Aplicación para integrar y gestionar conjuntos de datos provenientes de aplicaciones de terapia interactivas e independientes del Instituto de Niños Ciegos y Sordos

Joan Alejandro Sandoval Escobar
Cristian David Riascos Ramírez

Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias
Departamento de Electrónica y Ciencias de la Computación
Santiago de Cali
2025

Aplicación para integrar y gestionar conjuntos de datos provenientes de aplicaciones de terapia interactivas e independientes del Instituto de Niños Ciegos y Sordos

Joan Alejandro Sandoval Escobar
Cristian David Riascos Ramírez

Trabajo de grado como requisito para obtener el título de Ingeniería en Sistemas y Computación

Director Juan Carlos Martínez Arias
Codirector Andrés Adolfo Navarro Newball

Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias
Departamento de Electrónica y Ciencias de la Computación
Santiago de Cali
2025

Santiago de Cali, enero 17 de 2025

Doctor
Gerardo Mauricio Sarria M.
Director Carrera Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana Cali

Cordial Saludo:

Por medio de la presente me permito informarle que he revisado el proyecto de grado **"Aplicación para integrar y gestionar conjuntos de datos provenientes de aplicaciones de terapia interactivas e independientes del Instituto de Niños Ciegos y Sordos"** de los estudiantes Joan Alejandro Sandoval Escobar (código 8956421) y Cristian David Riascos Ramírez (código 8955010), del cual somos Director y Codirector lo consideramos finalizado y listo para sustentación.

Atentamente,

Juan Carlos Martínez Arias
Firmado digitalmente por Juan Carlos Martínez Arias
Fecha: 2025.01.17 16:46:27 -05'00'

Juan Carlos Martínez Arias
Director de Trabajo de grado
Pontificia Universidad Javeriana Cali

Andrés Adolfo Navarro Newball
Firmado digitalmente por Andrés Adolfo Navarro Newball
Fecha: 2025.01.17 12:01:54 -05'00'

Dr. Andrés Adolfo Navarro Newball
Codirector de Trabajo de grado
Pontificia Universidad Javeriana Cali

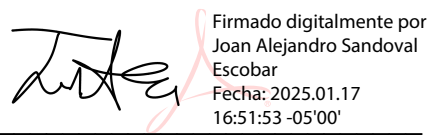
Santiago de Cali, enero 17 de 2025

Doctor
Gerardo Mauricio Sarria M.
Director Carrera Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana Cali

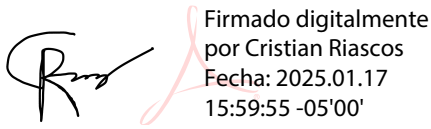
Cordial Saludo:

Me permito presentar a su consideración el proyecto de grado denominado: **“Aplicación para integrar y gestionar conjuntos de datos provenientes de aplicaciones de terapia interactivas e independientes del Instituto de Niños Ciegos y Sordos",** con el fin de cumplir con los requisitos exigidos por la Universidad para llevar a cabo el término de este y posteriormente optar por el título de Ingeniero de Sistemas y Computación.

Atentamente,



Joan Alejandro Sandoval Escobar
Código: 8956421



Cristian David Riascos Ramírez
Código: 8955010

CONTENIDO

Índice de figuras

1. ABSTRACT

Under the problem of inefficient management of statistical data corresponding to results from therapy applications at the Instituto para Niños Ciegos y Sordos, along with the challenge of quickly, effectively, and simply visualizing these results, this project addresses the need for an application to manage data generated by independent and interactive therapy applications at the Instituto para Niños Ciegos y Sordos del Valle del Cauca.

The solution is a web application that integrates different independent and interactive therapy applications into a single platform. It facilitates the administration of result data, allowing therapists to upload therapy session data and display patient progress through graphs. This application serves as a support tool for healthcare professionals to interpret results, translating into a more efficient and precise method to assist patients' conditions.

The methodology used is focused on resolving four critical components: first, designing a protocol to integrate data from various independent and interactive therapy applications and visualizing it graphically; second, designing a database to store the integrated data; third, developing and implementing the application using technologies like Next.js and FastAPI; and finally, conducting tests to ensure critical functionalities operate correctly. The project demonstrates that consolidating such therapies into a unified platform, using a protocol to simplify data management, provides an accessible solution for stakeholders needing straightforward access to information.

Key technical aspects include an ETL (Extract, Transform, Load) process for data integration, a relational PostgreSQL database for structured storage, user authentication with JWT tokens and two-factor authentication (2FA), and performance validation through unit and system testing. The application aims to enhance the efficiency of therapists and educators in tracking patient progress and making informed decisions.

2. RESUMEN

Bajo el problema de la poca eficiencia de la gestión de datos estadísticos correspondientes a los resultados obtenidos en aplicaciones de terapia del Instituto para Niños Ciegos y Sordos, además del problema para poder visualizar rápida, eficaz y simplemente los resultados obtenidos. Se analizaron las opciones y se culminó con la realización de este proyecto, que aborda la necesidad de un aplicativo que pueda dar administración a los datos que se generan a partir de aplicaciones de terapias interactivas e independientes del Instituto para Niños Ciegos y Sordos del Valle del Cauca.

La solución se basa en un aplicativo que permite la integración de diferentes aplicaciones de terapia interactivas e independientes en una sola aplicación, facilitando la administración de los datos de los resultados, permitiendo cargar los datos de las terapias realizadas, para así mostrar el progreso de los pacientes haciendo uso de gráficos. De esta forma, esta aplicación funciona a forma de apoyo para los profesionales de la salud encargados de leer e interpretar los resultados que al final se traduce en una forma más eficiente y precisa para ayudar a la condición de los pacientes.

La metodología se centró en la resolución de 4 partes críticas, inicialmente se planteó la creación de un protocolo que pudiera integrar los datos de los resultados provenientes de distintas aplicaciones de terapia interactivas e independientes, además de poder visualizar todos estos datos a forma de gráficos; después, se diseñó una base de datos con el objetivo de contener los datos obtenidos por el protocolo anterior; posteriormente se diseñó e implementó una aplicación, mediante el uso de tecnologías como NextJS y FastAPI; finalmente, se realizaron pruebas para garantizar que las áreas críticas funcionaban de forma correcta. Este proyecto muestra que se pueden juntar este tipo de terapias en una misma plataforma, usando un protocolo que haga más sencillo el manejo de los datos para las personas que necesiten acceder a ellos de una forma sencilla.

3. INTRODUCCIÓN

El Instituto para Niños Ciegos y Sordos del Valle del Cauca ha contado con el apoyo continuo de la Universidad Javeriana Cali, donde estudiantes de Ingeniería de Sistemas han desarrollado aplicaciones y terapias interactivas diseñadas para mejorar las dificultades motoras de los pacientes. Estas herramientas tecnológicas no solo ofrecen nuevas formas de terapia, sino que también generan datos valiosos sobre el desempeño y progreso de cada paciente, lo que ayuda aproximadamente a 150 pacientes que hacen parte de este instituto.

Sin embargo, un problema recurrente es que los resultados generados por estas aplicaciones permanecen almacenados localmente dentro de cada aplicación interactiva, lo que dificulta su acceso, análisis y comparación. Esto se debe a que para poder obtener estos datos se debería iniciar la aplicación, buscar al paciente dentro de esta aplicación, para así poder ver los resultados de este paciente, los cuales solo están almacenados en esa aplicación. Esta limitación no solo genera ineficiencias al momento de buscar información específica sobre un paciente, sino que también representa una pérdida de datos potencialmente útiles para evaluar la efectividad de las terapias interactivas.

En un contexto donde las tecnologías en la nube y el manejo avanzado de datos están cada vez más al alcance, surge la oportunidad de desarrollar una plataforma centralizada que organice y estructure esta información de manera eficiente. Un sistema de este tipo permitiría a los especialistas analizar patrones de progreso, identificar áreas de mejora en las terapias y evaluar su impacto en el desarrollo motor de los pacientes. Además, facilitaría el acceso remoto a la información, agilizando la toma de decisiones y optimizando el seguimiento a cada caso.

Por ello, resulta fundamental responder a la siguiente pregunta: ¿Qué características debe tener una solución tecnológica que reúna los resultados de las terapias interactivas y facilite su análisis por parte de los especialistas?

4. PLANTEAMIENTO DEL PROBLEMA

El Instituto para Niños Ciegos y Sordos del Valle del Cauca tiene como objetivo la rehabilitación de niños que han perdido parcial o totalmente, su capacidad de visión o audición. Para esto, tienen terapeutas profesionales que se encargan de la supervisión de los niños mientras desempeñan las aplicacaciones de terapia interactivas.

Los diversos tipos de aplicaciones que se emplean en las terapias son sistemas desarrollados previamente con la aprobación de los mismos terapeutas, con el fin de ayudar al niño en su rehabilitación. Es decir, un mismo niño puede estar realizando diversas aplicaciones de terapias interactivas e independientes y tener distintos tipos de progreso en cada una de ellas. De este modo, el terapeuta debe entender el progreso o regreso del niño con el fin de comprender y analizar esta información para llevar al niño a los mejores resultados posibles.

