Especificación de requisitos de software

Para

Aplicación de gestión hospitalaria

Versión 2.0 aprobado

Preparado por Diederik Montaño, Juan Pérez y Jesús Guzmán

Universidad Tecnológica de Bolívar

11 de octubre del 2024

Tabla de contenido

Historial de revisiones

Nombre	Fecha	Motivo de los cambios	Versión
Diederik Montaño, Jesús Guzmán	26/08/2024	Creación de la primera versión del SRS.	V0.1
Diederik Montaño, Juan Pérez	12/09/2024	Modificación de los requerimientos no funcionales por cambios en infraestructura de la aplicación.	V0.2
Diederik Montaño	23/09/2024	Incluir requerimientos de usuario y sistema.	V1.5
Diederik Montaño, Juan Pérez	02/10/2024	Actualizar requerimientos no funcionales por cambios en el despliegue de la aplicación.	V1.6
Diederik Montaño	09/10/2024	Anexar diagramas de proceso de negocio y su explicación.	V1.7
Juan Pérez, Jesús Guzmán	10/10/2024	Anexar diagrama de clases y casos de uso.	V1.8
Diederik Montaño, Juan Pérez	11/10/2024	Describir casos de uso.	V2.0

1. Introducción

La aplicación de gestión hospitalaria es un software de gestión de datos enfocado en facilitar el proceso de almacenamiento y accesibilidad de estos, que responde a las necesidades de actualización informáticas de un hospital en constante crecimiento. En el desarrollo se priorizan tanto la accesibilidad como la privacidad de los usuarios, escalabilidad de la aplicación y disponibilidad de esta.

Este documento busca facilitar la comprensión de los objetivos, segmentos del sistema y las tareas de desarrollo desde el nivel más básico hasta el más complejo, simplificando a los desarrolladores el cumplimiento de las necesidades del cliente y las consideraciones implícitas en el contexto. Su objetivo es únicamente establecer los requisitos del software, dividido en requerimientos funcionales, no funcionales, de usuario y de sistema; sin embargo, contiene otra información relevante para el desarrollo tal como diagramas de casos de uso, de procesos de negocio, de actividades, de secuencia y de máquina de estados.

2. Descripción general

La aplicación de gestión hospitalaria es un software único e independiente que reestructura el sistema de gestión de un hospital, reemplazando en gran medida la documentación física y digitalizando los datos fundamentales para el funcionamiento del recinto, con el fin de facilitar su acceso y optimizar los procesos internos de la organización.

Es, fundamentalmente, un sistema de gestión (creación, actualización y revisión) de datos enfocado en los datos de usuarios (exámenes e información personal) por medio de un sistema de cuentas para asegurar la privacidad de los pacientes. El software integra, además, la gestión de información relevante del hospital; como lo son las camas disponibles y el historial de servicios (consultas y urgencias).

El contexto implica necesidades especiales en cuanto a la gestión de la información, debido a la sensibilidad de los datos manejados, y en cuanto a la disponibilidad, pues de ella depende en gran medida el funcionamiento del proceso de negocio de la organización.

2.1 Clases de usuarios y características

La aplicación prevé tres tipos de usuarios distintos junto a un subtipo, cada uno con sus propios permisos y restricciones, que se encuentran listados a continuación.

2.1.1 Pacientes

Los pacientes son los clientes de la organización, quienes únicamente consumen la información disponible en plataforma para su cuenta y solo son capaces de editar su información personal o almacenar cierta cantidad de archivos.

Es el actor con el menor nivel de privilegios de todos, esto por motivos de privacidad y seguridad de los datos almacenados. Los pacientes no necesariamente comprenden el funcionamiento de la aplicación y su uso es altamente esporádico: un paciente normal puede pasar de no acceder a su cuenta durante varios años a utilizarla diariamente durante una semana. Esto se debe a la naturaleza del contexto, en la que este actor solo recurre a la aplicación bajo circunstancias muy particulares (necesidades médicas).

2.1.2 Médicos

Los médicos son actores que pertenecen a la organización, por lo que hacen parte del proceso de negocio interno y necesitan mayor accesibilidad a los datos. El personal médico debe poder acceder a todos y cada uno de los pacientes y ser capaz de realizar la lectura de sus datos en el sistema, entre ellos la información personal, historias médicas, notas de evolución y toda la documentación anexada.

Son uno de los actores más activos dentro del sistema, sus accesos a la plataforma son constantes y rara vez estarán más de una semana sin acceder a su cuenta. Normalmente, un médico accederá seis días de la semana a la aplicación durante al menos 4 o 5 horas, pero su acceso puede extenderse hasta 24 horas de forma continua, e incluso más.

2.1.3 Administradores

Los administradores poseen el nivel más alto de permisos dentro de todo el sistema pues no solo pueden acceder a toda la información de los pacientes sino también del personal médico, y tienen a su disposición la capacidad de crear cuentas de usuarios médicos y administradores y pueden acceder al sistema de gestión de camas.

Este rol también tiene acceso al sistema de indicadores del hospital, que les permite visualizar la disponibilidad de camas del hospital, los pacientes ingresados, la cantidad de servicios por día y muchos otros datos relevantes para la gestión.

2.1.4 Super administrador

El super administrador es un rol que existe fundamentalmente para resolver un problema de seguridad intrínseco en el sistema de usuarios, y es que resulta inseguro permitir a todos los administradores crear cuentas de otros administradores.

Con el objetivo de eliminar la brecha de seguridad mencionada, el super administrador es un usuario único, creado desde el momento en el que se ejecuta por primera vez el sistema en el servidor, que tiene todos los permisos del rol de administrador, pero es el único con la capacidad de crear administradores del sistema.

2.2 Supuestos y dependencias

Si bien es cierto que las pruebas del software serán realizadas en la nube con un servidor de Oracle, se espera que la aplicación final se ejecute en un servidor local gestionado por el hospital. Aunque es posible (y recomendable) que elementos fundamentales del sistema, tales como la base de datos, sean almacenados de manera redundante a nivel local y en la nube para obtener una mayor seguridad en los datos.

Que la aplicación se ejecute en un servidor físico perteneciente a la organización tiene beneficios en cuanto a accesibilidad, tiempos de respuesta y facilidad de modificación; sin embargo, también trae consigo nuevas problemáticas a nivel de ciberseguridad, costos y talento humano involucrado.

Aun así, estas complejidades son menester de la organización, pues escapan a las funcionalidades del software. Por otro lado, esta información permite tener certeza total de que el cliente desea una aplicación web, que bajo todas las circunstancias posibles será gestionada por un servidor, lo que hace adecuado las pruebas de producción en el servidor Oracle en la nube.

3. Requerimientos funcionales

Esta sección abarca todas las necesidades fundamentales del sistema, funcionando como un “deber ser” de las funcionalidades entregadas al usuario. Junto a otros requerimientos, es el fundamento del software y el diagrama base para la construcción de tareas durante el desarrollo del proyecto.

3.1 Gestionar camas

3.1.1 Descripción y prioridad

ID: RF–001.

El sistema debe permitir la gestión de las camas del hospital, permitiendo realizar operaciones CRUD (crear, actualizar, leer y eliminar) según las necesidades de los administradores. Las asignaciones de camas a pacientes se realizarán mediante el método de actualización, ingresando la información del paciente y otros datos pertinentes.

Prioridad: Alta.

