena combinación de estas dos tecnologías.

Semantic UI es un framework para crear el diseño de interfaces de manera responsive utilizando HTML/CSS legible y son tres las características que definen a Semantic UI: el lenguaje natural, la configurabilidad de Javascript y los componentes predefinidos.

**Back end**

En la parte del **Back end**(servidor) se decidió utilizar las siguientes tecnologías:

* Node js
* Apollo Server
* JSON Web Tokens
* mongoDB.
* GraphQL.

Se decidió utilizar Node js porque puede ejecutarse en una variedad de servidores, entre los que destacan Microsoft Windows, Mac OS X y Unix. Además, porque plataformas como LinkedIn, eBay o PayPal fueron creadas con Node.js, lo que indirectamente es una muestra de su calidad

GraphQL es un protocolo de consulta de datos originalmente elaborado por Facebook para uso interno. Desde 2015 existe en una versión open source que ha impulsado su desarrollo y que ha generado muchas ventajas y beneficios prácticos. GraphQL no es una librería ni un framework, se trata de un lenguaje de consulta que puede ser utilizado indistintamente en diferentes códigos de programación, incluidos algunos de los más usados hoy en día como JavaScript, Ruby, Java, PHP y Python.

Apollo es un framework para la integración con GraphQL, nos facilitarán la construcción de las consultas y mutaciones. El uso del mecanismo de las variables, de cara al consumidor, permitirá mantener estructuras en las cuales los valores serán sustituidos de forma dinámica.

Como utilizaremos Node js además de GraphQL es común usar Mongo debido a que los datos se almacenan en estructuras parecidas a un JSON, esto hace que el flujo de datos dentro de la aplicación no tenga mayores cambios en la estructura de datos. En MongoDB tenemos múltiples operadores que nos permiten crear consultas poderosas con poco código, además que contamos con las Agregaciones que son un mecanismo que nos permite realizar operaciones entre múltiples colecciones.

JWT es una solución moderna para un problema antiguo: ¿Cómo sé quién es ese usuario? Nos ayudan al tener firma, ser stateless (protocolo sin estado) y por tener un formato común de datos. Es un sistema de transferencia segura de datos, que puede enviarse por URL, POST o en un encabezado HTTP de manera segura. El usuario firma la información digitalmente a través de un algoritmo criptográfico. Una vez que el usuario está conectado, se incluirá el JT para cada pedido realizado, permitiendo el acceso en los servicios y recursos del sistema.

Los JSON Web Token funcionan en diferentes lenguajes de programación: .NET, Python, Node.js, Java, PHP, Ruby, Go, JavaScript y Haskell. Lo que permite su uso en diferentes escenarios.

# **Descripción de los Actores del Sistema**

Tipos de usuarios del sistema:

**Administradores**

Los administradores son los aquellos que pueden realizar cualquier tipo de acción en el sistema y pueden ver todo lo que hay en él.

**Trabajadores**

Existen tres tipos de trabajadores en nuestro sistema los cuales son:

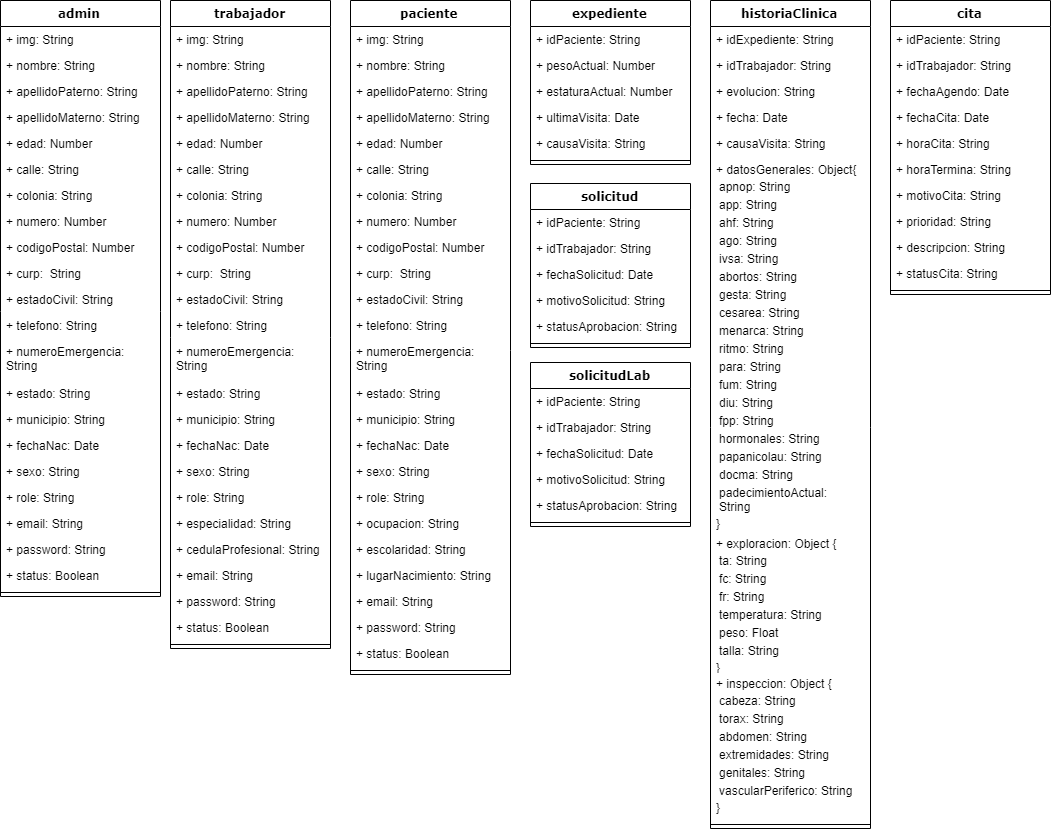
* Doctores
* Enfermer@s
* Secretari@s

Cada uno de los diferentes roles puede acceder a diferentes partes del sistema.