Actualmente, el instituto atiende diariamente alrededor de 150 niños, con un aproximado de 70 terapias distintas realizadas al día. Con un total de 51 profesionales de la salud de distintas áreas que requieren el acceso a la información y visualización de los resultados de las aplicaciones de terapias interactivas e independientes. Se tiene un seguimiento efectivo de cada niño trimestralmente, además de las intervenciones semanales.

No obstante, un obstáculo que impide la eficiencia de este proceso radica en la dificultad del acceso a los datos de los niños, este impedimento va desde la digitación de los datos de los niños, hasta la visualización del desempeño de los niños en aplicación de terapia interactiva. Es decir, la forma en que los datos son digitados por los profesionales es ineficiente debido a que actualmente estos datos son anotados en cuadernos y/o libretas para después ser pasados a Excel o al sistema.

Esto incluye dificultades en la digitación de los datos de los niños, así como en la visualización del desempeño de los niños en cada proyecto utilizado como terapia. Donse se puede entender que se requiere una tecnología que facilite la recopilación, gestión y visualización de los datos de los niños del Instituto de niños ciegos y sordos.

5. OBJETIVOS

5.1 OBJETIVO GENERAL

Desarrollar una aplicación que permita integrar y gestionar un conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes.

5.2 OBJETIVOS ESPECÍFICOS

- Elaborar un protocolo que permita la recepción de un conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes.
- Diseñar una base de datos que permita integrar un conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes, para el almacenamiento y acceso de una manera eficiente a información pertinente.
- Diseñar e implementar una aplicación que permita la visualización y gestión de los resultados del conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes.
- Validar el correcto funcionamiento de una aplicación que integra el conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes.

6. ALCANCE

Este proyecto desea contribuir al Instituto de Niños Ciegos y Sordos del Valle del Cauca mediante el desarrollo de un sistema que permita integrar y gestionar un conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes. El instituto ofrece diversas terapias que están implementadas para ser realizadas por los niños, convirtiéndolos en los usuarios de la aplicación, cada una diseñada para abordar necesidades específicas. Estas terapias generan resultados que son fundamentales para evaluar el progreso de los niños en su desarrollo cognitivo y motriz.

El sistema de gestión de resultados se centra en recibir, recopilar y almacenar los resultados de las actividades realizadas por los niños durante las terapias. Esta información es accesible para los terapeutas, quienes podrán utilizarla para realizar análisis rápidos del progreso de cada paciente.

El software tiene como enfoque el entendimiento, diseño e implementación de los resultados de al menos dos juegos que son utilizados como terapias. El primero, llamado Animal Maze, consiste en que el niño debe de ayudar a un animal a salir de un laberinto; el segundo, llamado Animal Rescue, se presentan animales cayendo y el niño trata de rescatar los que más pueda.

La implementación de este sistema tuvo como objetivo principal agilizar y mejorar el proceso de evaluación del desarrollo de los niños, permitiendo a los terapeutas acceder de manera rápida y precisa a los datos relevantes de cada paciente. Esto podrá facilitar la toma de decisiones informadas en cuanto a la planificación y ajuste de las terapias, así como la elaboración de informes detallados para los padres y otros profesionales involucrados en el cuidado de los niños.

En última instancia, la idea es que este sistema que permite integrar y gestionar un conjunto de datos provenientes de aplicaciones de terapia interactivas e independientes contribuya significativamente a la calidad de la atención brindada en el Instituto de Niños Ciegos y Sordos, mejorando el seguimiento del progreso individual de los pacientes y promoviendo su desarrollo integral. De esta forma, para garantizar el correcto funcionamiento de esta aplicación, se hará uso de pruebas unitarias.

7. ANTECEDENTES

Un trabajo similar fue realizado hace 3 años por , cuya problemática consistía en la realización de un sistema de administración de datos que ayudará a la economía de una región en Bali, Indonesia basados en productos de agricultura. Para el desarrollo del mismo, hubo una etapa de obtención de requisitos, diseño, implementación y testeo del sistema. Este sistema debía tener datos actualizados de productos agrícolas y contaba con acceso a distintos usuarios con diferentes permisos que incluían administradores, trabajadores y consumidores, dandoles así la opción de gestionar y actualizar los productos previamente mencionados de una forma eficiente; los consumidores por su parte tenían acceso a una forma de conocer precios y comprar productos con precios más transparentes, beneficiando así a todas las personas involucradas [1].

En el 2019 fue realizado un análisis de distintos tipos de aplicaciones móviles cuyo objetivo se centraba en ayudar al usuario a gestionar el uso de sus medicamentos diariamente, explicando los efectos y contraefectos de las medicinas que ingiere y monitoreando los resultados que tiene el paciente después de tomarlas[2]. El objetivo de este trabajo es similar, ya que se desea integrar y gestionar conjuntos de datos provenientes de aplicaciones de terapia interactivas.

Por otro lado, el sistema de salud danés es un ejemplo notable de cómo se pueden administrar datos provenientes de diversas aplicaciones y lugares mediante una estructura bien organizada y una digitalización avanzada. Dinamarca utiliza registros electrónicos de salud y estándares comunes de TI que permiten la comunicación eficiente entre hospitales, médicos generales, especialistas, laboratorios y servicios de atención domiciliaria. Esto asegura que los datos se puedan compartir y acceder fácilmente entre diferentes aplicaciones y lugares, esto se debe a que cada residente tiene un identificador único que permite el enlace preciso de registros individuales ya que la recolección de datos se realiza rutinariamente durante los contactos de salud y se almacenan en múltiples registros y bases de datos. Estos datos están disponibles para investigación clínica y proyectos, asegurando el manejo seguro y legal de los datos con respeto a la privacidad [3]. Con esto se puede evidenciar que ya existen formas de tener distintos sistemas o grandes cantidades de datos en distintos sistemas, aplicativos o base de datos de una manera conjunta, esto teniendo una estructura que se les pueda ofrecer a todos estos datos y también tener un identificador de los datos a que persona le pertenece.

8. MARCO TEÓRICO

Una de las etapas más importantes en la vida de cada individuo es su niñez, en esta etapa desarrollamos nuestras habilidades cognitivas, sociales, motoras, emocionales y también la autoestima. Para lograr el oportuno desarrollo de mencionadas habilidades, es de alta importancia la correcta educación y asimismo, estar rodeado de otros niños que complementen su desarrollo. No obstante, los niños que poseen pérdida total o parcial de sentidos como la audición o vista, no pueden o les es más difícil acceder a las instituciones educativas convencionales y desarrollar las habilidades referidas anteriormente.

El Instituto de Niños Ciegos y Sordos del Valle del Cauca, cuenta tanto como con una Clínica Visual y Auditiva que dispone con servicios de oftalmología, otorrinolaringología, optometría, terapia y óptica, para esto posee 120 profesionales de la salud que se encargan de realizar los exámenes. Además, posee un Centro de Rehabilitación con 18 aulas escolares donde existen diversos tipos de programas: aula consentidos, rehabilitación visual, rehabilitación auditiva, inclusión educativa, programa de familia, integración sensorial. Estos programas se encargan de desarrollar las capacidades sensoriales, sociales y educativas de los niños, que pueden matricularse en ciertos programas incluso teniendo meses de nacidos. Para lograr esto tienen 45 profesionales en áreas de educación: docentes especializados, fonoaudiólogos, terapeutas, psicólogos, trabajadores sociales, nutricionistas y coordinadores [4].

Cabe recalcar que un adulto que cuente con estas mismas condiciones bien sea de nacimiento o por algún accidente, tendrá que aprender a vivir de una forma distinta. De esta forma, el instituto también ayuda no solo a niños si no también a adultos, logrando un alcance mayor de beneficiados.

Por lo tanto el tener herramientas de ayuda que mejoren las terapias o la forma en cómo los terapeutas puedan brindar un mejor servicio a los niños como en el caso de SATReLO [5], es importante porque permite a los terapeutas personalizar las actividades terapéuticas de acuerdo a las necesidades específicas de cada paciente. Esto facilita la adaptación de los ejercicios a las habilidades y dificultades individuales de cada niño, lo que puede resultar en un proceso de rehabilitación más efectivo y personalizado, el uso de estas herramientas o estos juegos serios pueden ser importantes para mejorar la actitud y motivación de los niños en las terapias [6], pero no solamente los juegos serios, ya que los videojuegos comerciales basados en el movimiento se han encontrado efectivos como motivadores para realizar ejercicios de rehabilitación y ayudar a las personas con lesiones cerebrales a alcanzar sus objetivos terapéuticos, incluyendo mejoras en el equilibrio y rango de movimiento de las extremidades superiores [7].

8.1 BASE DE DATOS

Por eso es importante saber que es una base de datos, una base de datos es básicamente una colección organizada de información estructurada almacenada en un sistema informático. Sirve como un repositorio central donde los datos pueden ser almacenados, gestionados y accedidos de manera eficiente [8]. En la vida diaria se pueden ver distintos tipos de casos en los cuales se involucra el uso de base de datos, una de esas es en este proyecto, que requiere la implementación de una base de datos, pero existen distintos tipos de base de datos como las base de datos relacionales y las base de datos NoSQL, que son las bases de datos que pueden dar una solución al problema del proyecto.