3.1.2 Secuencias de estímulo/respuesta

El usuario debe acceder al menú de gestión de camas y luego seleccionar la operación desea realizar. En caso de crear o eliminar camas, debe ingresar la cantidad, y cuando desee actualizar una cama para asignarla, debe establecer el paciente que la utilizará.

3.1.3 Requisitos funcionales

REQ-001.1: El sistema debe permitir que se puedan agregar nuevas camas.
REQ-001.2: El sistema debe permitir que se puedan actualizar el estado de la cama de los pacientes hospitalizados: ocupado o desocupado.
REQ-001.3: El sistema debe permitir que se puedan eliminar camas.
REQ-001.4: El sistema debe ser capaz de obtener el estado de todas las camas dentro del hospital.

3.1.4 Excepciones

- No debe ser posible eliminar camas que se encuentran asignadas a un paciente.
- El sistema no debe permitir asignar pacientes si no hay camas disponibles.

3.2 Gestionar historias clínicas

3.2.1 Descripción y prioridad

ID: RF–002.

El sistema debe permitir la gestión de las historias clínicas de cada paciente mediante operaciones de creación, actualización y lectura, sin embargo, no debe ser posible realizar el borrado.

Prioridad: Alta.

3.2.2 Secuencias de estímulo/respuesta

El usuario ingresará la identificación del paciente cuya historia clínica desea crear o actualizar. Posteriormente, el sistema le permitirá realizar las modificaciones pertinentes del documento.

3.2.3 Requisitos funcionales

REQ-002.1: El sistema debe ser capaz de crear nuevas historias clínicas a los nuevos pacientes del sistema.
REQ-002.2: El sistema debe permitir que los doctores sean capaces de actualizar la historia clínica de los pacientes ya existentes en el sistema.
REQ-002.3: El sistema debe permitir que los doctores puedan obtener las historias clínicas de sus pacientes. Además, los pacientes también pueden acceder a sus propias historias clínicas.

REQ-002.4: El sistema debe ser capaz de almacenar las historias clínicas de los pacientes del hospital sin riesgo de pérdidas.
REQ-002.5: El sistema debe permitir la consulta de las historias clínicas por medio de diversos criterios (número de documento de identidad del paciente, tipo de documento de identidad, nombre del paciente, última fecha de consulta, etc.).

3.2.4 Excepciones

- No debe ser posible eliminar las historias de un paciente.

3.3 Gestionar usuarios

3.3.1 Descripción y prioridad

ID: RF–003.

El sistema debe permitir la gestión de los usuarios del hospital, teniendo en cuenta que existen tres roles diferentes: personal médico, administrador y paciente, cada uno de ellos con sus propios permisos y restricciones. Es posible encontrar las restricciones de cada rol usuario dentro de la sección *2.1 clases de usuarios y características.*

Las operaciones de gestión incluyen creación del usuario, actualización de la información de la cuenta y lectura de esta, sin embargo, no debe ser posible eliminar una cuenta; contrario a esto, debe existir un estado de "desuso" dentro de las bases de datos para aislar las cuentas de los pacientes que se encuentran inactivos dentro del sistema.

Prioridad: Alta.

3.3.2 Secuencias de estímulo/respuesta

Para la creación de una nueva cuenta de cualquier rol, debe ingresarse los datos del usuario al que pertenecerá, incluyendo información personal, de contacto y otros datos relevantes.

3.3.3 Requisitos funcionales

REQ-003.1: El sistema debe permitir a los administradores la creación de nuevos usuarios en el sistema, especificando la información esencial (número de documento de identificación, tipo de documento de identificación, nombre completo, fecha de nacimiento y sexo), rol, contraseña, teléfono de contacto y email.
REQ-003.2: El sistema debe permitir a los usuarios la actualización de su información no esencial (a excepción del rol).
REQ-003.3: El sistema no debe eliminar a los usuarios de manera inmediata, debe mantener a los usuarios durante cierto periodo de tiempo como "inactivos"
REQ-003.4: El sistema debe implementar mecanismos de seguridad para el almacenamiento de contraseñas dentro de la base de datos.
REQ-003.5: El sistema debe mantener un registro de acciones realizadas por cada usuario.
REQ-003.6: El sistema debe asegurar el correcto almacenamiento de los usuarios del sistema.

3.3.4 Excepciones

- No debe ser posible eliminar completamente una cuenta de manera directa, sino que esta será enviada al estado de inactividad.

- No se debe permitir que dos usuarios diferentes cuenten con el mismo número de identificación a la hora de crear una cuenta.

3.4 Generar reporte de indicadores

3.4.1 Descripción y prioridad

ID: RF–004.

El sistema debe ser capaz de analizar los datos de flujo de pacientes dentro del hospital y generar reportes de indicadores fundamentales para la gestión, tales como ocupación unitaria y porcentual, servicios ofrecidos por día, altas por servicio, altas por día y otros varios.

Prioridad: Alta

3.4.2 Secuencias de estímulo/respuesta

Al accederse al sistema de gestión de indicadores, este debe desplegar los indicadores fundamentales del hospital.

3.4.3 Requisitos funcionales

REQ-004.1: El sistema debe mantener un registro del flujo del hospital, lo cual incluye mantener registro del uso de las camas en tiempo real y durante periodos de tiempo específicos.
REQ-004.2: El sistema debe identificar los días con mayor flujo de pacientes en el año.
REQ-004.3: El sistema debe generar reportes de la estancia promedio por servicio.
REQ-004.4: El sistema debe dar informes de la cantidad de admisiones y altas por día.

3.4.4 Excepciones

No se contemplan excepciones para esta sección del sistema más allá de los permisos de cada rol.

3.5 Diferenciar ingresos

3.5.1 Descripción y prioridad

ID: RF–005.

El sistema debe permitir seleccionar uno de entre dos tipos de motivos de ingreso: urgencia y consulta. Las urgencias utilizarán una cama del hospital, mientras que las consultas no requieren de una.

Nota: los ingresos de urgencias requieren datos adicionales referentes a los signos vitales del paciente.
Prioridad: Alta.

3.5.2 Secuencias de estímulo/respuesta

Al intentar ingresar un paciente al sistema, será posible seleccionar si su ingreso es por urgencia o consulta.

3.5.3 Requisitos funcionales

REQ-004.1: El sistema permite clasificar a los pacientes como "consulta" o "urgencia" al momento de su ingreso al hospital.
REQ-004.2: El sistema no debe asignar cama a los pacientes ingresados como "consulta".
REQ-004.3: El sistema debe asignar cama a los pacientes ingresados como "urgencia".
REQ-004.4: El sistema debe permitir el cambio de clasificación de un paciente de "consulta" a "urgencia" si el doctor lo ve necesario.

3.5.4 Excepciones

No se contemplan excepciones para esta sección del sistema más allá de los permisos de cada rol.

3.6 Generar alertas

3.6.1 Descripción y prioridad

ID: RF–006.

El sistema debe generar automáticamente alertas de seguridad cuando se llega a niveles de ocupación demasiado altos, el umbral desde el que se genera una alerta debe ser seleccionado por un administrador.

Prioridad: Baja.

3.6.2 Secuencias de estímulo/respuesta

El sistema detecta que la ocupación rebasó el umbral de alerta y genera la notificación automáticamente.

3.6.3 Requisitos funcionales

REQ-007.1: El sistema debe permitir a los administradores definir el valor umbral.
REQ-007.2: El sistema debe generar alertas automáticas cuando los niveles de ocupación de las camas superen el valor umbral.
REQ-007.3: El sistema debe mostrar la alerta en la interfaz a todas las sesiones de los administradores cuando estos entren a la plataforma.
REQ-007.4: El sistema debe permitir a los administradores avisar al personal médico para que utilicen mejor los recursos del hospital.