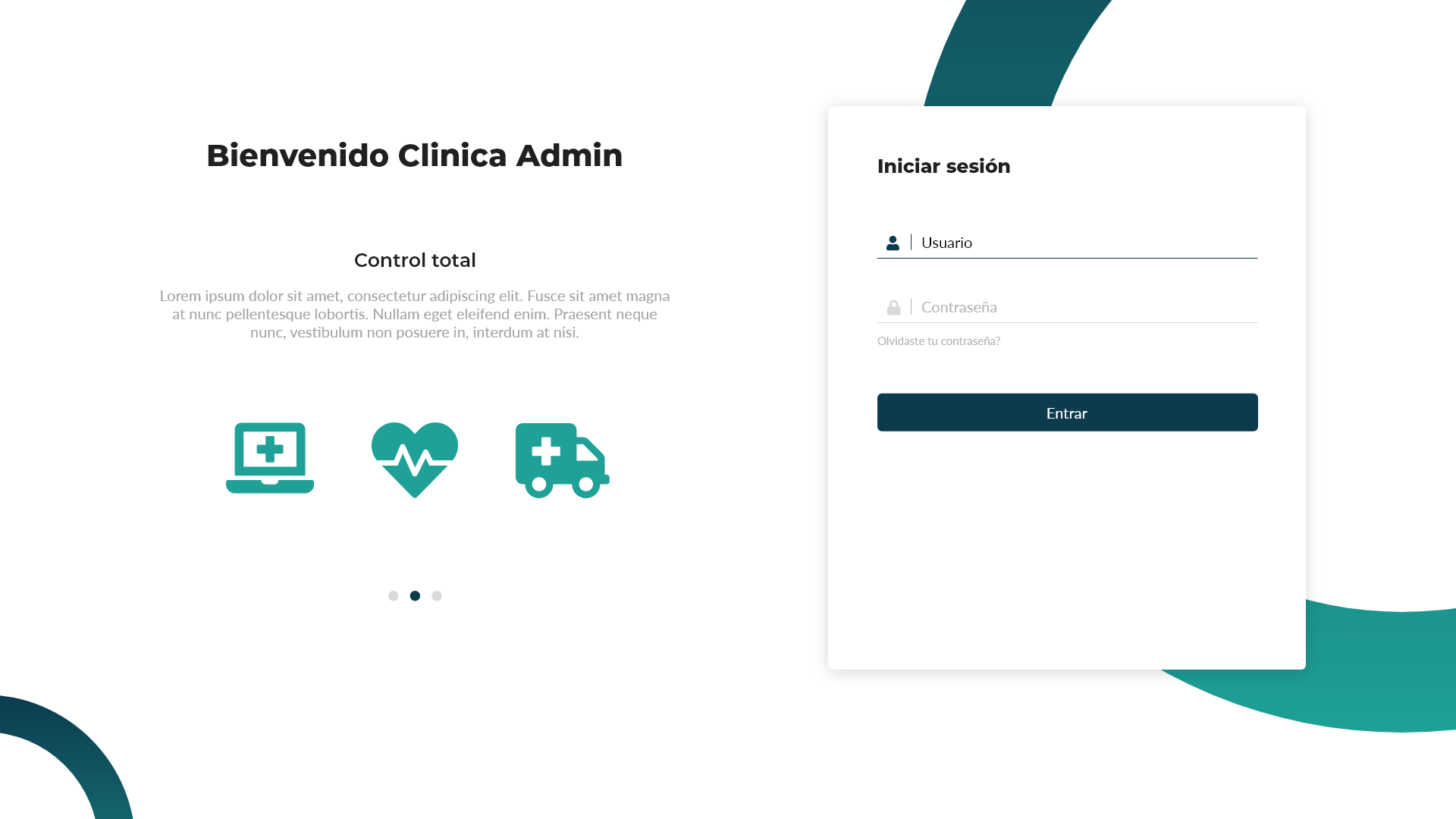
**Paciente**

Los pacientes son los usuarios que pueden solicitar citas médicas a la clínica y nada más.