8.1.1 BASE DE DATOS RELACIONAL Una base de datos relacional representa la base de datos como una colección de relaciones. Una relación se considera como una tabla de valores, cada fila en la tabla representa una colección de datos relacionados. Una fila representa un hecho que típicamente corresponde a una entidad o relación del mundo real. El nombre de la tabla y los nombres de las columnas se utilizan para ayudar a interpretar el significado de los valores en cada fila [9]. Una de las herramientas más populares que se utilizan para la creación de las bases de datos es ORACLE SQL.

8.1.1.1 ORACLE SQL

Oracle SQL es un lenguaje diseñado para interactuar con bases de datos relacionales, como Oracle Database. Sus fortalezas ofrecen beneficios a una amplia gama de usuarios, desde programadores de aplicaciones hasta administradores de bases de datos, gerentes y usuarios finales. Técnicamente hablando, SQL es un sublenguaje de datos cuyo propósito es proporcionar una interfaz a la base de datos relacional, donde todas las declaraciones SQL son instrucciones para la base de datos [9]. Esta herramienta es una de las que pueden generar una solución al proyecto, pero por otro lado está la base de datos NoSQL.

8.1.1.2 POSTGRESQL

PostgreSQL es un sistema de gestión de bases de datos relacional (RDBMS, por sus siglas en inglés) de código abierto, altamente robusto, escalable y extensible. Soporta SQL estándar y extensiones como CTEs, JSONB y consultas analíticas.

Maneja tipos de datos geométricos, arrays, JSON, XML, entre otros.

8.2 BASE DE DATOS NO SQL

Las bases de datos NoSQL (también conocidas como ”no solo SQL”) son bases de datos no tabulares que almacenan datos de manera diferente a las tablas relacionales. Estas bases de datos vienen en una variedad de tipos basados en su modelo de datos, siendo los principales tipos: documento, clave-valor, de columnas anchas y de gráficos. Proporcionan esquemas flexibles y escalan fácilmente con grandes cantidades de datos y cargas de usuarios altas [10]. También como el caso anterior una de las herramientas más populares para la creación de la base de datos NoSQL es MongoDB.

8.2.1 MONGODB MongoDB es una base de datos de documentos que se destaca por su escalabilidad y flexibilidad, además de ofrecer un avanzado modelo de consultas e indexación. En MongoDB, los datos se almacenan en documentos flexibles similares a JSON (formato clave-valor), lo que significa que los campos pueden variar entre documentos y la estructura de datos puede cambiar con el tiempo. Este enfoque permite una adaptabilidad dinámica a medida que evolucionan las necesidades de almacenamiento [10].

Es importante saber que el uso de las tecnologías es una herramienta más que se les brinda a los terapeutas, ya que los terapeutas perciben beneficios del uso de la tecnología en rehabilitación, como un mayor compromiso y participación de los pacientes en la terapia, una mejora en la prestación del servicio y el potencial de aumentar la finalización de ejercicios a través de una mayor motivación y compromiso. También informan beneficios personales, como una mayor productividad, ahorro de tiempo y la capacidad de ofrecer mejores métodos de rehabilitación adaptados a las necesidades individuales de los pacientes [11].

9. REQUISITOS

9.1 REQUISITOS FUNCIONALES

RF1: La aplicación debe contar con acceso por medio de credenciales y, debe entrar por medio de su nombre de usuario y correspondiente contraseña.

RF2: La aplicación debe tener un sistema de doble autenticación para cuando algún usuario desee ingresar. Esta doble autenticación será realizada por medio de códigos OTP que tendrán 6 dígitos enviados al correo del usuario. La doble autenticación se activará después de que el nombre de usuario y contraseña hayan sido previamente validados.

RF3: La aplicación debe permitir al usuario resetear su contraseña cuando se le hayan olvidado sus credenciales. Para realizar esto, el usuario debe de ingresar su correo electrónico, si existe entonces, le llegará un link para el formulario de reseteo de contraseña. El formulario le requerirá 2 campos: nueva contraseña y confirmar nueva contraseña.

RF4: La aplicación debe permitir al usuario restablecer su contraseña cuando este lo requiera. Para hacerlo, debe ingresar a la sección de configuración de perfil y llenar el formulario que le requerirá 2 campos: nueva contraseña y confirmar nueva contraseña.

RF5: La aplicación debe permitir al usuario visualizar su nombre, nombre de usuario, correo electrónico y rol en la sección de configuración de usuario.

RF6: La aplicación deberá contar con 3 roles para los usuarios: terapeuta, docente y administrador.

RF7: La pestaña de manejo de usuarios solo debe ser accesible para usuarios de tipo administrador y restringir el acceso para usuarios de tipo terapeuta o docente.

RF8: En la pestaña de manejo de usuarios, debe de haber un listado de todos los usuarios registrados en el sistema. Permitiendo editar o borrar usuarios de dicho listado. En el caso de editar usuarios, solo será posible modificar el nombre, correo electrónico o rol.

RF9: El sistema debe garantizar que los usuarios solo puedan editar su propia contraseña y no la de otros usuarios. Independientemente si son administradores o no.

RF10: La aplicación debe permitir añadir usuarios nuevos en la sección de manejo de usuarios. Los datos requeridos para el registro son el nombre, nombre de usuario, contraseña, correo electrónico y rol.

RF11: La aplicación debe garantizar que no existan ID de usuario, nombres de usuarios o correos electrónicos duplicados al crear o modificar usuarios.

RF12: La aplicación debe de permitir asignar pacientes existentes a un usuario existente. Para esto, se debe de poder elegir el paciente existente por medio de su ID.

RF13: La aplicación debe de permitir a un usuario agregar un paciente nuevo. Los datos requeridos para el registro son: ID, nombre, edad, nivel escolar, diagnóstico, género y usuario al que dicho paciente será asignado.

RF14: La aplicación debe asegurarse de que no sea posible que existan ID de pacientes duplicados.

RF15: La aplicación debe de permitir a un usuario visualizar a forma de listado a todos los pacientes que tiene asignados. Dentro de este listado, el usuario puede deasignar pacientes.

RF16: En el apartado de visualización de datos, la aplicación debe permitir al terapeuta filtrar cada terapia con su correspondiente nivel para obtener la visualización de los resultados que obtuvo el paciente correspondientes a la terapia y nivel seleccionado por el terapeuta. Esto se hará por medio de una barra desplegable para seleccionar la terapia y otra barra desplegable para seleccionar el nivel.

RF17: En el momento en que un usuario desee ver los resultados de un paciente y active la barra desplegable para seleccionar la terapia, la aplicación debe contar con íconos únicos para distinguir cada terapia.

RF18: La aplicación debe incluir una sinopsis pequeña de cada terapia que incluya la descripción de la terapia seleccionada.

RF19: La aplicación debe permitir al usuario visualizar los resultados de las terapias de cada paciente asignado en forma de gráficos.

RF20: Animal Maze debe de tener 2 gráficos: tiempo por número de partidas y puntos por número de partidas.

RF21: Animal Rescue debe de contener 3 gráficos: capturados por lado izquierdo por número de partidas, capturados por lado derecho por número de partidas y animales diferentes al jaguar capturados por número de partidas.

RF22: En la sección de Datos, la aplicación debe permitir al usuario visualizar los datos del paciente seleccionado. Estos datos son: ID, nombre, edad, diagnóstico, nivel escolar y género.

RF23: Se debe desarrollar un ETL que tenga la capacidad de extraer, transformar y cargar los datos de las terapias en la base de datos.

RF24: La aplicación debe ser capaz de extraer los datos de las terapias desde un bucket S3 de AWS. Los datos de las terapias estarán en formato JSON y deben de ser convertidos en data frames.

RF25: La aplicación debe de poder añadir a los data frames extraídos el ID del jugador y la fecha en que fue jugada la partida, esto se logra extrayendo estos datos del nombre de cada archivo.

RF25: La aplicación debe de transformar los dataframes de Animal Rescue y Animal Maze para que contengan exactamente las variables de la tabla variableResult.

RF26: La aplicación debe cargar los datos transformados de las terapias en la base de datos, garantizando la integridad de los datos y evitando duplicados

RF27: La aplicación debe de contar con un botón para ejecutar manualmente el ETL para obtener los resultados de las terapias.

RF28: La aplicación debe de garantizar que el nombre de un usuario y paciente deben de contener entre 1 y 50 caracteres antes de registrar o editar un usuario.

RF29: La aplicación debe garantizar que el email de un usuario tenga un formato válido antes de registrar o añadir un editar nuevo.

RF30: La aplicación debe garantizar que el nombre de usuario debe de contener entre 4 y 13 caracteres. Además, solo puede incluir letras mayúsculas o minúsculas, números y guiones bajos.

RF31: La aplicación debe garantizar que la contraseña que un usuario elija tenga entre 8 y 20 caracteres.

RF32: La aplicación debe garantizar que la edad de los pacientes se encuentre entre 0 y 18 años.

RF33: La aplicación debe garantizar que el diagnóstico de un paciente se encuentre entre 1 y 100 caracteres.

9.2 REQUERIMIENTOS NO FUNCIONALES

RNF1: La aplicación debe manejar un aumento de resultados de aplicaciones terapias interactivas sin afectar el desempeño actual.