3.6.4 Excepciones

No se contemplan excepciones para esta sección del sistema más allá de los permisos de cada rol.

4. Requerimientos no funcionales

4.1 Seguridad y privacidad de los datos

4.1.1 Descripción y prioridad

ID: RNF–001.

El sistema debe implementar medidas robustas de seguridad y privacidad para proteger la información sensible de los pacientes y cumplir con las regulaciones de protección de datos personales en salud.

Prioridad: Alta.

4.1.2 Requisitos funcionales

RNF-001.1: El sistema debe implementar autenticación segura para todos los usuarios que tengan acceso a los datos sensibles.
RNF-001.2: Se deben implementar mecanismos de control de acceso basados en roles para asegurar que los usuarios solo puedan acceder a la información necesaria para su función.

4.1.3 Criterios de aceptación

- El sistema debe aprobar una evaluación de seguridad realizada por el equipo de desarrollo.
- Los registros de auditoría deben estar completos y ser inalterables.

4.2 Rendimiento y escalabilidad

4.2.1 Descripción y prioridad

ID: RNF–002.

El sistema debe ser capaz de manejar altos volúmenes de usuarios concurrentes, manteniendo tiempos de respuesta rápidos y siendo escalable para futuro crecimiento.

Prioridad: Alta.

4.2.2 Requisitos funcionales

RNF-002.1: El sistema debe ser capaz de manejar al menos 1000 usuarios concurrentes sin degradación significativa del rendimiento.
RNF-002.2: Los tiempos de respuesta no deben exceder los 2 segundos bajo carga normal.
RNF-002.3: Las bases de datos debe ser capaz de manejar cientos de miles de registros en pacientes sin impacto significativo en el rendimiento.

4.2.3 Criterios de aceptación

- Se deben realizar pruebas de estrés y de carga que demuestren que el sistema es capaz de soportar el volumen de usuarios especificado.
- Se debe tener un monitoreo de los tiempos de respuesta para comprobar que se cumplan los tiempos de respuesta especificados.

4.3 Disponibilidad y fiabilidad

4.3.1 Descripción y prioridad

ID: RNF–003.

El sistema debe mantener un alto nivel de disponibilidad y fiabilidad, minimizando el tiempo de inactividad y asegurando la integridad de los datos.

Prioridad: Alta

4.3.2 Requisitos funcionales

RNF-003.1: El sistema debe tener una buena disponibilidad de al menos 95% de las horas de operación normales.
RNF-003.2: Se debe contar implementar un sistema de respaldo y recuperación que permita realizar una restauración completa de los datos.
RNF-003.3: Se deben realizar copias de seguridad incrementales diarias y copias de seguridad completas de todos los datos semanales.

4.3.3 Criterios de aceptación

- Los registros de tiempo de actividad deben demostrar el cumplimiento del objetivo de disponibilidad.
- No debe haber perdida de datos debido a fallos del sistema.

4.4 Ejecución en el servidor con contenedores Docker

4.4.1 Descripción y prioridad

ID: RNF–004.

El sistema debe estar construido implementando contenedores Docker para facilitar su ejecución en la nube.

Prioridad: Alta.

4.4.2 Requisitos funcionales

RNF-004.1: El código desarrollado debe contener toda la información de ejecución de los contenedores Docker.
RNF-004.2: Las pruebas de ejecución serán realizadas en un servidor en la nube de Oracle que consuma los contenedores construidos.

4.4.3 Criterios de aceptación

- El sistema se comporta adecuadamente en la ejecución dentro del servidor utilizando los contenedores Docker.

4.5 Permanencia y almacenamiento de datos

4.5.1 Descripción y prioridad

ID: RNF–005.

La información de todos los usuarios junto a sus archivos relacionados debe ser permanente en el sistema. Esto implica no solo el uso de bases de datos sino también la inexistencia de acciones de borrado absoluto dentro de la aplicación, pues va en contra de las políticas del hospital e implica una falla en el proceso de negocio.

Prioridad: Alta.

4.5.2 Requisitos funcionales

RNF-005.1: Se construyen bases de datos SQL para almacenar la información de los usuarios.
RNF-005.2: Se construyen bases de datos NoSQL para llevar un registro de las interacciones de los usuarios con la aplicación.
RNF-005.3: Se construye una base de datos capaz de albergar los documentos de los usuarios para consultas posteriores.

4.5.3 Criterios de aceptación

- La información almacenada en cada base de datos tiene permanencia en el sistema, es decir, es accesible entre secciones independientemente de la cantidad de tiempo que pase entre ellas.

5. Requerimientos de usuario

5.1 Facilitar el acceso a la información

ID: RU–001.

El sistema debe permitirle a los pacientes acceder a toda su información médica, que incluye notas médicas y resultados de exámenes.

Prioridad: Alta.

5.1.1 Requisitos funcionales

RU-001.1: Los pacientes son capaces de acceder a sus resultados de exámenes por medio de la aplicación.
RU-001.2: Los pacientes pueden consultar sus historias médicas e imprimirlas por medio de la aplicación.

5.1.2 Criterios de aceptación

- Se cumple con los requisitos funcionales.

5.2 Interfaz amigable

ID: RU–002.

La interfaz de usuario resulta intuitiva y fácil de aprender gracias a la disposición clara de los objetos e información en pantalla. Además, la selección de colores resulta amigable y refleja los objetivos e imagen de marca que desea transmitir el hospital.

Prioridad: Media.

5.2.1 Requisitos funcionales

RU-002.1: La interfaz de usuario resulta intuitiva gracias a una disposición evidente de los objetos.
RU-002.2: Se utilizan los colores de la marca del hospital.

5.2.2 Criterios de aceptación

- Se cumple con los requisitos funcionales.

6. Requerimientos del sistema

6.1 Encriptado de claves

ID: RS–001.

Con el objetivo de proteger al sistema de filtraciones de claves, las contraseñas de usuarios dentro de las bases de datos deben almacenarse encriptadas.

Nota: Este requerimiento se complementa con el RNF–001, formando junto a este y los requerimientos RS–002 y RS–003 el fundamento de la capa de seguridad de la aplicación.

Prioridad: Alta.

6.1.1 Requisitos funcionales

RS-001.1: Las contraseñas de los usuarios deben ser almacenadas utilizando funciones hash.

6.1.2 Criterios de aceptación

- Se cumple con los requisitos funcionales.

6.2 Implementación de Tokens.

ID: RS–002.

La información de sesión debe estar condensada en tokens de usuario, centralizando los datos sensibles y facilitando su protección en la web.

Nota: Este requerimiento se complementa con el RNF–001, formando junto a este y los requerimientos RS–001 y RS–003 el fundamento de la capa de seguridad de la aplicación.

Prioridad: Media.

6.2.1 Requisitos funcionales

RS-002.1: Las contraseñas de los usuarios deben ser almacenadas utilizando funciones hash.

6.2.2 Criterios de aceptación

- Se cumple con los requisitos funcionales.

6.3 Registro de auditoría.

ID: RS–003.

El sistema debe almacenar las interacciones de todos y cada uno de los usuarios de forma permanente en un registro, que contenga tanto la acción realizada como el usuario que estuvo involucrado.