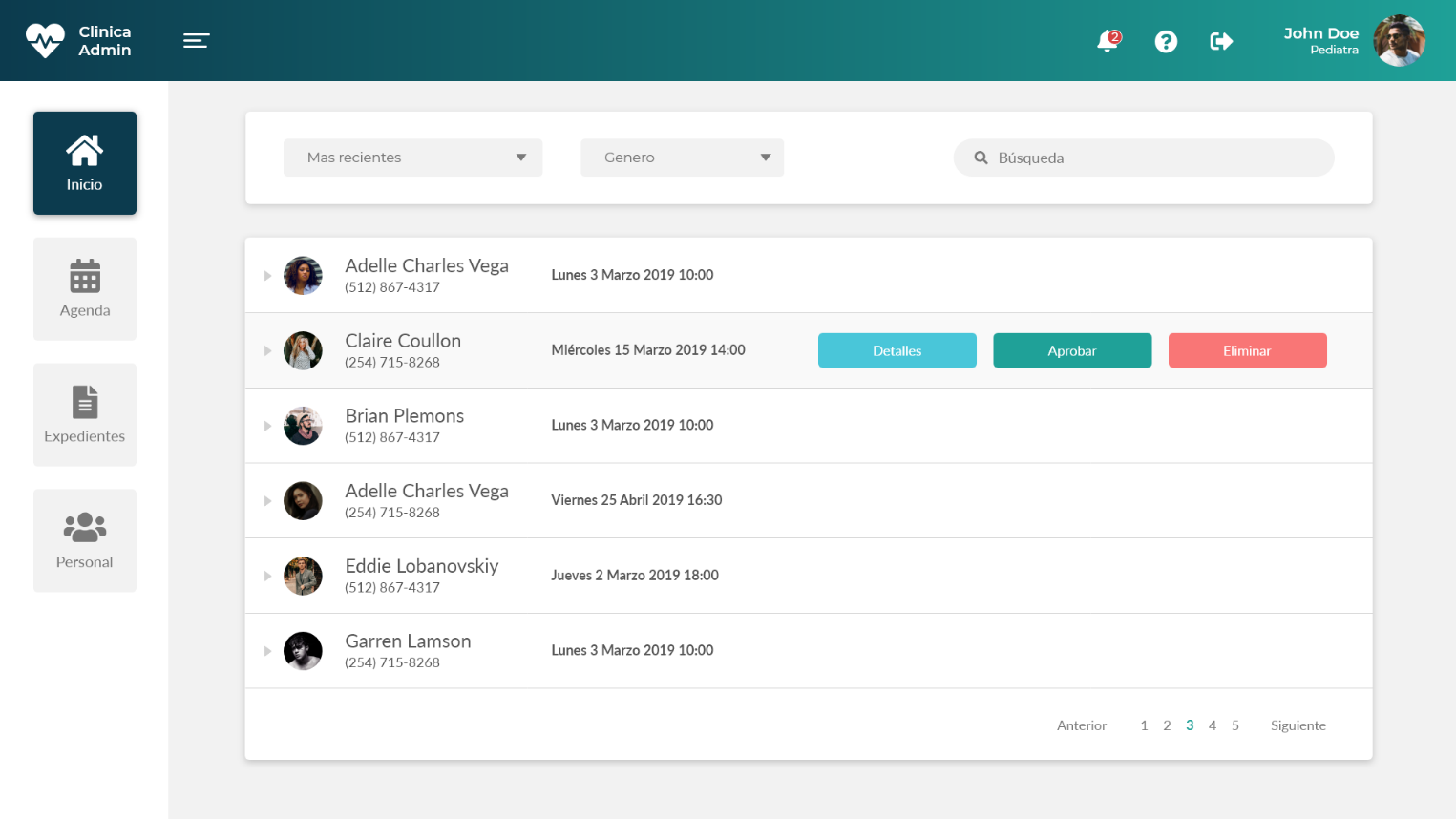
# **Modelo Lógico de Datos**

