HospiTEC

Documentación Técnica v1.0

HospiTEC - Documentación Técnica

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computadores

Guillen Jorge, Marín Kendall, Núñez Henry, Solís Isaac, Zheng Kun

Versión 1.0

Tabla de contenidos

Objetivos

Objetivo general

- Desplegar una aplicación web totalmente funcional que cumpla con los requisitos especificados, proporcionando una interfaz intuitiva y atractiva para que los usuarios interactúen con el sistema de manera eficiente y efectiva.

Objetivos específicos

1. Diseñar un sistema que administre tres tipos de usuarios diferentes (doctores, pacientes y personal administrativo).
2. Desarrollar un sistema de registro para los pacientes nuevos.
3. Construir un sistema de inicio de sesión para los trabajadores del hospital y los pacientes.
4. Elaborar un sistema que gestione la información de las facilidades del hospital, la información de pacientes y personal.
5. Elaborar un sistema que gestione la información del equipo perteneciente a los espacios de reservación en el hospital.
6. Construir un sistema que se encargue de generar reportes de los departamentos a partir de las evaluaciones recuperadas de los pacientes recuperadas de la base de datos en MongoDB.
7. Diseñar un sistema que permita el ingreso de doctores o enfermeros por parte del personal administrativo.
8. Diseñar un sistema que permita la gestión de pacientes por parte de los doctores.
9. Elaborar un sistema que permita la reservación de camas y la respectiva gestión por parte de los administrativos.
10. Elaborar un sistema que permita la creación y gestión de salones con sus respectivas características, gestionada por los administradores.