RNF2: La aplicación debe tener una interfaz de usuario intuitiva y accesible, con un tiempo de aprendizaje de menos de 1 hora para nuevos usuarios.

RNF3: La aplicación debe de usar tokens de autorización con un tiempo limitado de 2 horas para evitar problemas de seguridad.

RNF4: La aplicación debe de usar tokens para el reseteo de contraseña con un tiempo limitado de 7 minutos.

RNF5: La aplicación debe de almacenar datos sensibles (contraseñas) de forma encriptada en la base de datos.

RNF6: La aplicación debe utilizar acceso asíncrono a la base de datos para garantizar un manejo eficiente de múltiples solicitudes concurrentes.

RNF7: La aplicación debe de limitar el tiempo a 3 minutos de duración para los códigos OTP en la doble autenticaión.

RNF8: La aplicación debe de usar variables de entorno para credenciales de correo electrónico y llaves secretas.

RNF9: La aplicación debe de poder manejar hasta 30 usuarios en paralelo sin que se vea afectado su rendimiento.

RNF10: La aplicación debe tener un tiempo de respuesta inferior a 2 segundos para la mayoría de las solicitudes de usuario.

RFN11: El ETL debe de tener un tiempo de respuesta inferior a 2 minutos para 100 partidas almacenadas.

RFN12: La base de datos debe ser capaz de ejecutar consultas complejas en menos de 3 segundos.

9.3 RESTRICCIONES

R1: La aplicación debe ser desarrollada en NextJS y Fastapi
R2: El ETL debe de ser desarrollado en Python.
R3: La base de datos debe de estar implementada en PostgreSQL.
R4: La aplicación debe de usar Redis para almacenar la información en memoria caché del código OTP y correo asociado.
R5: La aplicación debe de usar tokens JWT.

10. ELABORACIÓN DEL PROTOCOLO PARA RECEPCIÓN DE UN CONJUNTO DE DATOS PROVENIENTES DE APLICACIONES DE TERAPIA INTERACTIVAS E INDEPENDIENTES

Como fue específicado en la sección 5, serán consideradas 2 aplicaciones de terapia interactivas e independientes: Animal Maze y Animal Rescue. Retomando en qué consiste cada uno con un poco más de detalle; Animal Maze consiste en un laberinto con una posición de inicio y solo una posible salida, el personaje del paciente es un animal. El paciente debe mover el dispositivo en el que se juega que es idealmente una tablet con el sistema operativo de Android, con los movimientos del dispositivo mueve al animal en la dirección acorde, es decir, si el dispositivo es inclinado hacia el frente entonces el animal se mueve hacia arriba, de esta forma se guía al animal hasta llegar a la salida. Mientras el paciente mueve al animal, en caso de chocarse emite sonidos para indicarle al jugador que está colisionando con una estructura.

Por otra parte; Animal Rescue radica en el uso del Kinect de Xbox. La pantalla estará dividida en la sección izquierda y derecha, donde caen distintos tipos de animales y el paciente debe atrapar jaguares con la mano acorde a la sección, es decir, si un jaguar cae por la sección derecha, entonces debe atraparlo con la mano derecha. Adicionalmente, no todos los animales deben ser atrapados, solo uno jaguares, los animales emiten sonidos al ser atrapados.

Para extraer los datos de los resultados, se consideró el uso de ETL, que consiste en un proceso de extraer, transformar y cargar los datos. Este método consiste en tomar datos de múltiples fuentes para cargarlos en un repositorio central [12]. Consiste en 3 pasos:

1. Extraer los datos necesarios de las fuentes.
2. Transformar los datos al formato necesario.
3. Cargar los datos en la base de datos objetivo.

10.1 DISEÑO DEL ETL

10.1.1 Campos a considerar Ambas aplicaciones de terapia interactivas e independientes almacenan distintos tipos de resultados, uno con más elementos que otro. Animal Maze considera únicamente el tiempo que le toma al paciente llegar a la salida del laberinto y una medida de puntos que se determina por las colisiones que tenga el jugador y el tiempo. Animal Rescue considera una cantidad de valores más complejas:

- CIMC: Captura izquierda mano correcta
- TSI: Total spawned izquierda
- CIMI: Captura izquierda mano incorrecta
- PI: Perdidos izquierda
- CDMC: Captura derecha mano correcta
- TSD: Total spawned derecha
- CDMI: Captura derecha mano incorrecta
- PD: Perdidos derecha
- ADJC: Animales distintos al jaguar capturados

10.1.2 Tipos de datos de los campos Para el laberinto, como se explicó previamente se va a tomar tanto el tiempo como los puntos de los jugadores. Dado que esta aplicación ya tiene establecidos sus tipos de datos, se consideró pertinente usar los mismos tipos de datos. Es decir, el tiempo se tomará como un número flotante y los puntos como un número entero.

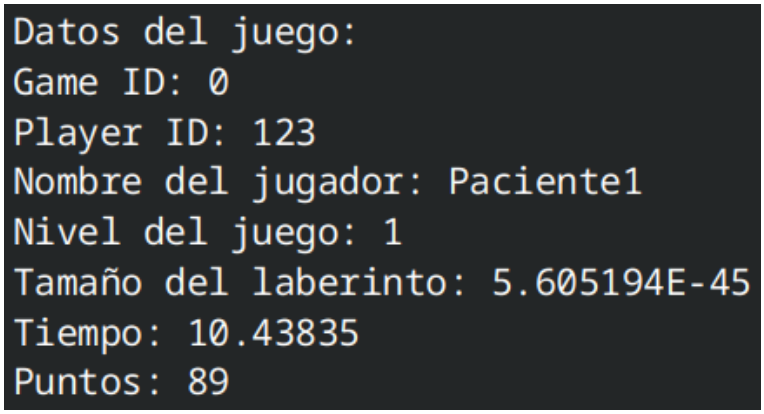


Figura 1. Ejemplo de resultados de una partida en Animal Maze

De forma similar, para Animal Rescue se tomarán los tipos de datos que fueron usados en el desarrollo de dicha aplicación. Ver Sección 9.1.1. En este caso, se manejarán datos de tipo entero flotante.

10.1.3 Valores En esta sección del diseño se toma a consideración el rango de valores posible para cada métrica de resultado en las aplicaciones de terapia interactivas e independientes. Para Animal Maze, el tiempo máximo estará definido entre 0.00 a N, debido a que no existe un tiempo límite; el número de puntos depende del nivel de dificultad, donde en su mayor dificultad el usuario inicia con 300 puntos y comienza a bajar mientras pasa el tiempo o el jugador colisione, es decir, el número de puntos estará entre 0 y 300.

Animal Rescue, por su parte, considera rangos que van desde 0.00 e incrementan dependiendo del desempeño del paciente en la partida. Es decir, los rangos de las 9 variables van desde 0.00 hasta N.

10.1.4 Frecuencia Se determinó que el ETL no funcionará de forma automática apenas una nueva partida haya sido jugada, en cambio, funcionará de forma manual que se activará como un botón por parte de los usuarios.

10.1.5 Herramienta Existen diversas herramientas para la implementación del ETL cada una teniendo sus puntos fuertes, por lo que es necesario elegir acorde a las necesidades y especificaciones del ETL que se va a desarrollar. Es necesario primeramente aclarar cómo se encuentran los datos para poder llegar a una conclusión, esto se abordará por cada aplicación de terapia interactiva e independiente de forma individual, así como considerando posibles aplicaciones a futuro.

A pesar de que existen herramientas que incluyen GUI, fue decidido usar Python gracias a que tiene la capacidad de personalización mayor respecto al ETL, su facilidad de aprendizaje y uso, lo convierte en una de las herramientas más accesibles para desarrolladores de todos los niveles. Python se caracteriza por una sintaxis clara y legible, incluso su gran cantidad de librerías y paquetes especializados proporciona soluciones para casi cualquier necesidad en el desarrollo. También, en caso de que el futuro se realicen demasiadas consultas, haciendo que Pandas sea ineficiente, sería posible integrar PySpark para manejar grandes volúmenes de datos.

10.2 IMPLEMENTACIÓN DEL ETL

10.2.1 Extracción Este proceso consiste en obtener los resultados de las aplicaciones de terapia de tal forma como son exportados por los juegos. Primero se debe recalcar que para ambos juegos, los archivos son subidos a un Bucket S3 de AWS. Dentro del proceso de extracción, se configura el acceso al bucket y se extraen los archivos que se encuentren para que posteriormente sean enviados al proceso de transformación.

Explicando más en detalle, para el caso de Animal Maze los archivos que

guardan la información de cada partida están en una carpeta llamada games y los archivos con los resultados de cada partida son guardados en formato binario. Fue necesario modificar el código para permitir el envío de estos archivos en formato JSON antes de ser serializados al Bucket S3. Es decir, apenas una partida de complete los datos son envíados al Bucket S3.

10.2.2 Transformación En esta etapa se toman los datos extraídos y se transforman al formato necesario para poder ser usados en la base de datos. Ver Figura 2.