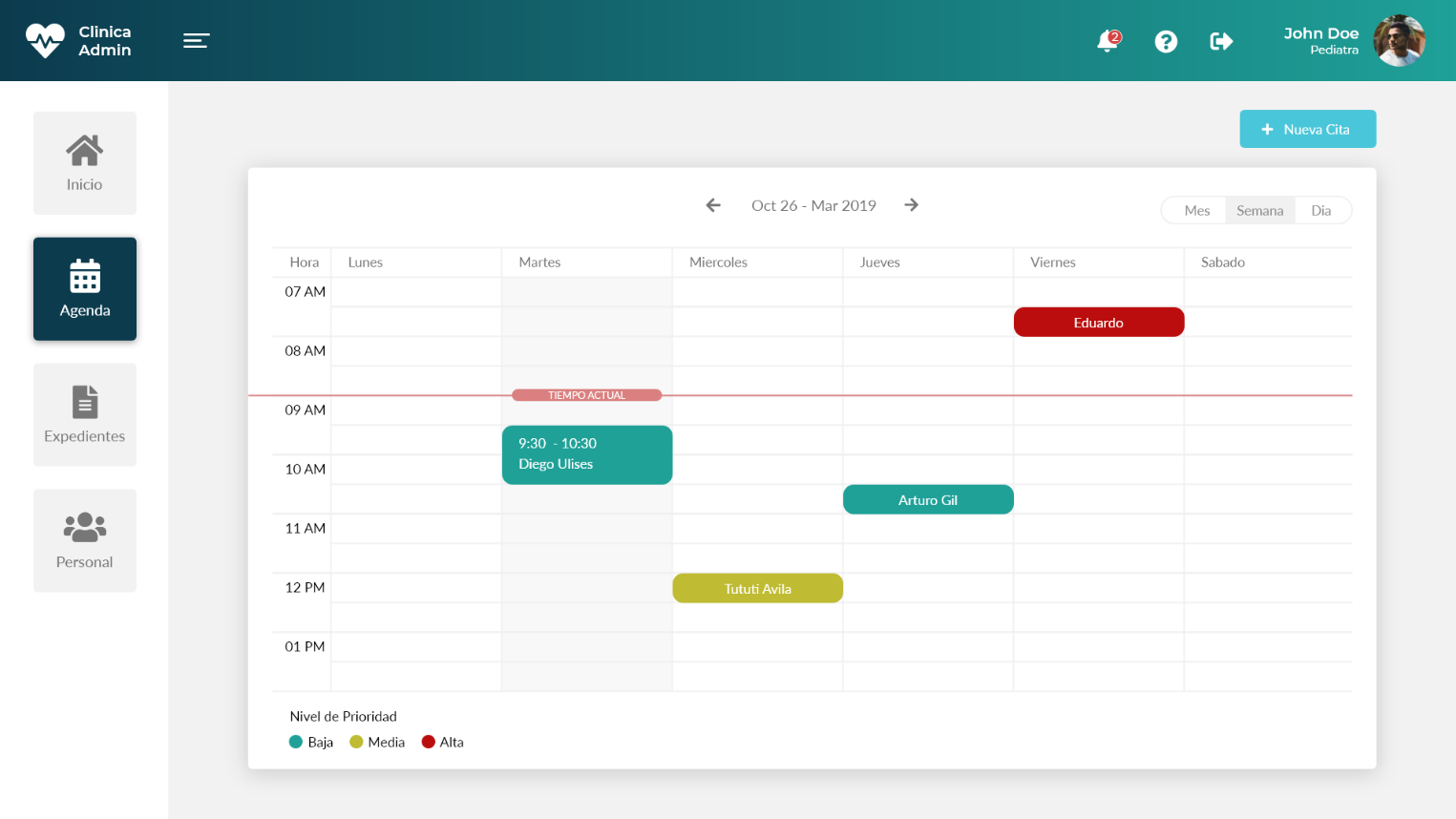


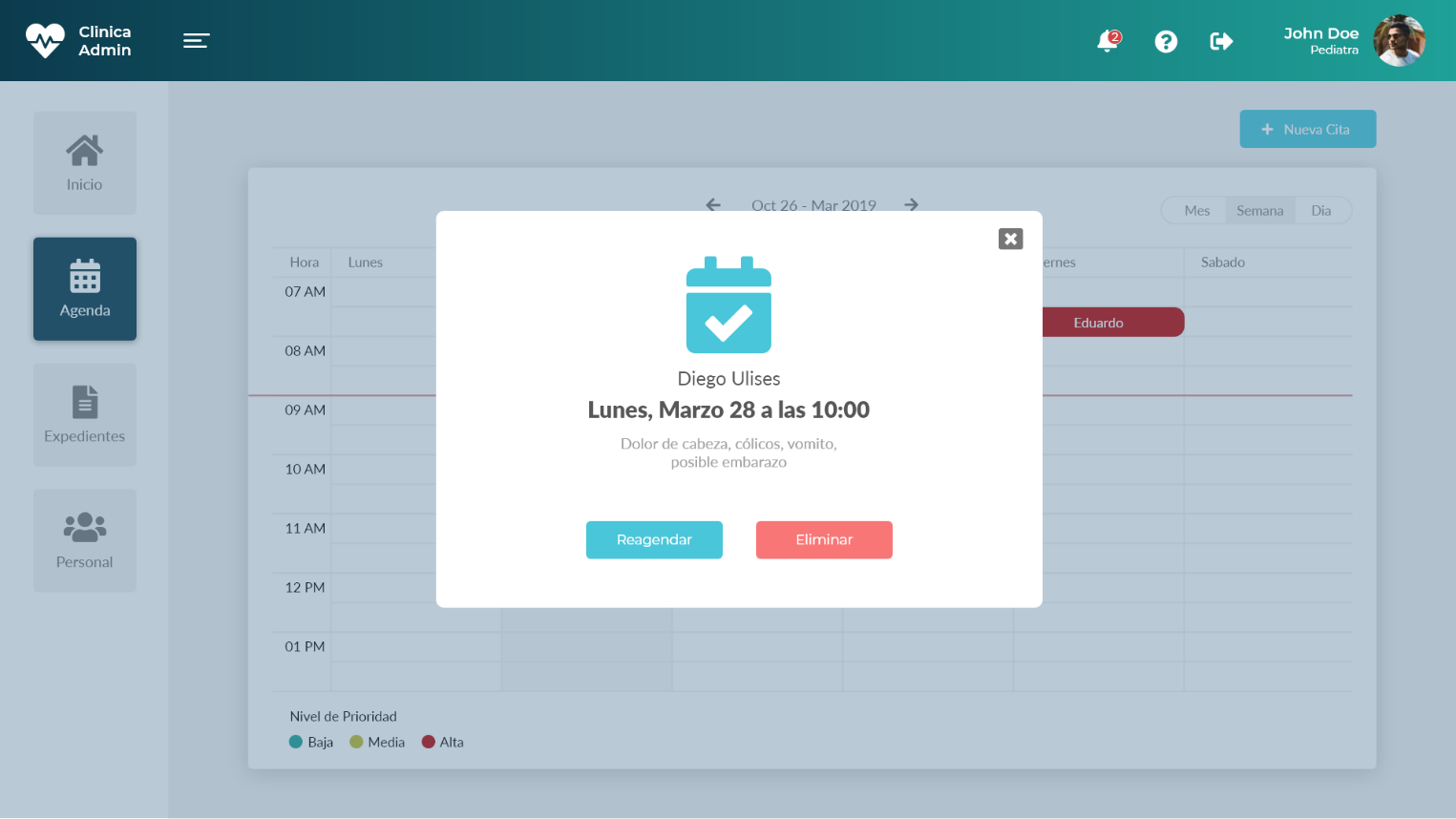