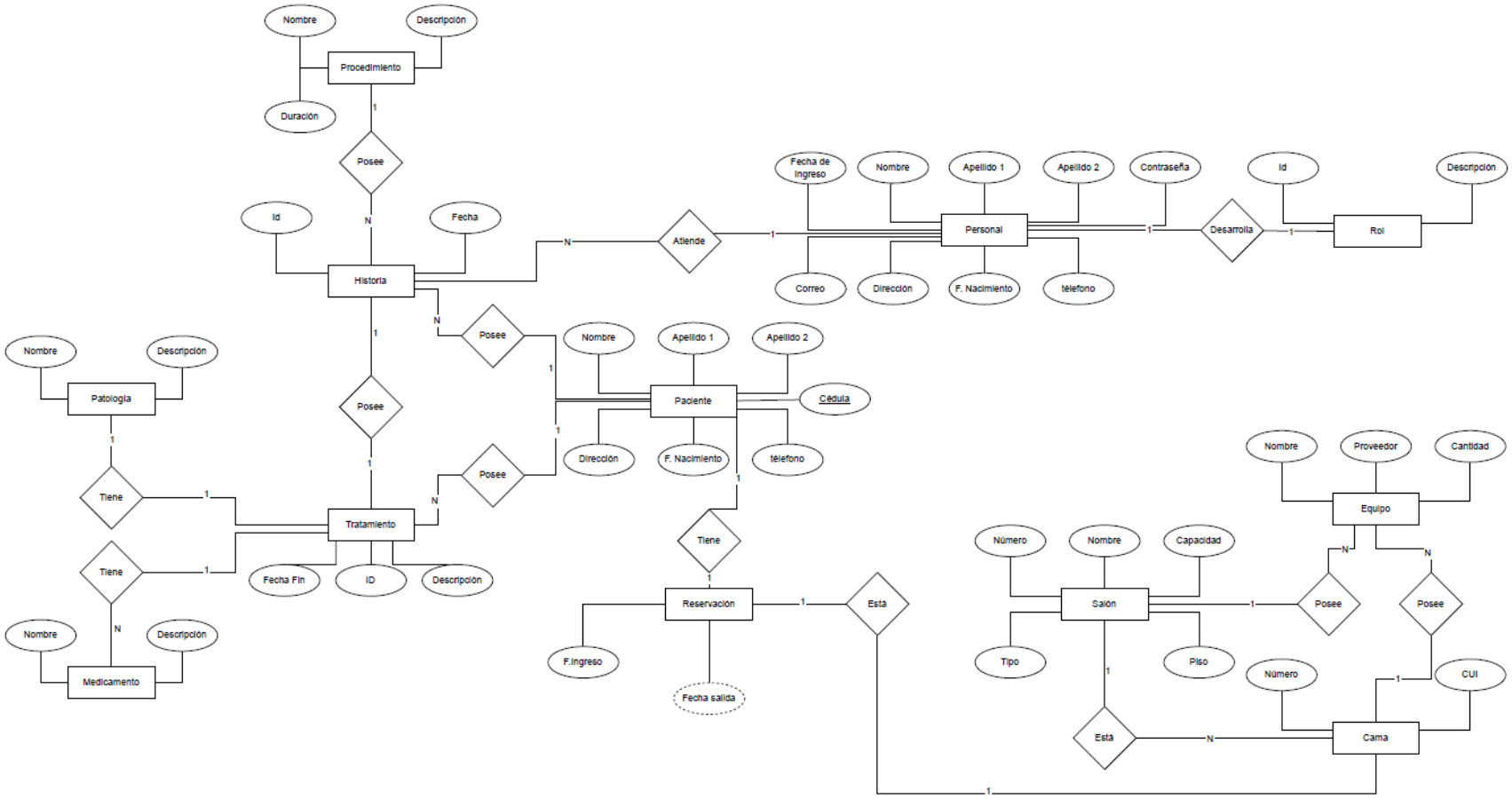
Descripción del problema

Debido a la última pandemia de COVID-19 y al colapso que afectó al sistema médico nacional, el Instituto Tecnológico ha decidido transformar su clínica de salud en el Hospital Tecnológico para que pueda atender cualquier emergencia a nivel nacional. Para asegurar la sostenibilidad financiera del hospital, se ha decidido que los servicios ofrecidos por el Hospital también estarán disponibles para el público en general, lo cual hace necesario crear un sistema de gestión hospitalaria. HospiTEC se compone principalmente de tres vistas:

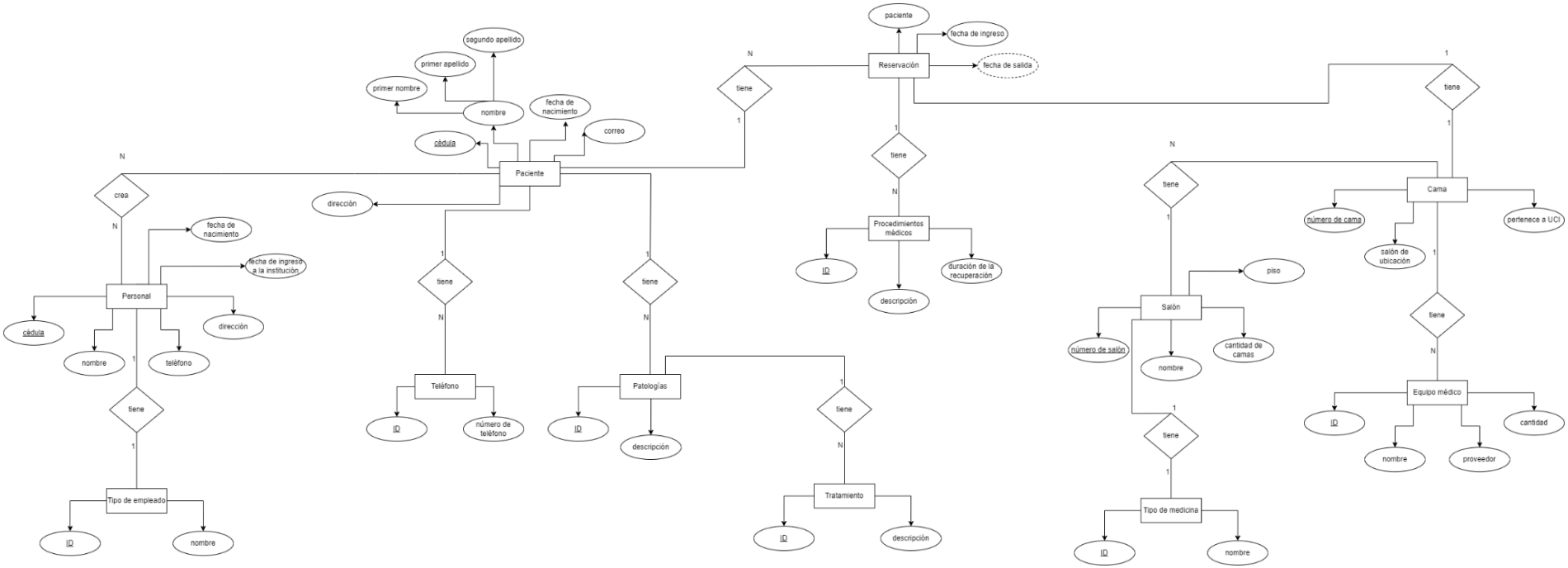
1. Vista paciente: se le permite a los pacientes realizar una reservación para procedimientos médicos disponibles en el hospital. Además, ofrecerá la posibilidad de consultar su historial clínico y calificar tanto la estadía en el hospital como al personal que lo atendió.
2. Vista personal administrativo: se le permite al personal administrativo ingresar y gestionar información general del sistema, incluyendo salones, camas, equipo médico, procedimientos, usuarios, y más.
3. Vista doctor: se le permite a los doctores agregar pacientes, modificar su historial clínico y registrar los medicamentos utilizados en el tratamiento de cualquier patología.