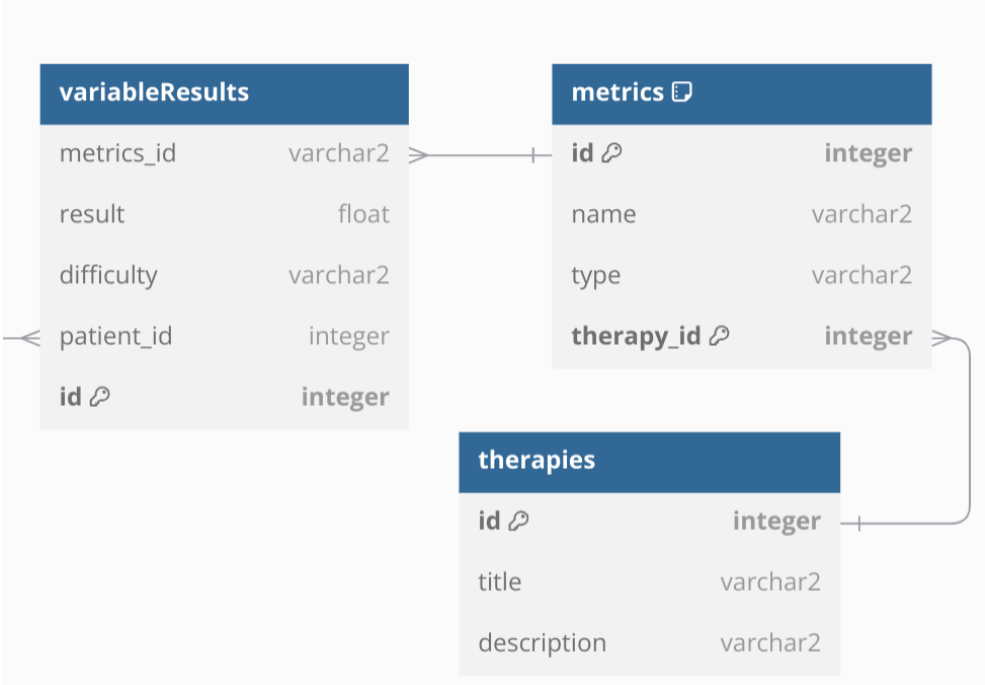


Figura 2. Tablas usadas en la transformación

En este caso, la tabla en donde se almacenan los resultados específicamente es variableResults, no obstante, primero se debe de entender las tablas metrics y therapies.

Therapies contiene el nombre y descripción de cada aplicación interactiva, además tiene un ID que además de ser su llave primaria, es un valor único. Esta tabla permite distinguir cada aplicación individualmente, actualmente que solo cuenta con dos terapias distintas pero será posible añadir más a futuro usando esta tabla.

Metrics contiene las métricas en que se mide cada juego (ver 8.1.1). Cada métrica contiene un ID, nombre, tipo de dato y a qué aplicación interactiva le corresponde respectivamente. Por ejemplo, una métrica que va en esta tabla es el tiempo que se mide en Animal Maze. Esta tabla permite organizar la estructura de la toma de los resultados e igualmente puede ser usada cuando se agreguen más aplicaciones.

Finalmente, variableResults contiene la información de los resultados de las partidas de las terapias. En el proceso de transformación, se convierten los datos obtenidos de la extracción para que puedan ser posteriormente subidos en esta tabla. Específicamente almacena el ID de la métrica a la que le corresponde, el resultado, la dificultad, el ID del paciente que jugó dicha partida y un ID para diferenciar entre cada resultado y serían estos los valores a los que los datos se transforman.

10.2.3 Carga Esta es la última parte del ETL, que consiste en tomar los datos transformados para ser finalmente subidos a la base de datos. En esta última fase se creó la sesión para conectarse a la base de datos y se hizo la consulta para subir los datos transformados.

11. DISEÑO DE LA BASE DE DATOS PARA INTEGRAR EL CONJUNTO DE DATOS PROVENIENTES DE APLICACIONES DE TERAPIA INTERACTIVAS E INDEPENDIENTES

Al inicio de la planeación de la base de datos se pensó en utilizar una base de datos no relacional, ya que sería mucho más fácil el manejo de los resultados de las aplicaciones de terapias interactivas e independientes debido a que los resultados son guardados como .json, con esto se dio a un primer modelo:

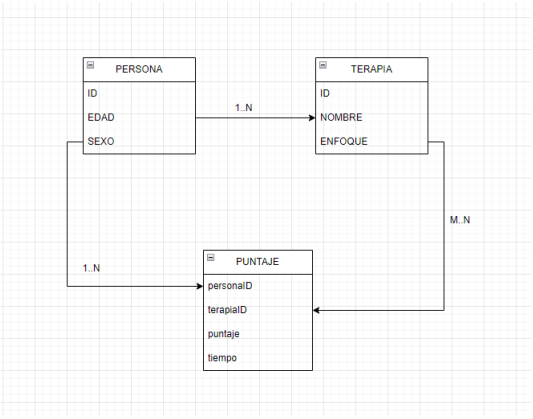


Figura 3. Primer modelo de la base de datos

Con distintos análisis sobre el primer Modelo y petición del Instituto para Niños Ciegos y Sordos del Valle del Cauca sobre la consistencia en los datos de los niños, se optó por el uso de una base de datos relacional, ya que este tipo de base de datos cumplen con los principios ACID, dando confiabilidad e integridad a los datos.

El primer problema encontrado al utilizar una base de datos relacional es el guardado de los resultados, ya que al tener los resultados en formato json se debe de pensar una manera en que los resultados coincidan con el paciente que los hizo. Con esto se llegó al siguiente modelo relacional que tiene en cuenta las diferentes variables dependiendo el tipo de terapia que se este empleando y su respectivo resultado. Ver Anexo 1

Figura 4. Segunda versión de la base de datos

Como se puede apreciar en el diagrama anterior Fig. 4, las tablas de resultadoVariable, nombreMetrica y terapia, son las encargadas de la administración de los datos de resultados de las terapias. De la siguiente manera esta estructurado la base de datos:

- **Users:** Es el encargado de guardar los usuarios que están disponibles para hacer uso de la aplicación, teniendo en cuenta el rol del usuario para así poder mostrar lo necesario a este usuario. Respecto a los roles de usuario, por ahora existen 3: administrador, terapeuta y docente; el administrador está encargado de la supervisión de los usuarios (terapeuta y docente); el terapeuta es aquel que ayuda y supervisa a los pacientes mientras realizan los ejercicios; el docente es aquella persona que está en el salón de clase junto a los niños.
- **Paciente:** Esta tabla es encargada de guardar los datos importantes de los niños.
- **Terapias:** Esta tabla es la encargada de guardar nombre y una descripción de las terapias disponibles.
- **nombreMetrica:** Esta tabla es la encargada de guardar los nombres de las métricas o variables que tiene cada terapia, esta tabla nos asegura que cierta métrica proviene de una terapia en específico.
- **ResultadoVaribale:** Esta tabla es la encargada de guardar los resultados de cada métrica (variable que posee la terapia) que tenga la terapia, teniendo en cuenta el idPaciente que lo realizo y el idMetrica que nos indica que métrica tiene cierta terapia.

No obstante, parte del personal del Instituto para Niños Ciegos y Sordos del Valle del Cauca informó como retroalimentación que no se debe de considerar todos los atributos de la tabla de paciente de la figura anterior Fig. 4. En cambio, solo se debía considerar el nombre, edad, nivel escolar, diagnóstico y sexo. Por lo que la versión final del modelo incluye estos cambios además de que se cambió el idioma a inglés.

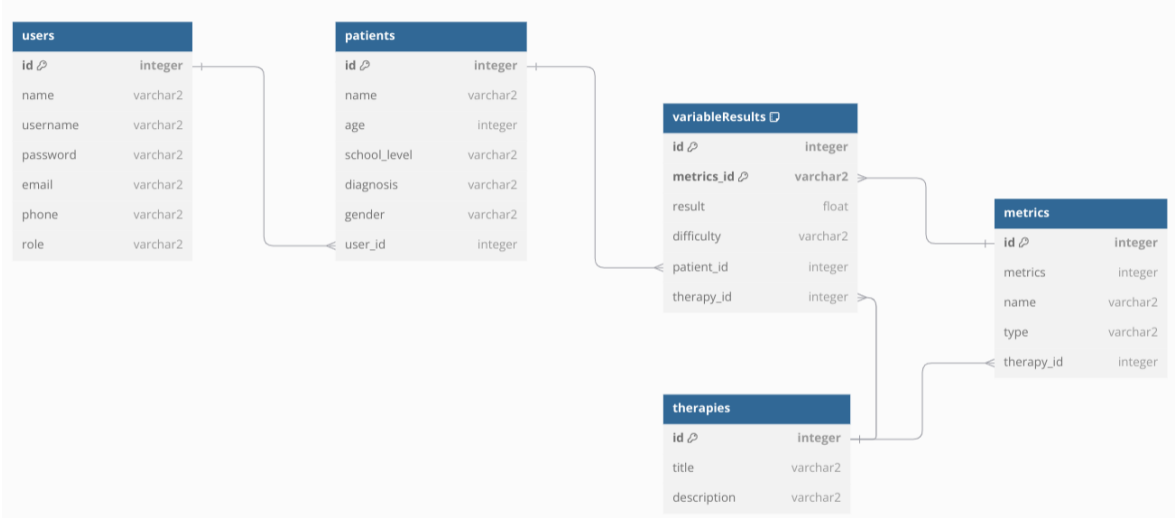


Figura 5. Versión final de la base de datos

11.1 PROTOCOLO PARA AÑADIR NUEVAS APLICACIONES DE TERAPIA INTERACTIVAS E INDEPENTIENTES

La base de datos y el ETL fueron diseñados teniendo en cuenta la alta posibilidad de la adición de nuevas aplicaciones de terapias interactivas e independientes, de tal forma en que este proceso no fuera complejo. Se deben de seguir los siguientes pasos cuando sea requerido añadir otra aplicación de terapia:

1. Añadir manualmente una entrada en la tabla de Therapies que consiste en un ID, título la terapia y una pequeña descripción de la aplicación de terapia. Considerar que los ID 1 y 2 ya están tomados.