# **Diseño de Pantallas**

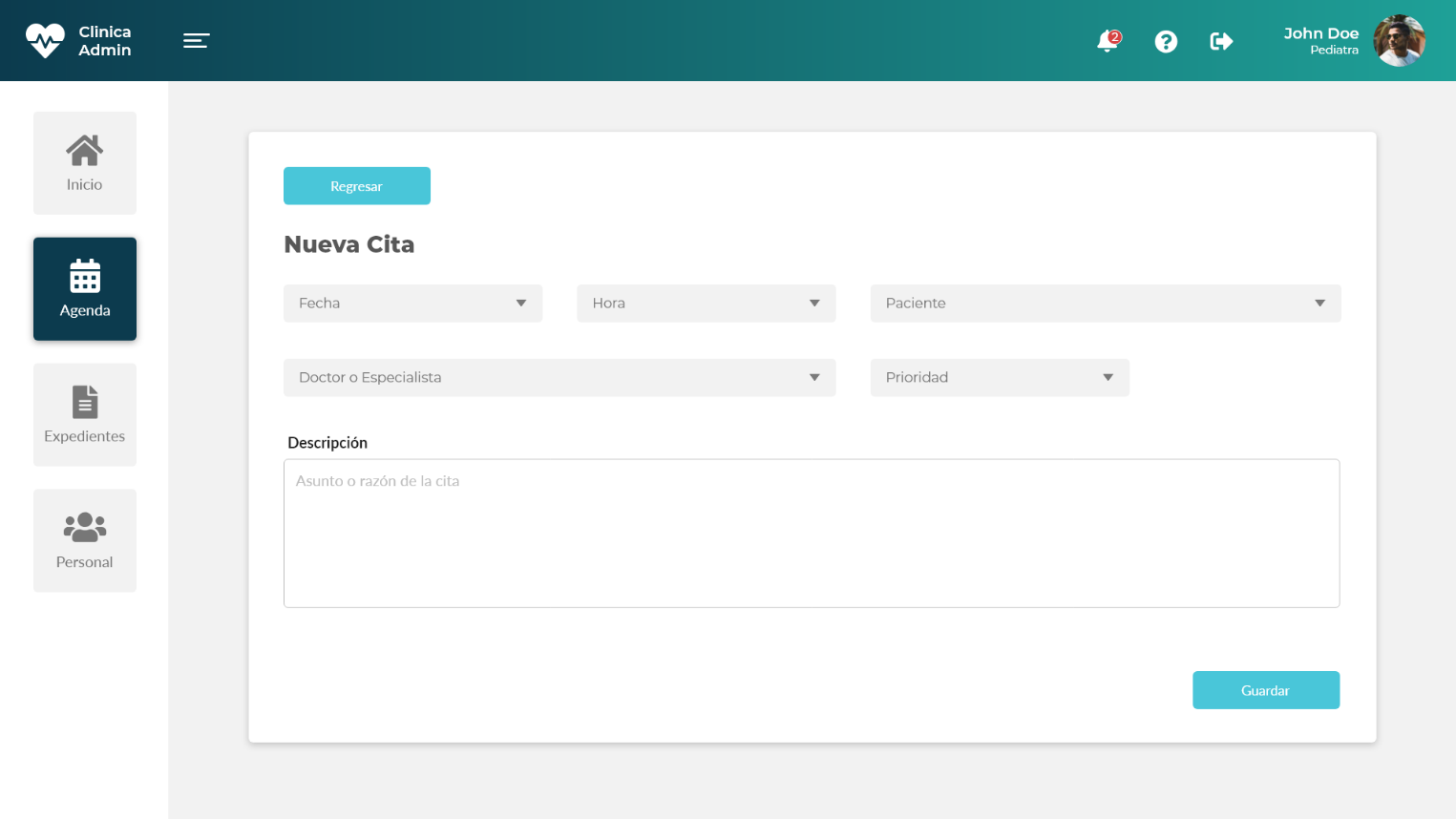