Modelo conceptual

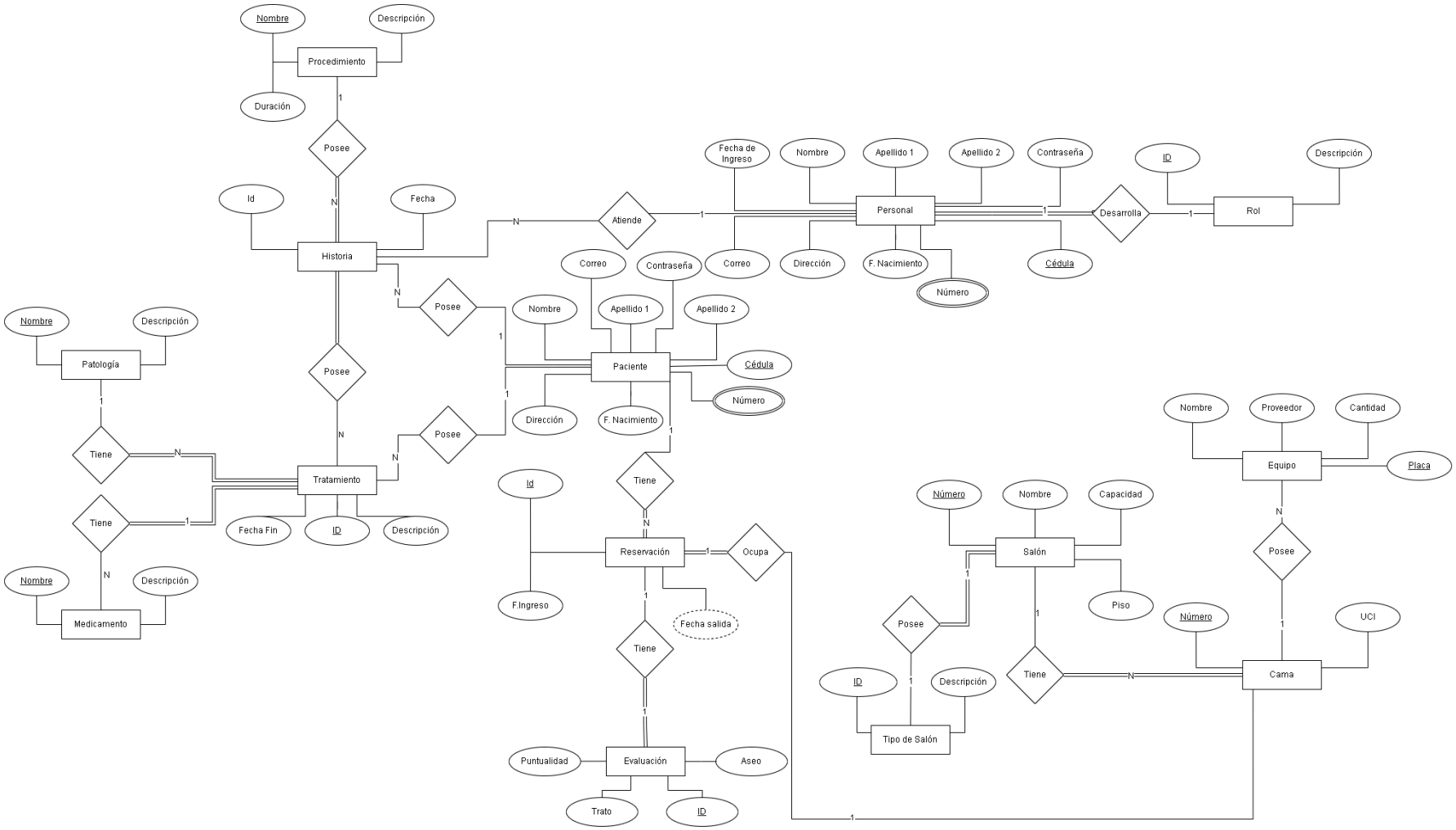
➢ Primer modelo conceptual



➢ Segundo modelo conceptual



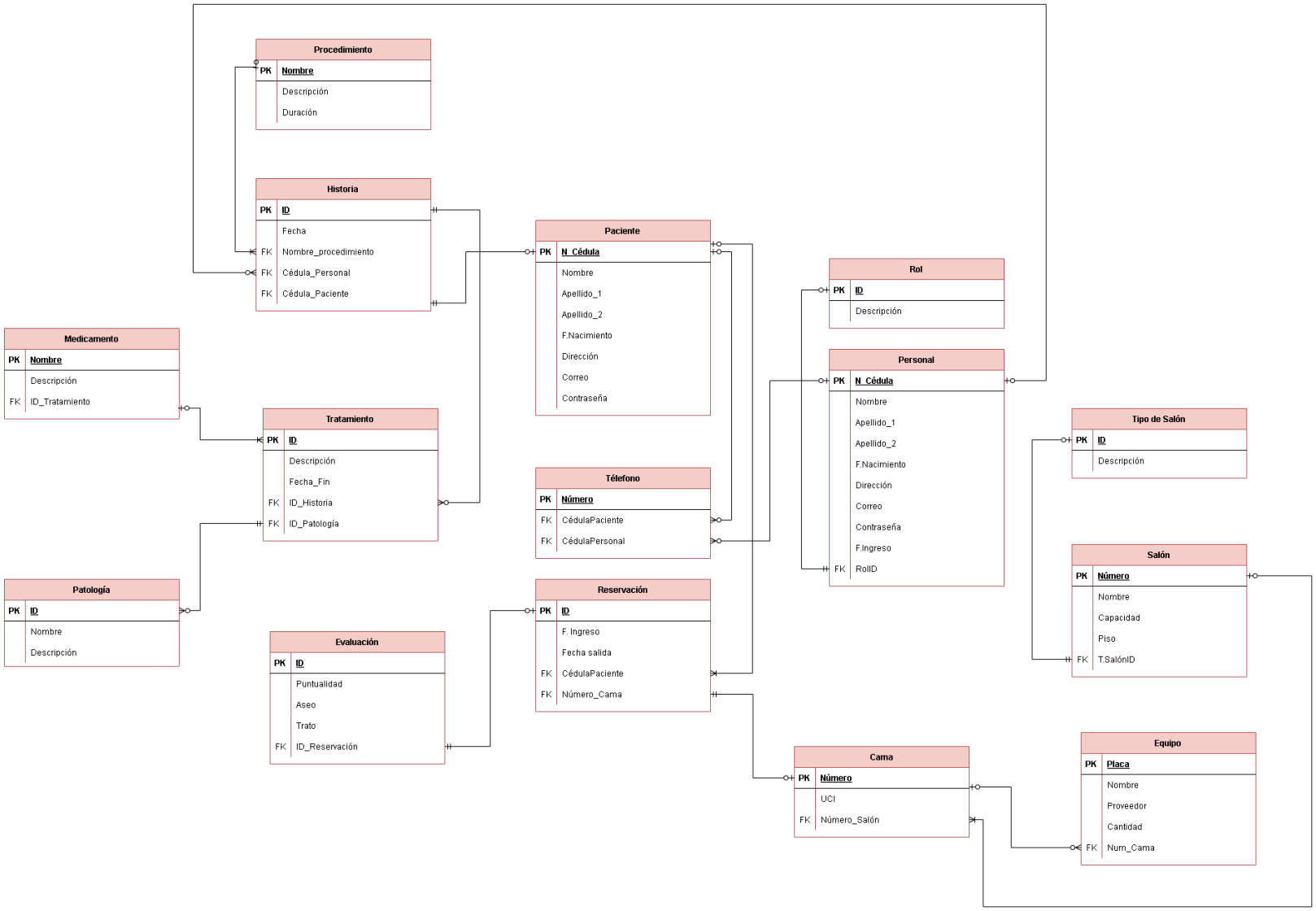
➢ Modelo elegido



Link:

https://app.diagrams.net/#G1bCvpHF-ivfYaXScJq3h8ej8xn2HGJcwU#%7B%22pageId%22%3A%224ByHGanSRbSz3iMyp9aS%22%7D

Modelo relacional



Link:
https://app.diagrams.net/#G1TCFWk1tW2tCO_G8ZIy0AWj37aZxduiDo#%7B%22pageId%22%3A%22CDxvmtDvXnWhhpZx3TVz%22%7D

Pasos seguidos:

1. **Mapeo de tipos de entidades fuertes**

- Las entidades fuertes en el diagrama conceptual son:
- Patología
- Tratamiento
- Historia
- Medicamento
- Paciente
- Evaluación
- Procedimiento
- Personal
- Reservación
- Salón
- Equipo
- Rol
- Tipo de Salón
- Cama

2. **Mapeo de tipos de entidades débiles**