2. Añadir manualmente cada tipo de métrica en la tabla Metrics que consiste en el ID de la terapia, nombre de la métrica, tipo de dato la métrica y el ID de terapia

a la que pertenece. EL ID de la terapia inicia en 1 por cada terapia. Por ejemplo, si está una aplicación de terapia interactiva e independiente con 3 tipos de métricas, entonces las métricas para esta terapia estarían numeradas entre 1 y 3 sin ningún orden particular.

3. Este paso debe de realizarse en el desarrollo de la aplicación de terapia interactiva e independiente. Se debe de implementar una forma para enviar los datos a un bucket S3 de AWS. La nomenclatura del archivo que contenga los resultados de un paciente debe de ser: ID del paciente + _ fecha en que se jugó la partida + '_ hora en que se jugó la partida.

4. Se debe de modificar el código transform.py del ETL, en este caso, se debe de modificar dependiendo de cómo esté estructurado el archivo que contiene los resultados.

12. IMPLEMENTACIÓN DE LA APLICACIÓN PARA VISUALIZAR Y GESTIONAR LOS RESULTADOS DEL CONJUNTO DE DATOS PROVENIENTES DE APLICACIONES DE TERAPIA INTERACTIVAS E INDEPENDIENTES

12.1 INSTRUMENTOS DE DISEÑO DEL FRONT-END

La fase del desarrollo implica el uso de tecnologías esenciales para crear un front-end eficaz, avanzado e ideal. En caso, los usuarios utilizan las primeras: Next.js, Formik con Yup, Material UI y Axios, cada una de las herramientas juega su papel clave en el enfoque distinto del diseño y la función.

12.1.1 Next.js Next está hecho en React Framework para optimizar tanto el rendimiento como la experiencia del usuario, puede realizar renderizado del servidor y generación de sitios estáticos. Sus ventajas son:

- Mejorar significativamente el rendimiento de las páginas y la optimización para motores de búsqueda (SEO) al pre-renderizar las vistas antes de enviarlas al cliente.
- Garantizar que solo se cargue el código necesario para cada vista, optimizando la experiencia del usuario.
- Soporte nativo para CSS y Sass, facilitando la integración de estilos en los componentes.

El motivo del uso de esta herramienta es debido a la facilidad de entender el flujo del aplicativo, también la buena documentación a conceptos importantes hacen que se entienda cómo funciona.

12.1.2 Formik y Yup Formik está diseñado para gestionar formularios con validaciones complejas de manera sencilla. Formik admite validaciones tanto sincrónicas como asincrónicas a nivel de formulario y de campo. Sus ventajas son:

- Facilitar la creación y mantenimiento de formularios complejos, reduciendo significativamente el código necesario.
- Yup proporciona un sistema flexible para definir validaciones complejas de manera declarativa y eficiente.

- Ambas herramientas trabajan en conjunto para así crear formularios robustos con validaciones y manejo de datos fácilmente.

Esta librería es utilizada por la gran facilidad del manejo de formularios, ya que nos permite saber de una manera rápida los valores iniciales, las acciones que pueden ser ejecutadas al presionar un botón y al integrarlo con Yup, nos proporciona un manejo rápido de validaciones de los datos que se estén integrando en el formulario.

12.1.3 Material UI Material UI es una biblioteca de componentes de React de código abierto que implementa el diseño Material de Google. Incluye una amplia colección de componentes preconstruidos listos para ser utilizados en producción desde el primer momento y ofrece un conjunto de opciones de personalización que facilitan la implementación de un sistema de diseño propio sobre sus componentes. Sus ventajas son:

- Proporciona elementos listos para usar como botones, tablas y formularios, acelerando el desarrollo y proporcionando mejor entendimiento de la interfaz del aplicativo.
- Permite adaptar fácilmente los estilos y funcionalidades de los componentes para cumplir con los requisitos del proyecto.

El motivo del uso de esta librería es debido a su buena documentacion, dando un buen entendimiento de los componentes que se quisieron utilizar en el proyecto, para así poder dar una mejor interacción al usuario.

12.1.4 Axios Axios es una biblioteca basada en JavaScript para realizar solicitudes HTTP. Es ampliamente utilizada en proyectos que necesitan interactuar con APIs externas. Sus ventajas son:

- Su sintaxis es más clara y concisa que Fetch, facilitando la interacción con APIs.
- Nos da un manejo automático en la conversión de datos JSON, reduciendo el esfuerzo necesario en la manipulación de respuestas.
- Mejor manejo errores y manipulación de datos con flexibilidad para mostrar los errores que se muestran en el Back-end.

Axios fue seleccionado por su flexibilidad y funcionalidad avanzada, ofreciendo un mejor manejo de errores y configuración en comparación con Fetch.

12.2 INSTRUMENTOS DE DISEÑO DEL BACK-END

Para el desarrollo del back-end se deben de considerar herramientas que sean modernas, eficientes y seguras. Para esto, se optó por el uso de los siguientes frameworks y librerías: FastApi, JWT, OAUTH2, SQLAlchemy, Pydantic, Redis, SMTP Server y Secrets.

12.2.1 FastApi Es un microframework de Python empleado para la creación de interfaces de programación de aplicaciones usadas en el campo del desarrollo web. Caracterizado por ser moderno y de alto desempeño [13]. Sus ventajas son:

- Proporcionar uno de los rendimientos más altos en comparación a otros frameworks de Python.
- Permitir el uso de código concurrente y asíncrono de forma nativa .
- Ofrecer una documentación automática del código, reduciendo el tiempo de desarrolo y haciendo que explicar el código sea más sencillo.

El motivo por la selección de este framework es gracias a que una de las características que requiere la aplicación, es el uso de funciones asíncronas debido a la necesidad de operaciones de entrada y salidad. Flask es otro framework enfocado al desarrollo web y el que usualmente los desarrolladores web consideran usar, no obstante, Flask está diseñado para funciones síncronas y aunque sí es posible usar funciones asíncronas con Flask, su rendimiento es mucho menor en comparación a FastApi [14].

12.2.2 JWT Los JSON Web Token son un método compacto para transmitir información de forma segura como un objeto JSON. Sus ventajas son:

- Más compacto que otros tipos de token como SWT o SAML. Haciendo que su tamaño sea menor en comparación a los mencionados anteriormente.
- Permitir su uso a escala de internet, haciendo que sea más fácil su procesamiento en los dispositivos de los usuarios.
- Ofrecer una seguridad mayor que SWT y SAML al emplear llaves públicas y privadas que pueden ser firmadas de forma simple en el formato de un JSON [15].

El propósito del uso de esta herramienta consiste en evitar posibles problemas de seguridad dado que se maneja información sensible de personas reales. El manejo de esta tecnología se basa principalmente para motivos de autorización, para garantizar que el usuario esté validado para acceder a las diferentes funciones de la aplicación.

12.2.3 OAUTH2 Es un framework enfocado a la autorización, utilizado para permitir el acceso limitado a funcionalidades de un sistema sin compartir las credenciales del usuario por medio de token [16]. Sus ventajas son:

- Ofrecer autorización sin necesidad de compartir credenciales del usuario mejorando así la seguridad de la aplicación.
- Control de autorización limitado en cuestiones de tiempo gracias a que la duración de los token es fijo y es establecida durante el desarrollo.

Usado en conjunto con los JWT y como se mencionó anteriormente, el motivo del uso de esta tecnología es por objetivos de seguridad. OAUTH2 se usa para definir el flujo de la autorización mientras que los JWT establecen una forma segura de transmitir los datos.

12.2.4 SQLAlchemy Es un ORM para realizar consultas y manipulaciones de bases de datos relacionales en Python [17].

Esta herramienta es necesaria debido a que se requería realizar consultas, añadir y/o modificar datos de las tablas en la base de datos que contiene la información pertinente desde credenciales para la autenticación de los usuarios, hasta la información individual de resultados que los pacientes tuvieron mientras desarrollaban las aplicaciones de terapias.

12.2.5 Pydantic Es una librería para validar de datos usando Python. Sus ventajas son:

- Ofrecer una validación de datos eficiente dado que su lógica está hecha en Rust.
- Posee validadores integrados debido a que permite el uso de Type Hints.
- Brinda la posibilidad de hacer validadores personalizados [18].