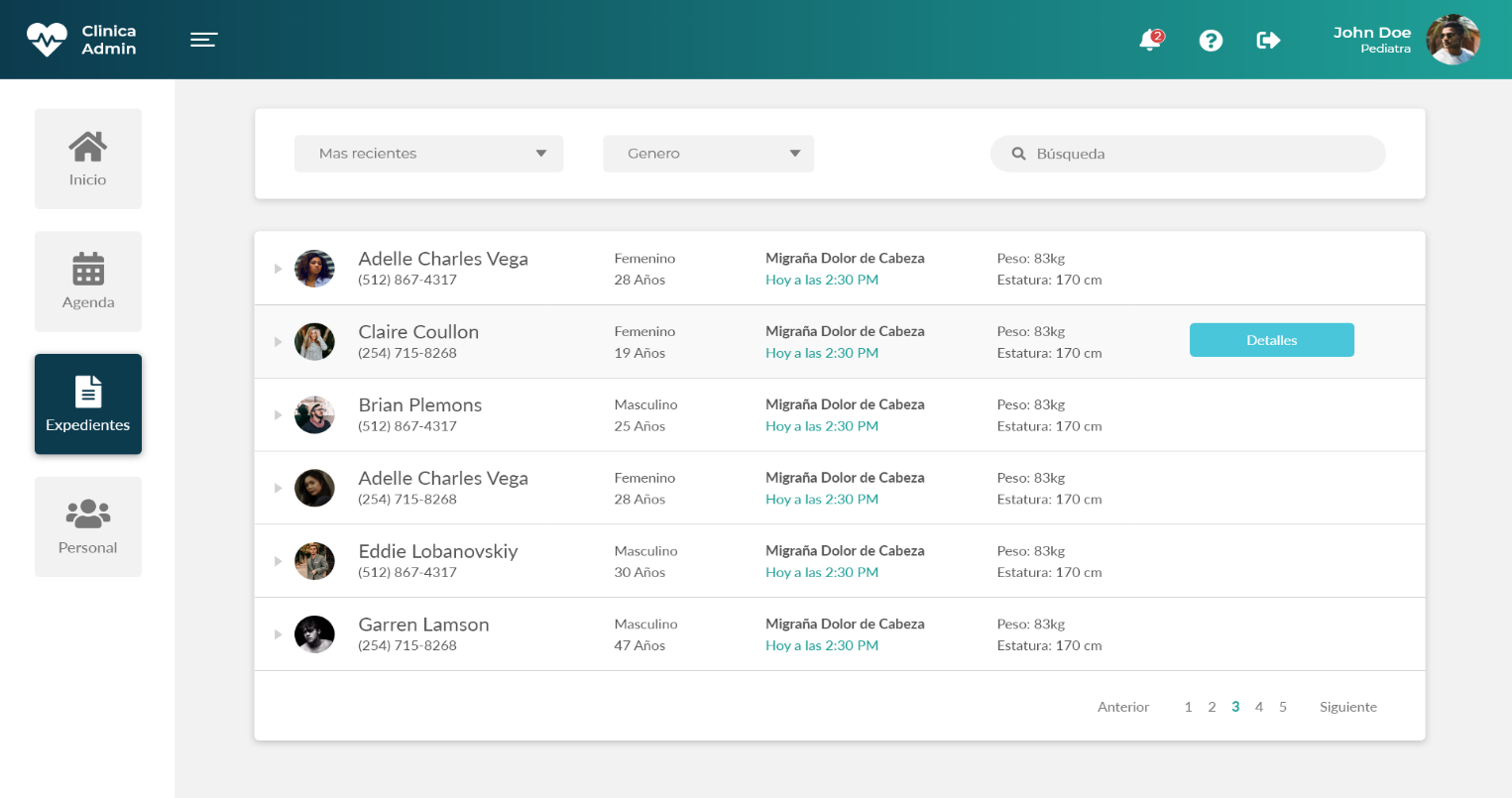
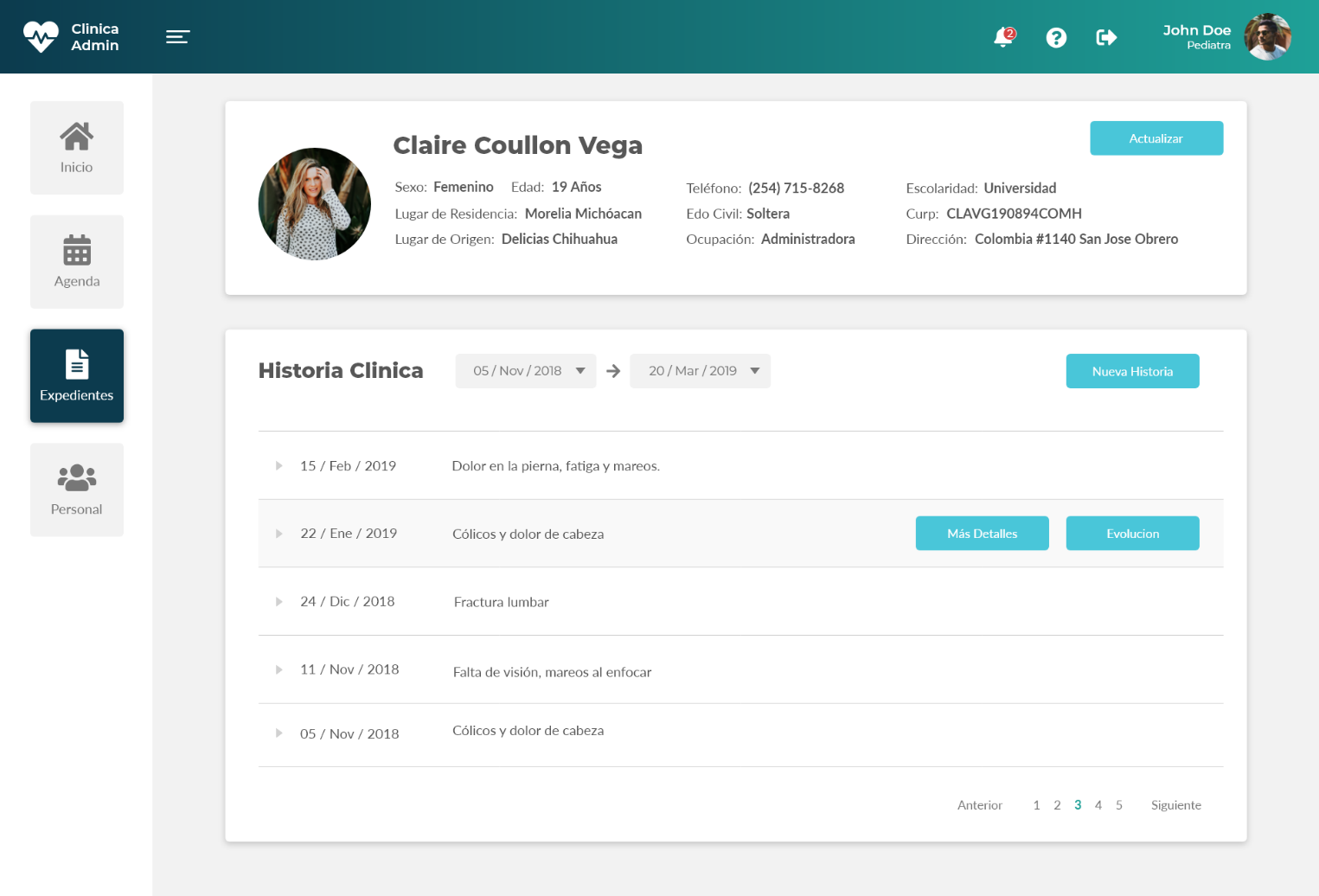