Nota: Este requerimiento se complementa con el RNF–001, formando junto a este y los requerimientos RS–001 y RS–002 el fundamento de la capa de seguridad de la aplicación.

Prioridad: Media.

6.3.1 Requisitos funcionales

RS-003.1: Se almacena un registro de interacciones en una base de datos NoSQL.
RS-003.2: El registro de auditoría es inalterable para todos los roles en el sistema.

6.3.2 Criterios de aceptación

- El superadministrador puede acceder al registro para auditar el sistema, pero no tiene la capacidad de modificarlo de ningún modo.
- El registro de auditoría es permanente entre las sesiones y el tiempo, además de almacenar todas las interacciones de los usuarios con la aplicación.

6.4 Capacidades del servidor

ID: RS–004.

El sistema debe ser capaz de ejecutarse de forma eficiente en un servidor en la nube que posea 4GB de RAM, 4GB de almacenamiento y un procesador de 2 núcleos a 3.2 GHz.

Nota: Este requerimiento hace referencia al servidor de pruebas. Es un requisito planteado por el cliente para asegurar que el sistema se encuentra lo suficientemente optimizado como para ejecutarse adecuadamente a gran escala en su servidor personal.

Prioridad: Alta.

6.4.1 Requisitos funcionales

RS-004.1: El servidor de pruebas tiene 4GB de RAM, 4GB de almacenamiento y un procesador de dos núcleos de 3.2 GHz.
RS-004.2: El sistema cumple adecuadamente las especificaciones planteadas en los requerimientos RNF-002 y RNF-003 en el servidor de pruebas.

6.4.2 Criterios de aceptación

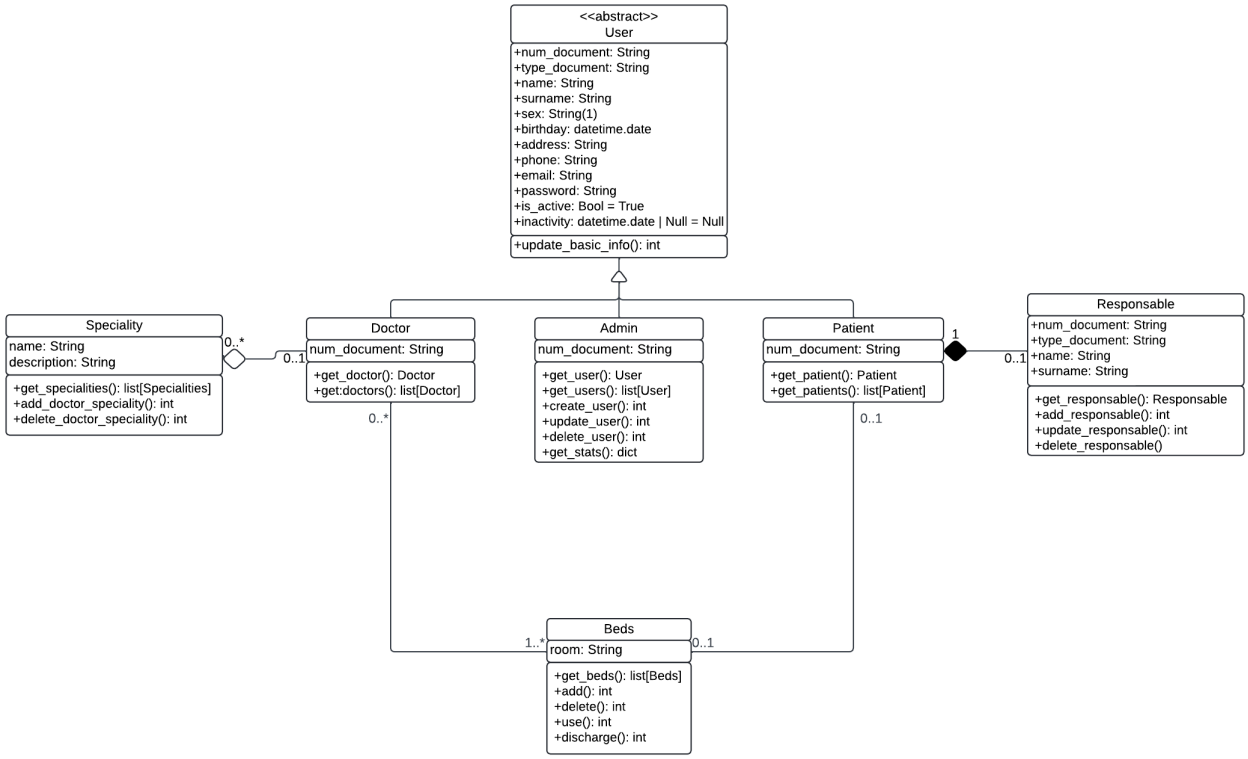
- El sistema completa las pruebas establecidas en los requisitos funcionales.

7. Modelos de análisis

En esta sección se presenta documentación descriptiva de la lógica de negocios de la aplicación, describiendo tanto los objetos implicados en el sistema como las interacciones que realizan los usuarios en cada uno de los casos de uso.

7.1 Diagrama de clases

El diagrama de clases sirve para representar cada una de las clases contempladas dentro del sistema, sus atributos, métodos y las relaciones que pueden llegar a existir entre ellas. Se espera que sirva como fundamento para el desarrollo del código del proyecto y facilite la comprensión de la lógica de negocio.



Nota: El diagrama anterior también es accesible mediante el enlace denominado “Diagramas”, que lo redirigirá a un link de Google drive con todos los diagramas almacenados en SVG para una mayor calidad de visualización.

En el diagrama anterior se contempla una clase abstracta que sirve como padre de todos los tipos de usuario existentes, llamada “User”. De ella heredan Doctor, Admin y Patient, cada una con propiedades que resultan necesarias para el adecuado funcionamiento del proceso de negocios del hospital y el funcionamiento de la aplicación.

Las camas son administradas mediante la clase “Beds”, y “Responsable” se refiere a un individuo asociado a los pacientes, que los acompaña en caso de que se encuentre internados. Finalmente, “Speciality” hace referencia a la especialidad de los doctores, que permite identificarlos con facilidad dentro del sistema.