No se utilizaron identidades débiles en el diagrama conceptual.

3. **Mapeo de tipos de asociaciones binarias 1:1**

- Tipo de Salón-Salón
- Personal-Rol
- Reservación-Cama
- Reservación-Evaluación

4. **Mapeo de tipos de asociaciones binarias 1:N**

Las relaciones de este tipo en el diagrama conceptual son:

- Procedimiento-Historia
- Paciente-Historia
- Paciente-Tratamiento
- Historia-Tratamiento
- Patología-Tratamiento
- Medicamento-Tratamiento
- Paciente-Reservación
- Personal-Historia
- Salón-Cama
- Cama-Equipo
- Personal-Telefóno
- Paciente-Teléfono

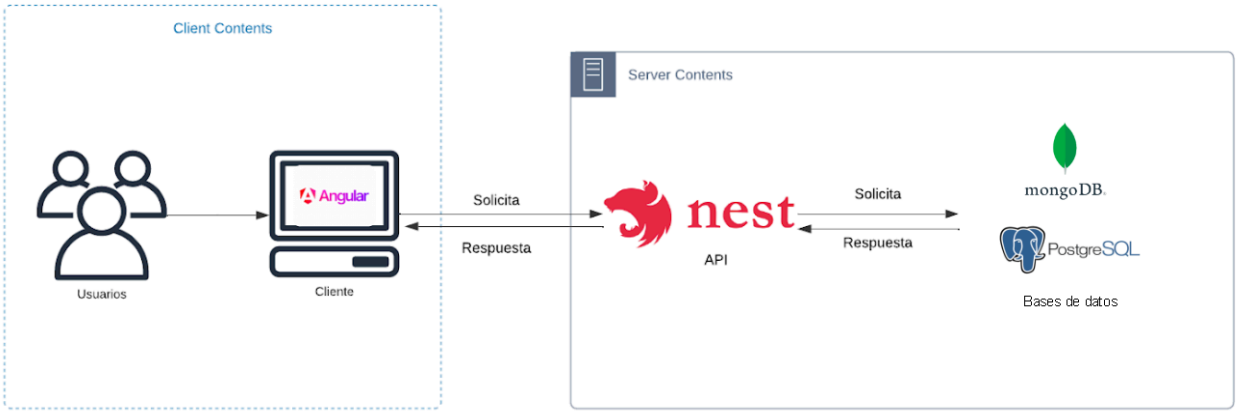
5. **Mapeo de tipos de asociaciones binarias N:M**

No se definieron relaciones de este tipo en el diagrama conceptual.