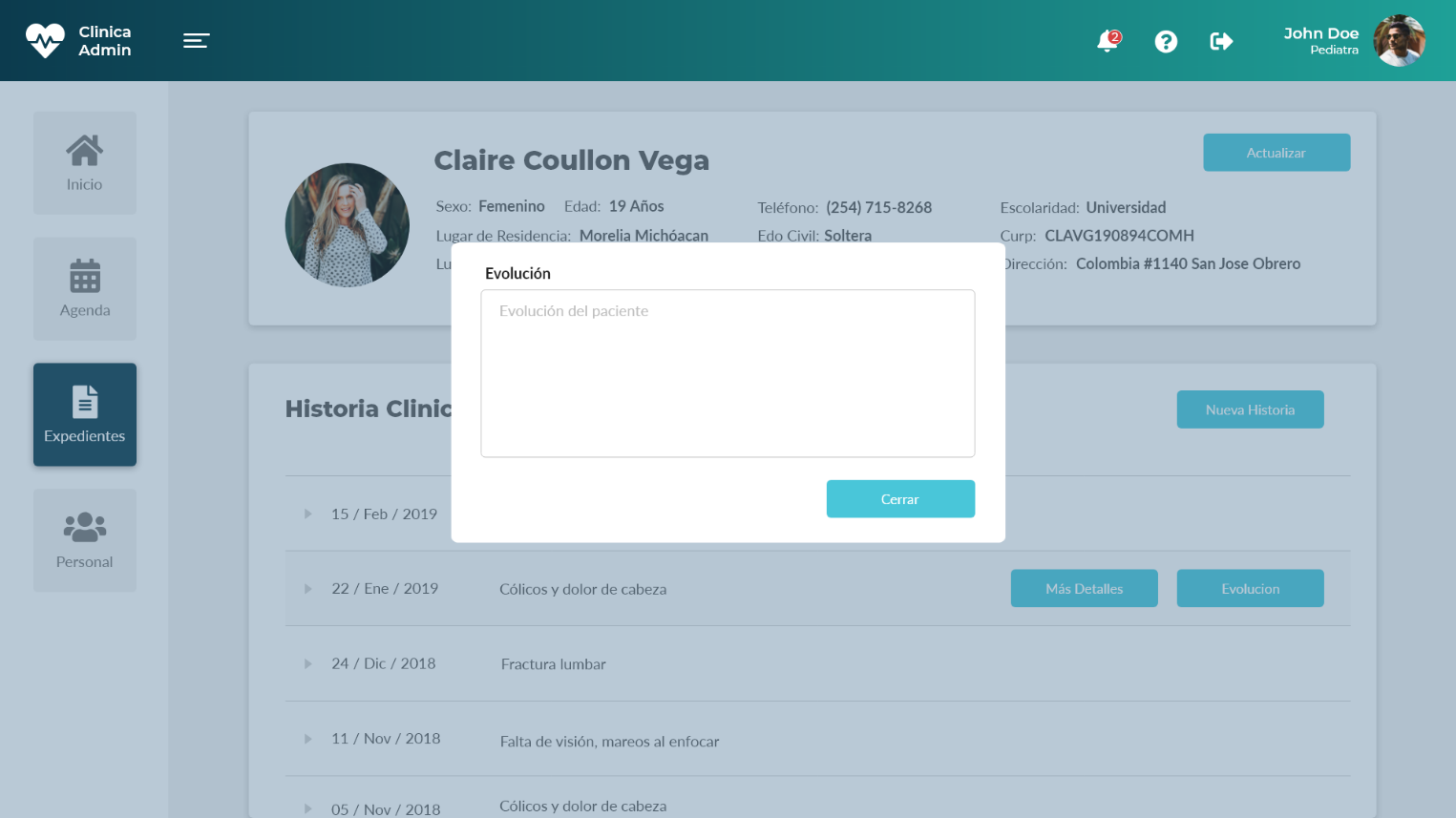




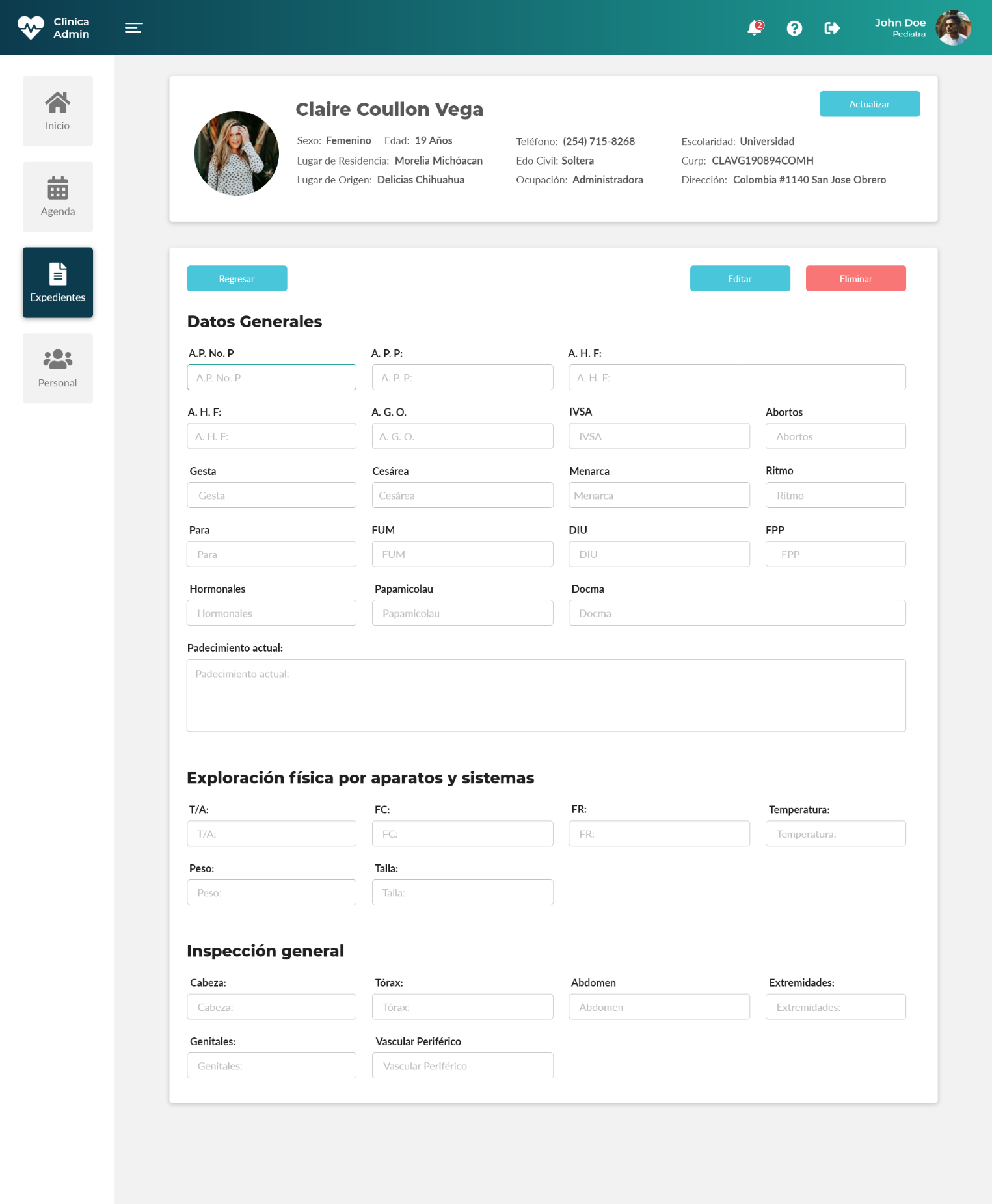


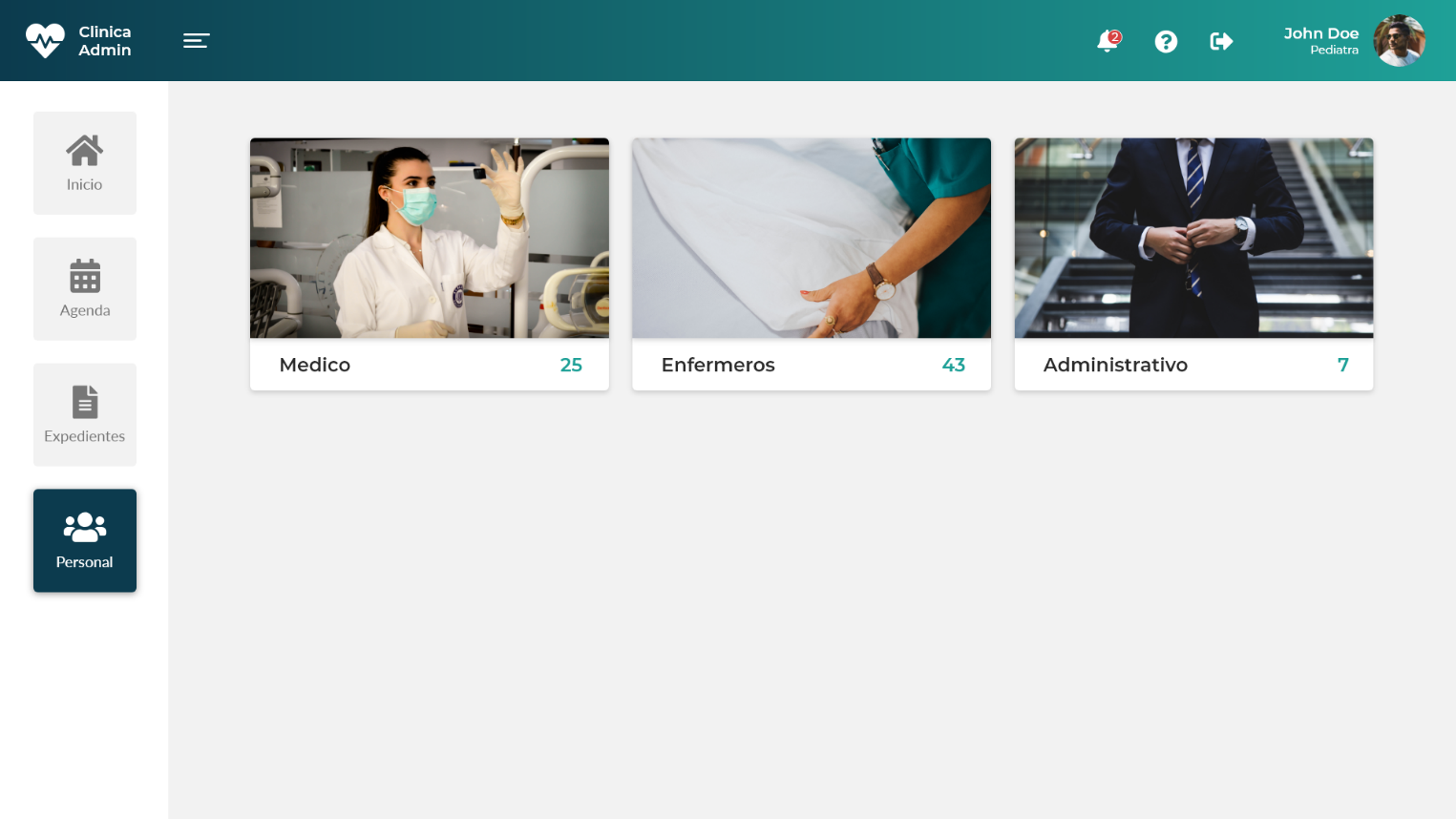








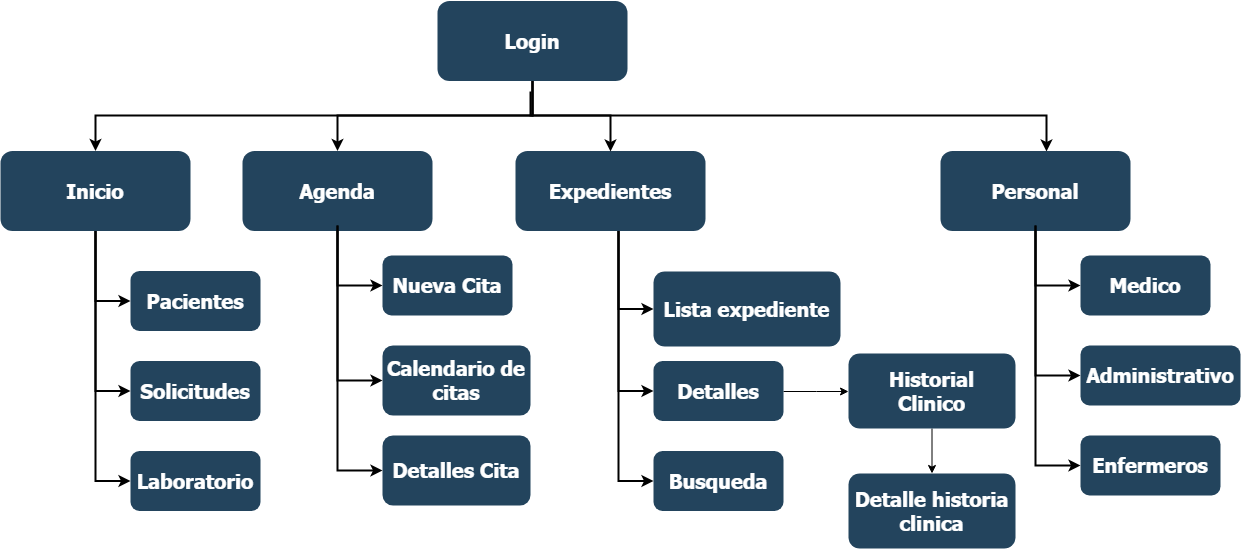




# **Vista de Implementación**



# **Diagrama de Navegación del Sistema**



# **Glosario de Términos**

**Front-end**: en diseño de software el front-end es la parte del software que interactúa con los usuarios y el back-end es la parte que procesa la entrada desde el front-end.

**React**: es una biblioteca escrita en JavaScript, desarrollada en Facebook para facilitar la creación de componentes interactivos, reutilizables, para interfaces de usuario. Se utiliza en Facebook para la producción de componentes, e Instagram está escrito enteramente en React

**JavaScript**: JavaScript (abreviado comúnmente JS) es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, ​ basado en prototipos, imperativo, débilmente tipado y dinámico. ... Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web.

**Sematic UI**: es un framework para crear el diseño de interfaces de manera responsive utilizando HTML/CSS legible. ... Viene integrado con otros frameworks o librerías como son Angular, React, Ember o Meteor. Utiliza cinco categorías descriptivas para definir los componentes: UI Element: un bloque de construcción básico.