7.2 Modelos y notación de los procesos de negocio

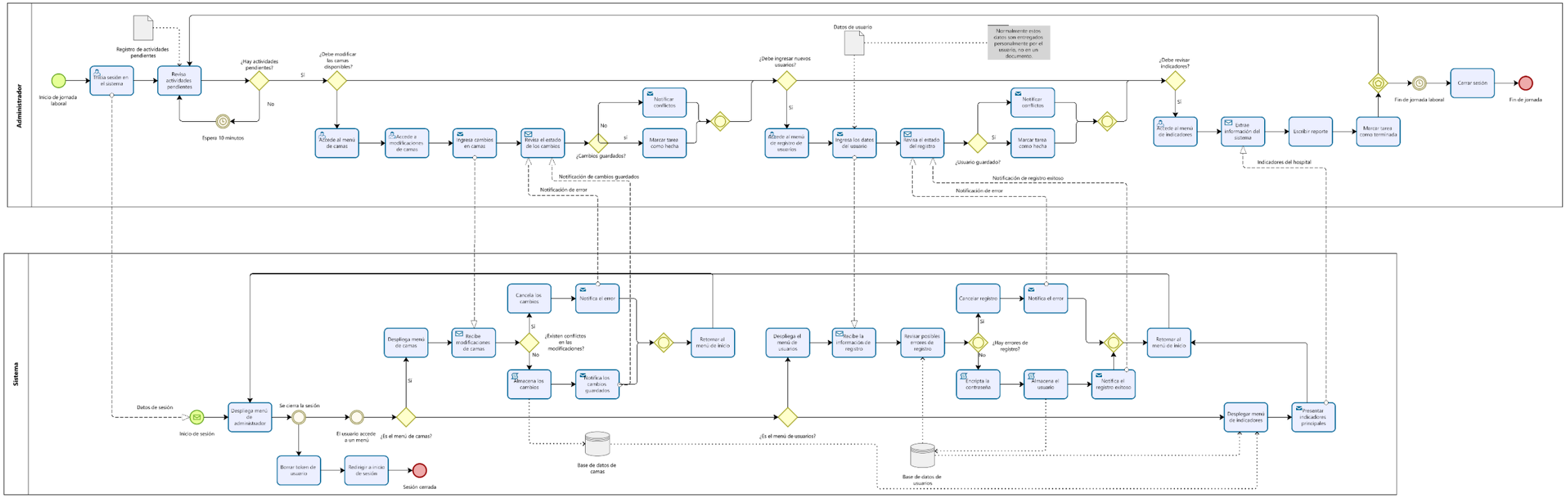
Los modelos de procesos de negocio facilitan la comprensión, análisis y optimización del flujo de trabajo. En este caso, servirán como apoyo para la comprensión de los requerimientos, ayudando al equipo de desarrollo a entender el flujo de actividades, decisiones y eventos dentro de la mayor parte de las interacciones del usuario.

Debido a la distribución de permisos por roles que posee la aplicación, no es óptimo unificar todas las actividades en un mismo BPD, sino que estas serán divididas en tres diagramas distintos teniendo en cuenta que los objetivos principales de cada rol son: administradores – evaluación de indicadores y gestión de camas y pacientes, médicos – revisión y edición de documentos e ingreso de pacientes, pacientes – revisión de documentos personales.

7.2.1 Administradores

Las funciones principales del rol de administrador son la creación de usuarios dentro del sistema, administración de camas y revisión de indicadores. Estas tres alternativas se encuentran contempladas

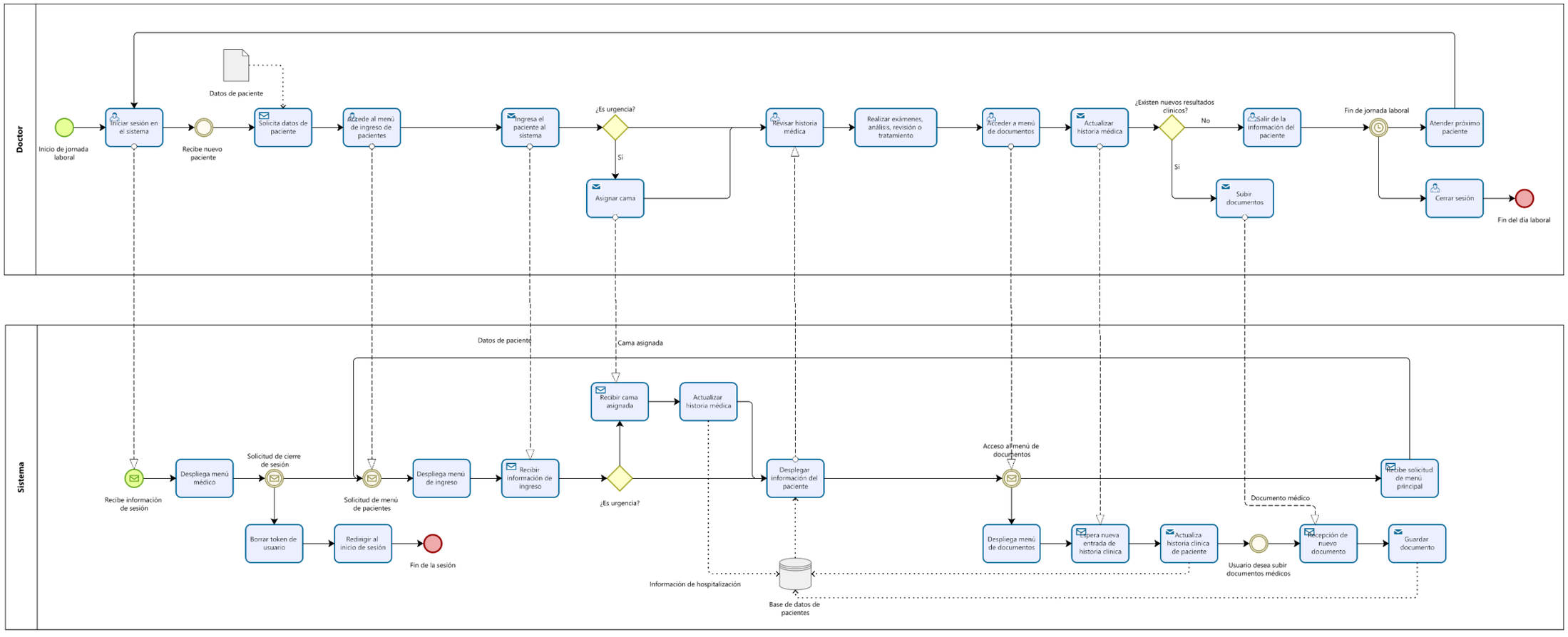
Si bien es cierto que existe una subclase denominada “Super administrador” realmente esta no tiene nuevas funcionalidades, sino el permiso de creación de administradores, que sigue el mismo flujo de trabajo que la creación de usuarios comunes, por lo que no resulta necesario crear un nuevo diagrama de proceso de negocio para describir dicha instancia. De la misma forma, el sistema solo puede ser auditado por el super administrador, y esta acción sigue el mismo flujo de trabajo que la revisión de indicadores.



Nota: El diagrama anterior también es accesible mediante el enlace denominado “Diagramas”, que lo redirigirá a un link de Google drive con todos los diagramas almacenados en SVG para una mayor calidad de visualización.