6. **Mapeo de atributos multivaluados**

Los atributos multivaluados utilizados en este diagrama conceptual son los teléfonos de los pacientes y del personal.

Diagrama de arquitectura



La arquitectura del sistema es cliente-servidor, el cliente está construido en Angular, el cual es un framework de diseño web para el *frontend*, este está basado en una arquitectura de componentes, donde cada componente puede ser reutilizado dependiendo de los requerimientos en las vistas. Por su parte el cliente está enlazado al servidor a través de una API construida con Nest JS, la cual a su vez está enlazada con la base de datos construida en PostgreSQL y MongoDB.

Descripción de las estructuras de datos desarrolladas

1. Tabla Pacientes

La tabla Pacientes almacena toda la información personal de los pacientes registrados en el hospital. Cada registro en esta tabla incluye el nombre, apellidos, cédula, número de teléfono, dirección, fecha de nacimiento y patologías del paciente. La cédula debe ser única para cada paciente, garantizando que no haya duplicados. Esta tabla es esencial para mantener un registro detallado y actualizado de todos los pacientes que reciben atención en el hospital.

2. Tabla Reservaciones

La tabla Reservaciones contiene información sobre las reservaciones de camas realizadas por los pacientes. Cada registro incluye el identificador del paciente que realiza la reservación, la fecha de ingreso, los procedimientos médicos que se llevarán a cabo y la fecha de salida, la cual se calcula automáticamente basándose en los días requeridos para los procedimientos. Esta tabla es fundamental para la gestión de camas y la planificación de recursos del hospital.

3. Tabla HistorialClinico

La tabla HistorialClinico almacena el historial médico de cada paciente, registrando los procedimientos realizados, la fecha de dichos procedimientos y los tratamientos prescritos. Esta tabla permite a los médicos acceder y actualizar el historial clínico de los pacientes, asegurando un seguimiento detallado de su salud y tratamientos a lo largo del tiempo.

4. Tabla Salones

La tabla Salones contiene información sobre los salones del hospital, incluyendo su número, nombre, capacidad en camas, tipo (por ejemplo, si es para mujeres, hombres o niños) y el piso en el que se ubica. Esta tabla es crucial para la organización y gestión de los espacios físicos dentro del hospital, permitiendo una asignación eficiente de los recursos.

5. Tabla EquipoMedico

La tabla EquipoMedico almacena detalles sobre el equipo médico disponible en el hospital, como su nombre, proveedor y cantidad disponible. Esta tabla es importante

para la gestión de inventarios y la planificación del uso de equipos médicos, asegurando que siempre haya suficiente equipo disponible para las necesidades del hospital.

6. Tabla Camas

La tabla Camas registra información sobre cada cama en el hospital, incluyendo su número, el equipo médico asociado, el salón en el que se encuentra, su estado (disponible u ocupada) y si es una cama de la unidad de cuidados intensivos (UCI). Esta tabla es esencial para la gestión de camas, permitiendo una asignación adecuada de camas a los pacientes y asegurando que la información sobre la disponibilidad de camas esté siempre actualizada.

7. Tabla Personal

La tabla Personal almacena información sobre el personal administrativo y médico del hospital, incluyendo su nombre, apellidos, cédula, número de teléfono, dirección, fecha de nacimiento, fecha de ingreso a la institución y tipo de personal (administrativo, doctor o enfermero). Esta tabla es clave para la gestión del personal, permitiendo un seguimiento detallado de la información de contacto y roles de cada miembro del personal.

8. Tabla Evaluaciones

La tabla Evaluaciones recoge las evaluaciones de los pacientes sobre los servicios recibidos en el hospital. Cada registro incluye el identificador del paciente que realiza la evaluación, el aspecto evaluado (como el aseo del hospital, el trato del personal o la puntualidad de las citas) y la nota recibida. Esta tabla permite al hospital recopilar feedback valioso de los pacientes y realizar análisis para identificar áreas de mejora en los servicios prestados.