Esta herramienta fue elegida porque se requería la validación de datos en varias operaciones respecto a los datos y tipos de datos manejados en la base de datos. Además, gracias al uso de los Type Hints hubo varias validaciones que directamente venían incluidas, por ejemplo, si un campo necesita un valor de tipo númerico y el usuario ingresa una letra o símbolo, directamente le daba un error explicando el porqué. También, en varios casos hubo la necesidad de crear validaciones personalizadas como por ejemplo que la contraseña tenga cierta cantidad de caracteres definidos en el código.

12.2.6 Redis Traducido al español como Servidor de Estructuras de Datos, permite almacenar datos en la memoria caché de una página web. Sus ventajas son:

- Ofrecer mejores tiempos de respuesta al realizar operaciones de lectura y escritura al almacenarse en memoria caché [19].
- Ayudar a reducir la carga en la base de datos
- Permitir establecer la expiración de los datos almacenados [20].

Este servicio fue empleado para almacenar códigos OTP y la información del usuario correspondiente. Dada que dichos códigos debían tener una duración limitada, fue pertinente el manejo de este servicio ya que emplear esta necesidad en la base de datos hubiese sido más ineficiente. En comparación a Memcached que es otra tecnología que ofrece un servicio similar, Redis ofrece mejores tiempos de respuesta [19].

12.2.7 Servidor SMTP Permite el envío de correos electrónicos haciendo uso del Protocolo de Tranferencia Simple de Email. Este es un protocolo de comunicación que facilita la transmisión de correos electrónicos por medio de servidores aprovechando la capa TCP/IP, es decir, a través del internet [21]. Sus ventajas son:

- Garantizar una comunicación confiable gracias al protocolo TCP/IP.
- Control de errores en caso de que el correo el electrónico no haya podido ser enviado.
- Simplicidad al momento de configurar el servidor [22].

Uno de los aspectos que fueron considerados para aumentar la seguridad al momento de la autenticación del usuario fue el uso de la Autenticación de Doble Factor (2FA). Adicionalmente de las credenciales de usuario válidas, se envía un

código al correo electrónico correspondiente al usuario. Adicionalmente, en caso de querer cambiar de contraseña, se envía un link al usuario a su correo electrónico. Por estas razones, el uso de un servidor SMTP fue pertinente.

12.2.8 Secrets Librería de Python encargada de generar números criptográficamente seguros para números sensibles como contraseñas o tokens de seguridad [23]. Sus ventajas son:

- Tiene la capacidad de generar números aleatorios más seguros que la librería Random.

Esta librería fue usada para generar contraseñas de un solo uso (OTP) usadas al momento de realizar la doble autenticación. Por razones de costo fue elegido el uso de esta librería en comparación a otros servicios que tienen costo como Authy. Igualmente, dado que Secrets tiene como enfoque la seguridad en los números generados, la librería Random no fue considerada.

12.3 IMPLEMENTACIÓN DEL FRONT-END

Para la implementación del Fron-End se hizo uso de librerías que facilitaron la construcción del aplicativo, como principal dato que se debe de tener en cuenta se utilizó App Router ya que esto nos brinda nuevas características mas actualizadas de Next.js. A continuación se mostrarán los componentes mas importantes de la página para dar una breve demostración de la implementación.

12.3.1 Login Esta es la página inicial en la que todo usuario debe pasar, ya que dependiendo el tipo de usuario tendrá una interfaz de usuario distinta, como primer librería llamada yup facilitó la validación de los formularios que hay en las distintas pestañas, se utilizó en la aplicación de la siguiente forma:

```
const LoginSchema = Yup.object().shape({
  username: Yup.string().required('Usuario es requerido'),
  password: Yup.string().required('Contraseña es requerida'),
});
```

Figura 6. Validaciones Yup Login

En el código anterior Fig. 6 haciendo uso de Yup se definió un esquema de validación llamado “LoginSchema”. Donde se especifica que tanto el campo de usuario como el de contraseña son obligatorios. Si alguno de estos campos está vacío, Yup generará un mensaje de error.

Para el manejo de los formularios la libreria Formik simplifica la gestión de formularios en React. Maneja el estado del formulario, la validación y el envío, haciendo uso de esta en la pagina de Login de la siguiente manera:

```
<Formik
    initialValues={{ username: '', password: '' }}
    validationSchema={LoginSchema}
    onSubmit={handleLogin}
>
    {({ isSubmitting }) => (
        <Form className="">
            <h2 className="text-lg font-bold mb-4 text-black">Iniciar sesión</h2>
            {error && <p className="text-red-500 mb-4">{error}</p>}
            <div className="mb-4">
                <label htmlFor="username" className="block text-gray-700">Usuario</label>
                <Field
                    type="text"
                    name="username"
                    className="w-full px-3 py-2 border rounded-lg"
                    disabled={isLoading}
                />
                <ErrorMessage name="username" component="div" className="text-red-500 text-sm mt-1" />
            </div>
```

Figura 7. Formulario del Login

Donde la configuración se realiza con:

- initialValues: Define los valores iniciales de los campos del formulario.
- validationSchema: Utiliza el esquema de validación de Yup que se definió anteriormente.
- onSubmit: Especifica la función que se ejecutará cuando se envíe el formulario.

Formik también proporciona ofrece como “<Field>” y “<ErrorMessage>” para manejar los inputs y mostrar errores de validación.

Para la conexión con el Back-end se hizo uso de Axios, ya que el uso de esta librería brinda un mejor manejo con los mensajes de errores.

```
const handleLogin = async (values: { username: string; password: string }) => {
    setIsLoading(true);
    setError(null);
    try {
        const response = await axios.post('http://127.0.0.1:8000/authentication/token',
            new URLSearchParams({
                username: values.username,
                password: values.password,
            }), {
                headers: {
                    'Content-Type': 'application/x-www-form-urlencoded',
                },
            }
        );

        const token = response.data.access_token;
        if (token) {
            localStorage.setItem('token', token);
            window.location.href = '/mainPage';
        }
    } catch (error) {
        setError('Credenciales inválidas o error en el servidor');
    } finally {
        setIsLoading(false);
    }
};
```

Figura 8. Llamada al Back-End con Axios

En la Figura 8 se ve que Axios realiza una petición POST a la URL especificada, enviando el nombre de usuario y la contraseña como datos del formulario. La respuesta del servidor se maneja para extraer el token de acceso y almacenarlo en el localStorage.

Por último, la librería de material UI brinda distintos componentes gráficos que mejoran la interacción con el usuario.

```
<Backdrop
    sx={{ color: '#fff', zIndex: (theme) => theme.zIndex.drawer + 1 }}
    open={isLoading}
>
    <CircularProgress color="inherit" />
</Backdrop>
```

Figura 9. Código animación de carga Login

En el login se utilizan dos componentes que realizan lo siguiente:

- Backdrop: Crea un fondo oscuro que cubre toda la pantalla cuando ‘isLoading‘ es true.

- CircularProgress: Muestra un indicador de carga circular dentro del Backdrop.

12.3.2 CancelDialog Uno de los componentes necesarios para mejorar la interfaz de usuario es un mensaje de cancelación, esto nos permite asegurar que si un usuario a la hora de estar realizando una actividad desea cancelar un formulario se confirme esta acción de cancelación.

```tsx
interface CancelDialogProps {
  open: boolean;
  onClose: () => void;
  onConfirm: () => void;
}

export function CancelDialog({ open, onClose, onConfirm }: CancelDialogProps) {
  return (
    <Dialog
      open={open}
      onClose={onClose}
      aria-labelledby="alert-dialog-title"
      aria-describedby="alert-dialog-description"
    >
      <DialogTitle id="alert-dialog-title">
        {"Confirmar cancelación"}
      </DialogTitle>
      <DialogContent>
        <DialogContentText id="alert-dialog-description">
          ¿Estás seguro de que deseas cancelar?
        </DialogContentText>
      </DialogContent>
      <DialogActions>
        <Button onClick={onClose} color="primary">
          No
        </Button>
        <Button onClick={onConfirm} color="primary" autoFocus>
          Sí
        </Button>
      </DialogActions>
    </Dialog>
  );
}
```

Figura 10. Código mensaje de cancelación

Se puede ver el uso mayoritario de Material UI para dar una buena interfaz de usuario.

12.3.3 Navbar Este componente es uno de los mas importantes ya que brinda las distintas páginas disponibles que tiene el usuario para interactuar, con esto el usuario puede elegir que actividad desea hacer, ya que sin un Navbar el usuario no tendría la posibilidad de saber que actividad va a realizar.

```
return (
  <nav className="bg-blue-300 p-6 flex justify-between items-center">
    {/* Sección Izquierda */}
    <div className="flex items-center space-x-4">
      <div className="w-10 h-10 bg-gray-300 rounded"></div>
      <div className="flex space-x-2 text-black">
        <Link href="/mainPage" className="bg-white py-2 px-4 rounded hover:bg-gray-200">Inicio</Link>
        <Link href="/patientsData" className="bg-white py-2 px-4 rounded hover:bg-gray-200">Datos</Link>
        <Link href="/userManagement" className="bg-white py-2 px-4 rounded hover:bg-gray-200">Gestionar usuario</Link>
        <Link href="/assignPatients" className="bg-white py-2 px-4 rounded hover:bg-gray-200">Asignar Paciente</Link>
      </div>
    </div>

    {/* Sección Derecha */}
    <div className="flex items-center space-x-4">
      <a href="/userProfile" className='flex space-x-0.5 shadow-xl hover'>
        <span className="text-black font-bold content-center">{userInfo ? userInfo?.username : 'Cargando...'}</span>
        <svg xmlns="http://www.w3.org/2000/svg" width="2em" height="2em" viewBox="0 0 24 24"><path fill="currentColor" d="M12 12q-1.65 0-2.825-1.175T8 8t1.175-2.825T12 4t2.825 1.1
      </a>
      <svg onClick={handleLogout} xmlns="http://www.w3.org/2000/svg" width="2em" height="2em" className='shadow-xl' viewBox="0 0 24 24"><path fill="#ff0000" d="M5 21q-.825 0-1.412-
    </div>
  </nav>
);
```

Figura 11. Código barra de navegación

La funcionalidad de este componente es el evitar que un usuario pueda entrar donde no esta permitido, comprobando el token y la sesión que se esta ingresando.