7.2.2 Médicos

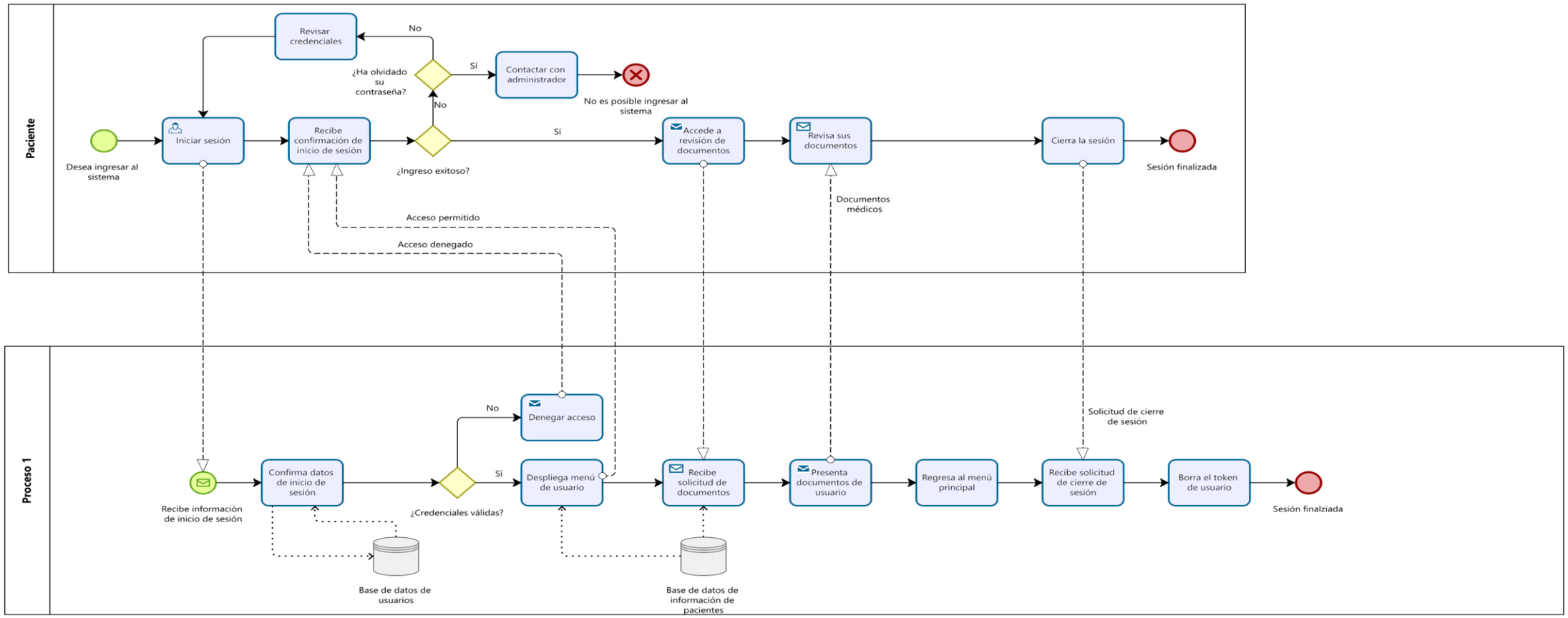
Las funciones principales del rol de médico son el ingreso de pacientes en forma de consultas y urgencias, además de la edición de historias médicas y la capacidad de subir documentos a los archivos de los pacientes, actualizando así los resultados de exámenes y análisis de laboratorios almacenados.



Nota: El diagrama anterior también es accesible mediante el enlace denominado “Diagramas”, que lo redirigirá a un link de Google drive con todos los diagramas almacenados en SVG para una mayor calidad de visualización.

7.2.3 Pacientes

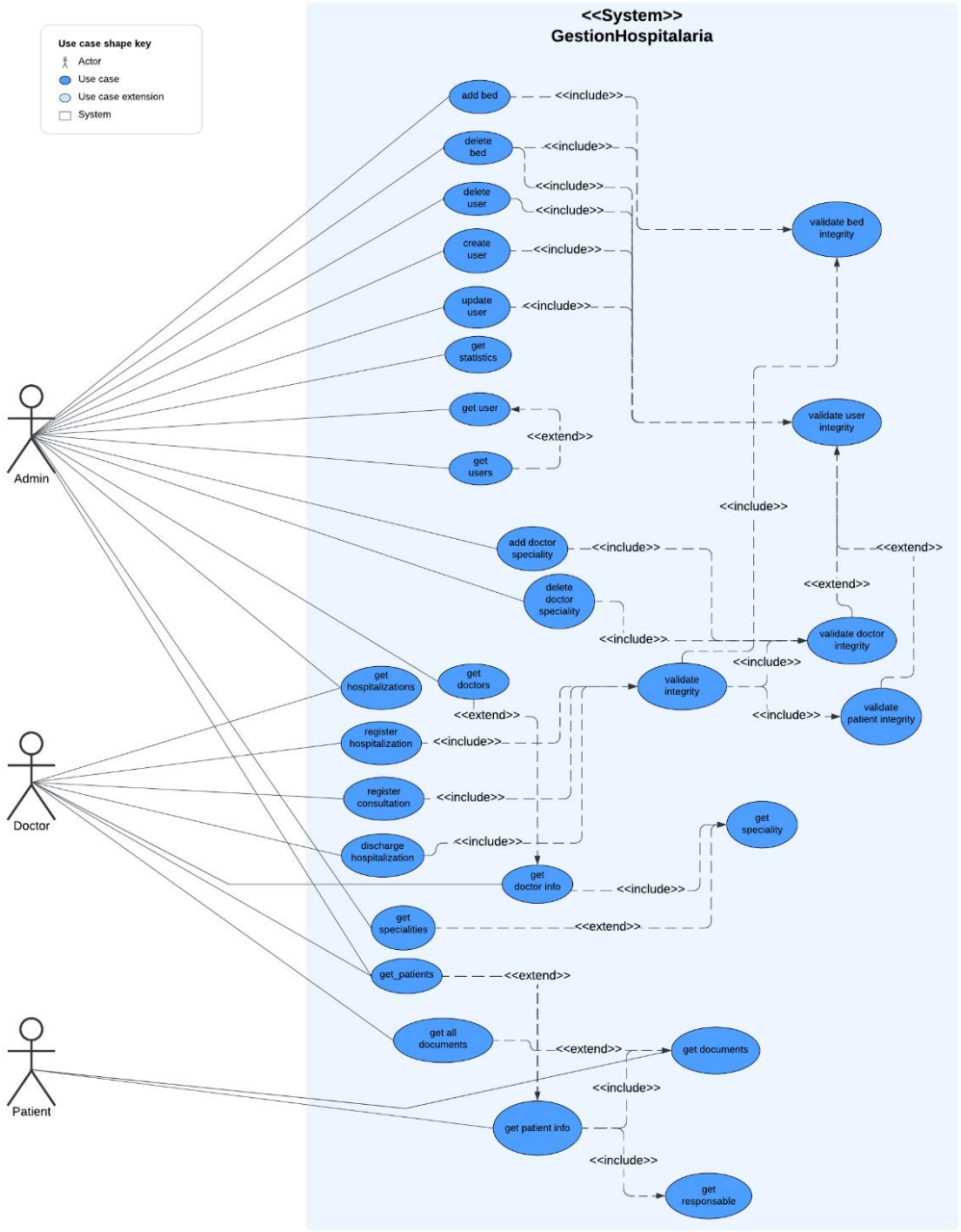
La única actividad que los pacientes pueden realizar dentro del sistema, además de acceder a su usuario personal creado por un administrador, es visualizar sus datos personales y documentos clínicos registrados. Estos documentos pueden ser historias clínicas, resultados de exámenes y laboratorio, etc., existiendo un mismo proceso para acceder a todos ellos, por lo que el diagrama de proceso de negocio de los pacientes solo comprende un único flujo de procesos.



Nota: El diagrama anterior también es accesible mediante el enlace denominado “Diagramas”, que lo redirigirá a un link de Google drive con todos los diagramas almacenados en SVG para una mayor calidad de visualización.

7.3 Diagrama de casos de uso

El diagrama de casos de uso describe todas las interacciones posibles que existen entre los actores involucrados en el uso de la plataforma y el sistema en sí mismo. Es fundamental a la hora de comprender el flujo de procesos interno del sistema y cuáles deben ser todas sus capacidades.



Nota: El diagrama anterior también es accesible mediante el enlace denominado “Diagramas”, que lo redirigirá a un link de Google drive con todos los diagramas almacenados en SVG para una mayor calidad de visualización.