Descripción de los Store Procedures y Triggers implementados

Stores procedures

1. AgregarPaciente

Este procedimiento almacenado permite añadir un nuevo paciente a la base de datos del hospital. Recibe como parámetros la información personal del paciente, como nombre, apellidos, cédula, teléfono, dirección, fecha de nacimiento y patologías. Al ejecutarse, inserta estos datos en la tabla Pacientes. Este procedimiento es fundamental para registrar a nuevos pacientes tanto en situaciones de emergencia como para cualquier persona que desee ser atendida en el hospital.

2. Crear Reservación

Este procedimiento se encarga de gestionar la creación de reservaciones para camas de hospital. Recibe como parámetros el ID del paciente, la fecha de ingreso y los procedimientos médicos que se realizarán. Calcula automáticamente la fecha de salida basada en los procedimientos médicos, sumando los días requeridos para la recuperación. Luego, inserta una nueva entrada en la tabla Reservaciones con la información proporcionada. Este procedimiento asegura que las reservas se gestionen de manera eficiente y precisa.

3. Modificar Historial Clínico

Este procedimiento permite a los médicos actualizar el historial clínico de un paciente. Recibe como parámetros el ID del historial, el procedimiento realizado, la fecha del procedimiento y el tratamiento prescrito. Actualiza los datos correspondientes en la tabla HistorialClinico, facilitando la actualización y mantenimiento de los registros médicos de los pacientes.

4. Gestionar Salones

Este procedimiento almacenado permite crear, modificar o eliminar registros de salones en el hospital. Recibe como parámetros una acción (crear, modificar o eliminar), el ID del salón (si aplica), el número del salón, el nombre del salón, la capacidad en camas, el tipo de salón (por ejemplo, si es para hombres, mujeres o niños) y el piso

donde se ubica. Dependiendo de la acción especificada, el procedimiento insertará, actualizará o eliminará los datos en la tabla Salones. Este procedimiento es crucial para la administración y organización de los espacios del hospital.

5. Reporte Áreas Mejora

Este procedimiento se encarga de generar un reporte de áreas de mejora basándose en las evaluaciones de los pacientes. Al ejecutarse, extrae las evaluaciones almacenadas en MongoDB, calcula las notas promedio para cada aspecto evaluado y almacena estos resultados en una tabla de reportes en PostgreSQL. Este procedimiento permite al hospital identificar y focalizar sus esfuerzos en las áreas que requieren mejoras, basándose en la retroalimentación de los pacientes.

Triggers

1. Trigger para validar camas disponibles antes de crear reservación

Este trigger se activa antes de insertar una nueva reservación en la tabla Reservaciones. Su función es validar que el hospital tenga camas disponibles en las fechas indicadas. Si no hay camas disponibles, genera una excepción y evita la inserción de la reservación. Este trigger es esencial para asegurar que el sistema no permita reservaciones que no se puedan cumplir por falta de disponibilidad de camas, mejorando la gestión de recursos del hospital.

2. Trigger para actualizar estado de cama al crear o eliminar reservación

Este trigger se activa después de insertar o eliminar una reservación en la tabla Reservaciones. Si se inserta una nueva reservación, el trigger actualiza el estado de la cama correspondiente a "OCUPADA". Si se elimina una reservación, actualiza el estado de la cama a "DISPONIBLE". Este trigger es fundamental para mantener actualizada la disponibilidad de camas en tiempo real, asegurando una gestión eficiente de los recursos hospitalarios y evitando conflictos de reservaciones.

Problemas conocidos

1. Integración Completa con MongoDB:

Aunque se ha implementado una conexión básica con MongoDB para almacenar las evaluaciones de los pacientes, aún existen desafíos en la integración completa y en la optimización de las consultas. La sincronización de datos entre PostgreSQL y MongoDB no está completamente automatizada, lo que puede llevar a inconsistencias de datos.

2. Optimización del Rendimiento:

Se han observado problemas de rendimiento cuando se manejan grandes volúmenes de datos, especialmente durante la generación de reportes y la gestión de reservaciones. La optimización de consultas SQL y el uso de índices adecuados aún están en proceso de mejora.

3. Gestión de Concurrencia:

Existen problemas en la gestión de concurrencia al realizar múltiples reservaciones simultáneamente. Aunque se han implementado triggers para validar la disponibilidad de camas, en situaciones de alta concurrencia, se han detectado posibles conflictos que aún necesitan resolverse.