**Back-end**: es la parte del desarrollo web que se encarga de que toda la lógica de una página web funcione. Se trata del conjunto de acciones que pasan en una web pero que no vemos como, por ejemplo, la comunicación con el servidor.

**Node js**: es una forma de ejecutar JavaScript en el servidor, además de mucho más. Node.js es un entorno Javascript del lado del servidor, basado en eventos. Node ejecuta javascript utilizando el motor V8, desarrollado por Google para uso de su navegador Chrome

**JSON Web Tokens**: JWT es un standard RFC 7519 que define un método compacto, autónomo y seguro para transmitir información entre 2 sistemas. Se usa un objeto JSON de forma segura y mucho mas ligera que XML.

**MongoDB**: es una base de datos orientada a documentos. Esto quiere decir que en lugar de guardar los datos en registros, guarda los datos en documentos. Estos documentos son almacenados en BSON, que es una representación binaria de JSON.

**GraphQL**: es un protocolo de consulta de datos originalmente elaborado por Facebook para uso interno. Desde 2015 existe en una versión open source que ha impulsado su desarrollo y que ha generado muchas ventajas y beneficios prácticos

**Aplicación web**: Las aplicaciones web son un tipo de software que se codifica en un lenguaje soportado por los navegadores web y cuya ejecución es llevada a cabo por el navegador en Internet o de una intranet (de ahí que reciban el nombre de App web).