7.4 Descripción de casos de uso

7.4.1 Add bed

Escenario: Agregar una nueva cama al hospital.
Evento desencadenante: Un administrador quiere agregar una cama.
Descripción: Cuando un administrador quiere agregar una nueva cama al hospital, se debe ingresar el cuarto a donde se quiere agregar la cama y se verifica si ya existe previamente una cama dentro del cuarto.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate bed integrity.
Interesados: Hospital: Para tener una mayor capacidad de pacientes que necesiten ser hospitalizados.
Condiciones previas: El cuarto no debe tener ninguna cama asignada previamente.
Condiciones posteriores: La asignación de la nueva cama en la base de datos debe ser creada.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide que quiere agregar camas al sistema. 2. Selecciona el menú de modificación de camas en el menú principal. 3. Ingresa la cama y su habitación. 4. Presiona el botón para registrar.	0.1 Despliega el menú principal de administradores. 2.1 Despliega el menú de modificación de camas. 4.1 Revisa la disponibilidad de la habitación y la cama dentro de la base de datos. 4.2 Añadir cama a la base de datos 4.3 Finaliza la creación de cama.

Excepciones:
4.1 En caso de que la habitación ya se encuentre ocupada, cancelar la operación y enviar notificación de imposibilidad al administrador.

7.4.2 Delete bed.

Escenario: Quitar una cama del hospital.
Evento desencadenante: Un administrador quiere quitar una cama.
Descripción: Cuando un administrador quiere quitar una cama al hospital, se debe ingresar el cuarto a donde se está la cama. Finalmente, se verifica si existe previamente una cama dentro del cuarto dado y si la cama está en uso por algún paciente.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate bed integrity.
Interesados: Hospital: Para poder realizarle trabajo de mantenimiento al cuarto donde está la cama.
Condiciones previas: El cuarto debe tener asignada una cama que no esté en uso por ningún paciente hospitalizado.
Condiciones posteriores: La cama debe ser eliminada de la base de datos.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide eliminar una cama del sistema. 2. Selecciona el menú de modificación de camas en el menú principal. 3. Ingresa los datos de la cama a eliminar. 4. Presiona “Eliminar cama”.	0.1 Despliega el menú principal de administradores. 2.1 Despliega el menú de modificación de camas. 3.1 Busca la cama en la base de datos 4.1 Borra la cama de la base de datos. 4.2 Finaliza el borrado de cama.

Excepciones:
3.1 Es posible que se busque una cama que no está dentro del sistema, caso en el cual no se produce un error, simplemente no se encuentran resultados en la búsqueda.

4.1 En caso de que la cama se encuentre ocupada, no puede ser borrada y el botón de “eliminar cama” se encontrará deshabilitado, por lo que el administrador tendrá que desistir hasta que la desocupe.

7.4.3 Delete user

Escenario: Un administrador quiere eliminar a un usuario para que ya no esté activo dentro del hospital.
Evento desencadenante: El administrador quiere eliminar a un usuario.
Descripción: Cuando un administrador quiere eliminar un usuario, lo que se está haciendo es que se coloca como inactivo dentro de la base de datos especificando su número de documento y su rol o tipo de usuario. Lo primero que se verifica es si el usuario se encuentra activo en el sistema a la hora de “eliminar” el usuario, luego si el usuario que se desea eliminar es un administrador y quien llama la función no es el super administrador, entonces no permite realizar la operación; en caso de que el usuario sea un paciente, se debe verificar que actualmente no se encuentre hospitalizado.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate user integrity.
Interesados: Hospital: Para liberar espacio en la base datos de clientes activos.
Condiciones previas:

- El usuario debe existir.
- En caso de querer eliminar a un administrador, entonces se debe ser el administrador.
- En caso de querer eliminar a un paciente, el paciente no debe estar hospitalizado.

Condiciones posteriores: El usuario debe mantenerse en la base de datos, cambiando su estado a “inactivo”.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide eliminar un usuario 2. Selecciona el menú de opciones de usuario en el menú principal. 3. Selecciona borrar usuarios. 4. Ingresa los datos del usuario. 5. Presiona “Eliminar”.	0.1 Despliega el menú principal de administradores. 2.1 Despliega el menú de usuarios. 3.1 Despliega el menú de borrado de usuarios. 4.1 Busca al usuario en la base de datos. 4.2 Despliega información básica de la cuenta. 5.1 Borra al usuario de la tabla de usuarios activos y lo registra en la tabla de inactivos. 5.2 Finaliza el borrado de usuarios.

Excepciones:

4.1 Es posible que se busque un usuario que no está dentro del sistema, caso en el cual no se produce un error, simplemente no se encuentran resultados en la búsqueda.

7.4.4 Create user

Escenario: Agrega la información básica de un nuevo usuario al sistema.
Evento desencadenante: Un administrador agrega dentro del sistema a un nuevo usuario.
Descripción: Cuando un administrador quiere agregar a un nuevo usuario al sistema, el administrador debe dar toda la información del usuario, y el sistema se encarga de verificar la existencia de algún usuario activo dentro del sistema, de contar con los permisos necesarios, de crear una nueva instancia dentro de la base de datos o de reactivar a un usuario inactivo en caso de ser necesario.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate user integrity.
Interesados:

- Hospital: En caso de agregar un nuevo administrador o doctor, tienen mejor capacidad de atender a las necesidades de los pacientes en el sistema y médicamente respectivamente.
- Paciente: Para ser atendido.

Condiciones previas:

- No debe haber un usuario activo con el mismo número de documento y rol.
- En caso de que el usuario creado sea un administrador, se debe ser un super administrador.

Condiciones posteriores:

- En caso de que el usuario nunca haya estado registrado dentro del sistema, se debe crear una nueva instancia.
- En caso de que el mismo usuario esté como inactivo dentro del sistema, se debe reactivar el estado.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide añadir un usuario 2. Selecciona el menú de opciones de usuario en el menú principal. 3. Selecciona crear usuarios. 4. Ingresa los datos del usuario. 5. Presiona “crear”.	0.1 Despliega el menú principal de administradores. 2.1 Despliega el menú de usuarios. 3.1 Despliega el menú de agregado de usuarios. 5.1 Guarda al usuario en la base de datos. 5.2 Finaliza creación de usuario

Excepciones:
4.1 Es posible que se ingrese un usuario que tenga el mismo número de identificación y rol que otro ya registrado, sin embargo, en estos casos el sistema debe cancelar la creación del usuario y enviar una notificación al administrador.

7.4.5 Update user

Escenario: Un administrador quiere actualizar a un usuario la información de algún paciente.
Evento desencadenante: El administrador quiere actualizar la información de un usuario
Descripción: Cuando un administrador quiere actualizar la información de algún usuario, primero se verifica que exista, de contar con los permisos necesarios y que los cambios que se realicen no presenten problemas de integridad luego.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate user integrity.
Interesados: Usuario: Interesado en actualizar su información.

Condiciones previas:
- El usuario debe existir.
- En caso de querer actualizar a un administrador, entonces se debe ser el super administrador.
- En caso de actualizar el número de documento, no debe estar en la base de datos anteriormente.