Problemas encontrados

1. Problemas con Cors:

Los CORS en consideración de algunas medidas de seguridad que tienen algunos exploradores, por lo tanto, se debió acceder a la configuración del navegador respectivo para desactivarlos y evitar bloqueos inesperados. En el caso de google chrome, se accedió a chrome://flags/#block-insecure-private-network-requests para desactivar esta protección.

2. Problemas de Conexión a la Base de Datos:

Inicialmente, hubo dificultades para establecer conexiones estables con PostgreSQL y MongoDB. Este problema se soluciona configurando adecuadamente los parámetros de conexión y optimizando las configuraciones del servidor de bases de datos.

3. Inconsistencias en la Gestión de Reservaciones:

Se detectaron inconsistencias en la gestión de reservaciones, especialmente en la lógica de cálculo de fechas de salida. Esto se resolvió revisando y corrigiendo los procedimientos almacenados responsables del cálculo de fechas, asegurando que la lógica fuera precisa y consistente.

Conclusiones

Este proyecto tiene un enfoque integral y detallado en la gestión hospitalaria, abordando una amplia gama de funcionalidades para satisfacer las necesidades de un entorno médico complejo. Su objetivo general de desplegar una aplicación web totalmente funcional se refleja en la implementación de sistemas de registro y gestión tanto para pacientes como para el personal administrativo y médico.

El modelado de datos, desde los modelos conceptuales hasta el modelo relacional, muestra una comprensión clara de la estructura de la base de datos y su interacción con la aplicación. Además, la arquitectura del sistema demuestra una división clara de responsabilidades entre frontend y backend, lo que facilita la escalabilidad y el mantenimiento del sistema a largo plazo.

La implementación de procedimientos almacenados y triggers en la base de datos indica un enfoque avanzado para garantizar la integridad y consistencia de los datos, así como para automatizar ciertas acciones, como la actualización del estado de las camas al crear o eliminar reservaciones.

En resumen, HospiTEC presenta una estructura sólida y un enfoque profesional para abordar los desafíos de la gestión hospitalaria, utilizando tecnologías modernas y buenas prácticas de desarrollo. Su implementación promete ofrecer una interfaz intuitiva y efectiva para que tanto el personal médico como los pacientes interactúen con el sistema de manera eficiente.

Recomendaciones

- Una práctica recomendada es separar los servicios de los componentes, de manera que los servicios se encarguen de recopilar datos y gestionar la comunicación con el servidor.
- Si se desea dar un seguimiento y mantenimiento al proyecto, es importante utilizar un sistema de control de versiones para realizar un seguimiento de los cambios en el código para permitir la reversión del historial para evitar la pérdida de datos.
- Si se tiene una versión actualizada y estable, es importante recordar actualizar el código de las máquinas virtuales utilizadas para los despliegues con el fin de que sean implementadas correctamente.
- Si se desea ingresar nuevos datos o cambios dentro de la base de datos es importante tomar en cuenta la normalización para evitar la redundancia y mejorar la integridad de los datos. Estos se puede lograr siguiendo las pautas de normalización de bases de datos.
- Es importante monitorear el rendimiento de la base de datos para identificar cualquier problema de rendimiento y realizar ajustes según sea necesario.

Bibliografía

Crehana. (s.f.). *Angular desde cero: Fundamentos (Xavier Ramos)*. Crehana. https://www.crehana.com/cursos-online-front-end/angular-desde-cero-fundamentos/

García, D. L. (2019). *Tutorial Angular Material - Aprende a usarlo rápido*. Coding Potions - Blog de programación y desarrollo web. https://codingpotions.com/angular-material

Kinsta. (2022). *¿Qué es Nest.js?* Un Vistazo al Framework Ligero de JavaScript. https://kinsta.com/es/base-de-conocimiento/nestjs/

SQLite Tutorial (2024). *SQLite Tutorial*. SQLite Tutorial. https://www.sqlitetutorial.net/

WilliamDAssafMSFT. (2023). *Tutoriales del motor de base de datos - SQL Server*. *Microsoft Learn*. https://learn.microsoft.com/es-es/sql/relational-databases/database-engine-tutorials?view=sql-server-ver16