Condiciones posteriores: La información del usuario debe cambiar en la base de datos.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide actualizar un usuario 2. Selecciona el menú de opciones de usuario en el menú principal. 3. Selecciona actualizar usuario. 4. Ingresa los datos del usuario. 5. Presiona "Buscar". 6. Ingresa la actualización de datos 7. Presiona "Actualizar".	0.1 Despliega el menú principal de administradores. 2.1 Despliega el menú de usuarios. 3.1 Despliega el menú de actualización de usuarios. 5.1 Busca al usuario en la base de datos. 5.2 Despliega la información del usuario y sus opciones de actualización. 7.1 Guarda al usuario en la base de datos. 7.2 Finaliza la actualización de datos de usuario

Excepciones:
5.1 Es posible que se busque un usuario que no está registrado en la base de datos, sin embargo, esto no debería ocasionar un error, simplemente no se encontrarán resultados en la búsqueda.
7.1 Las actualizaciones no pueden ocasionar conflictos con los datos previamente ingresados, en caso de que el sistema detecte alguno, se cancelará la operación y enviará una notificación de error.

7.4.6 Get statistics

Escenario: Un administrador quiere analizar las estadísticas del hospital.
Evento desencadenante: El administrador va al apartado sobre las estadísticas del hospital.
Descripción: Cuando un administrador quiere ver las estadísticas del hospital, se le retorna el porcentaje de ocupación hospitalaria, promedio de estancias de los pacientes en el hospital y cantidad de admisiones y altas por días.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate user integrity.
Interesados: Hospital: Con las estadísticas del hospital, hay una mejor capacidad de actuar.
Condiciones previas: Ninguna.
Condiciones posteriores: Se le debe retornar en forma clara al administrador las estadísticas.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide revisar las estadísticas. 2. Selecciona el menú de estadísticas en el menú principal. 3. Lee las estadísticas.	0.1 Despliega el menú principal de administradores. 2.1 Solicita la información relevante a la base de datos. 2.2 Calcula los estadísticos programados. 2.3 Produce las gráficas necesarias. 2.4 Despliega las gráficas y estadísticos en la capa de vistas.

Excepciones: No se contemplan excepciones.

7.4.7 **Get user**

Escenario: Obtener la información de un usuario
Evento desencadenante: El administrador quiere obtener la información de algún usuario del sistema.
Descripción: Cuando un administrador quiere obtener la información de algún usuario dentro del sistema, se debe especificar el número de documento y algunos parámetros más para ver más información. Finalmente, se verifica si existe el usuario con el número de documento dado.
Actores: Administrador.
Casos de uso relacionados: Ninguno.
Interesados: Administrador: Interesados en saber la información de algún usuario.
Condiciones previas:

- El número de documento del usuario que se desea buscar debe existir.
- Los valores dados a los parámetros deben ser de tipo booleano.

Condiciones posteriores: Se le debe retornar en forma clara al administrador la información del usuario en base a los parámetros especificados.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide revisar los usuarios del sistema. 2. Selecciona el menú para buscar un usuario. 3. Especifica los parámetros dados. 4. Lee la información del usuario.	0.1 Despliega el menú principal de administradores. 3.1 Hace la solicitud a la base de datos. 3.2 Retorna de forma clara la información del usuario deseado.

Excepciones:
3.1 Retorna un mensaje de error cuando el usuario no existe.

7.4.8 **Get users**

Escenario: Obtener la información de todos los usuarios
Evento desencadenante: El administrador quiere obtener la información de todos los usuarios del sistema.
Descripción: Cuando un administrador quiere obtener la información de todos los usuarios dentro del sistema, se debe especificar el número de documento y algunos parámetros más para ver más información.
Casos de uso relacionados: extiende: Get user.
Interesados: Administrador: Interesados en saber la información de todos los usuarios dentro del sistema.
Condiciones previas: Los valores dados a los parámetros deben ser de tipo booleano
Condiciones posteriores: Se le debe retornar en forma clara al administrador la información del usuario dependiendo de los parámetros dados.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide revisar los usuarios del sistema. 2. Selecciona el menú para mostrar todos los usuarios. 3. Especifica los parámetros deseados. 4. Lee la información de todos los usuarios.	0.1 Despliega el menú principal de administradores. 3.1 Hace la solicitud a la base de datos. 3.2 Retorna de forma clara la información del usuario deseado.

Excepciones: No se contemplan excepciones.

7.4.9 Add doctor speciality

Escenario: Agregar especialidad a un doctor.
Evento desencadenante: El administrador quiere agregarle una especialidad a un doctor.
Descripción: Cuando un administrador agrega una especialidad a un doctor, se valida que no haya problemas de integridad con la información previa del doctor.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate doctor integrity.
Interesados: Hospital: Cuando se puede filtrar en base a la especialidad de un doctor, es mucho más fácil saber a qué doctor llamar en caso de ser necesario, o analizar si existe carencia de personal médico.
Condiciones previas:

- El doctor debe existir dentro del sistema.
- En caso de no existir la especialidad, se debe dar la información necesaria para crearla.
- El doctor no debe tener registrado previamente la especialidad

Condiciones posteriores: Se debe almacenar dentro de la base de datos las especialidades de los doctores.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide agregar una especialidad al sistema. 2. Selecciona el menú de doctores en el menú principal. 3. Selecciona el doctor deseado. 4. Presiona “Añadir especialidades” 5. Selecciona una de las especialidades existentes o selecciona “Añadir nueva especialidad”. 6. Ingresa la información de la nueva especialidad. 7. Presiona “Añadir” en la creación de especialidades. 8. Presiona “Añadir” en las especialidades del doctor.	0.1 Despliega el menú principal de administradores. 2.1 Consulta los doctores en la base de datos. 2.2 Despliega a los doctores del hospital 2.3 Despliega toda la información del doctor seleccionado. 4.1 Despliega todas las especialidades existentes. 5.1 Despliega el menú de creación de especialidades. 7.1 Añade la especialidad a la base de datos. 8.1 Añade la relación entre el doctor y la especialidad en la base de datos. 8.2 Finaliza el proceso de añadir especialidad.

Excepciones: No se contemplan excepciones.

7.4.10 delete doctor speciality

Escenario: Eliminar una especialidad a un doctor.
Evento desencadenante: El administrador quiere eliminar una especialidad a un doctor.
Descripción: Cuando un administrador quiere eliminar una especialidad a un doctor, se valida que sí se pueda realizar la operación, para finalmente hacerse.
Actores: Administrador.
Casos de uso relacionados: Incluye: validate doctor integrity.
Interesados: Doctor: Puede darse el caso de que exista un error con su información profesional, para la cual desea eliminar datos inexactos.

Condiciones previas:

- El doctor debe existir dentro del sistema.
- El doctor debe tener registrado previamente la especialidad.

Condiciones posteriores: Se debe realizar el cambio dentro de la base de datos las especialidades de los doctores.

Flujo de actividades	
Actor	**Sistema**
1. El administrador decide agregar una especialidad al sistema. 2. Selecciona el menú de doctores en el menú principal. 3. Selecciona el doctor deseado. 4. Presiona "Añadir especialidades" 5. Selecciona el botón de eliminar en una de las especialidades ya añadidas al doctor.	0.1 Despliega el menú principal de administradores. 2.1 Consulta los doctores en la base de datos. 2.2 Despliega a los doctores del hospital 2.3 Despliega toda la información del doctor seleccionado. 4.1 Despliega todas las especialidades relacionadas al doctor. 5.1 Elimina la especialidad seleccionada de la base de datos. 5.2 Termina el proceso de borrado.

Excepciones: No se contemplan excepciones.

8. Anexos

8.1 Diagramas

Todos los diagramas inscritos en este documento se encuentran almacenados en formato SVG y PDF dentro de una carpeta de Google Drive accesible mediante: https://drive.google.com/drive/folders/1AfP9VbV6vc0N3cXXB5Qu3GksewvuDKUX?usp=sharing