12.3.4 UserForm El componente de UserForm es el encargado de los formularios de crear usuario y editar usuario, haciendo uso de Formik y Yup para gestionar y validar los formularios, dependiendo de lo que se quiera hacer se muestra los campos necesarios para cada actividad que se quiera realizar.

```
const router = useRouter();
const [openDialog, setOpenDialog] = useState(false);

const initialValues: UserFormData = {
  id: defaultValues?.id || '',
  name: defaultValues?.name || '',
  username: defaultValues?.username || '',
  email: defaultValues?.email || '',
  phone: defaultValues?.phone || '',
  role: defaultValues?.role || 'admin',
  password: '',
};

const validationSchema = Yup.object().shape({
  name: Yup.string().required('El nombre completo es requerido'),
  username: isEditMode ? Yup.string() : Yup.string()
    .required('El nombre de usuario es requerido')
    .min(4, 'El nombre de usuario debe tener al menos 4 caracteres')
    .max(13, 'El nombre de usuario no puede tener más de 13 caracteres')
    .matches(/^[\w]+$/, 'El nombre de usuario solo puede contener letras, números y guiones bajos'),
  email: Yup.string().email('Email inválido').required('El email es requerido'),
  phone: Yup.string().required('El número de celular es requerido')
    .min(10, "El número celular debe ser de 10 dígitos numéricos"),
  role: Yup.string().oneOf(['admin', 'terapeuta', 'docente'], 'Rol no válido').required('El rol es requerido'),
  password: isEditMode
    ? Yup.string()
    : Yup.string()
      .required('La contraseña es requerida')
      .min(4, 'La contraseña debe tener al menos 4 caracteres')
      .max(20, 'La contraseña no puede tener más de 20 caracteres'),
});
```

Figura 12. Validación y valores iniciales

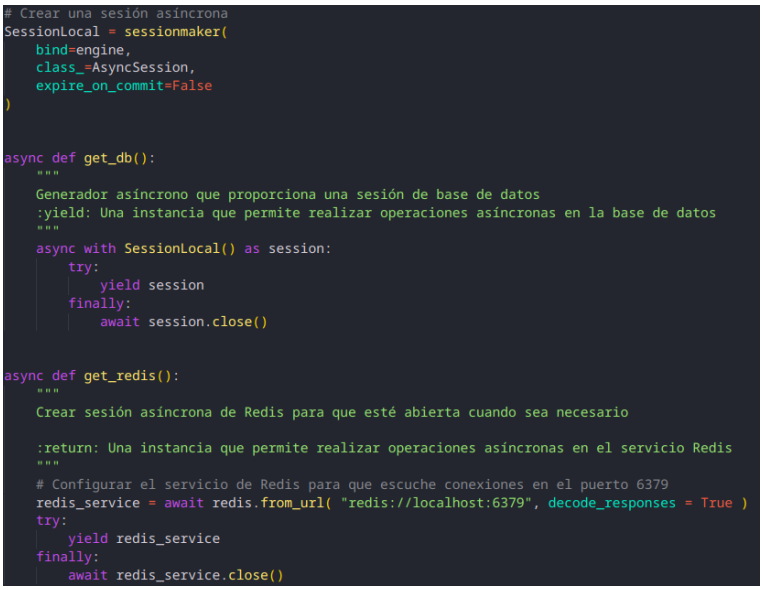
Con esto se evita que se repita código muy similar en crear usuario y editar usuario, ya que ambos tienen similitud en los datos.

12.4 IMPLEMENTACIÓN DEL BACK-END

La estructura general de los archivos del back-end consiste en un diferentes archivos con extensión de Python, cada uno enfocado en una funcionalidad puntual por motivos de orden el código, tanto para que sea más fácil para otra persona leer el código, como en caso de que se necesite realizar mantenimiento de un apartado específico. Dichos archivos están nombrados con la nomenclatura camelCase. Todos estos archivos se agrupan por medio de routers en el main. Adicionalmente, todos los archivos tienen documentación que cuentan con una descripción de qué hace la función, qué implica cada parámetro y qué retorna la función.

12.4.1 Base de datos Compuesto por los archivos database, models, schemas y crud. Cada uno cumple con una función específica enfocada a la implementación de diseño y uso de la base de datos.

Database establece la conexión a la URL de la base de datos y crea las conexiones asíncronas para la base de datos (get_db) y el servicio Redis (get_redis). Para el caso de Redis, se conecta con su puerto predeterminado que es el 6379. Ver Figura 13.



```python
# Crear una sesión asíncrona
SessionLocal = sessionmaker(
    bind=engine,
    class_=AsyncSession,
    expire_on_commit=False
)


async def get_db():
    """
    Generador asíncrono que proporciona una sesión de base de datos
    :yield: Una instancia que permite realizar operaciones asíncronas en la base de datos
    """
    async with SessionLocal() as session:
        try:
            yield session
        finally:
            await session.close()


async def get_redis():
    """
    Crear sesión asíncrona de Redis para que esté abierta cuando sea necesario

    :return: Una instancia que permite realizar operaciones asíncronas en el servicio Redis
    """
    # Configurar el servicio de Redis para que escuche conexiones en el puerto 6379
    redis_service = await redis.from_url( "redis://localhost:6379", decode_responses = True )
    try:
        yield redis_service
    finally:
        await redis_service.close()
```

Figura 13. Archivo database.py

Models representa todas las tablas, atributos y relaciones entre tablas especificados en la versión final del diseño de la base de datos. Para lograrlo, usa la libreria SQLAlchemy para posteriormente crear las tablas en el main. La forma en que cada tabla fue creada fue por medio de clases como se puede ver en la Figura 14.

```python
class User(Base):
    """
    Clase que representa el modelo de usuarios en la base de datos

    Atributos:
        id (int): Identificador único del usuario
        name (str): Nombre completo del usuario
        username (str): Nombre de usuario
        password (str): Contraseña del usuario, encriptada
        email (str): Dirección de correo electrónico única del usuario
        phone (str): Número de teléfono único del usuario
        role (str): Rol del usuario en el sistema (admin, terapeuta, docente)
    """
    __tablename__ = 'users'
    id = Column(Integer, primary_key=True, index=True)
    name = Column(String)
    username = Column(String, unique=True, index=True)
    password = Column(String)
    email = Column(String, unique=True )
    phone = Column(String, unique=True)
    role = Column(String)

    # Establecer relación con patients
    patients = relationship( "Patient", back_populates="user")
```

Figura 14. Archivo models.py

Schemas es el archivo donde se encuentran los esquemas usados a lo largo del desarrollo, cada uno cuenta con sus validadores específicos acorde a las necesidades requeridas. Esto se logra con el uso de la libreria Pydantic. Similar al archivo anterior, se emplean clases. En la Figura 5 aparece uno de los esquemas usados, exactamente siendo este el esquema de un usuario básico, donde se especifican los atributos y los validadores de los mismos. Para este caso, se encuentra un validador para garantizar la longitud del nombre y otro validador para que el campo de celular pueda ser vacío y en caso de que esté lleno, que cuente con 10 números.

```python
class UserBase(BaseModel):
    """Clase que representa un usuario básico"""
    name : str
    email : EmailStr
    phone : str | None = None

    @field_validator("name")
    def validate_name( cls, name_value ):
        """
        Validar que el nombre tenga una longitud entre 1 y 50 caracteres
        :param name_value: Cadena que contiene el nombre a validar
        :return: Nombre validado
        """
        if len(name_value) < 1 or len(name_value) > 50:
            raise ValueError( "El nombre debe contener entre 1 y 50 caracteres" )
        return name_value

    @field_validator("phone")
    def validate_phone_number(cls, phone_value):
        """
        Validar que el número de celular tenga una longitud de 10 y solo sean números
        :param phone_value: Cadena que contiene el número celular a validar
        :return: Número celular validado
        """
        # Verificar si se proporciona o no un número
        if phone_value is not None:
            if phone_value == "":
                return None  # Permitir que el campo sea None
            if not phone_value.isdigit() or len(phone_value) != 10:
                raise ValueError( "El número celular debe ser de 10 dígitos numéricos" )
        return phone_value
```

Figura 15. Archivo schemas.py

Crud contiene todos las operaciones relacionadas a la creación, lectura, modificación y eliminación de elementos de la base de datos. Se decidió crear un archivo aparte solo con estas operaciones para proveer una buena estructura del código. A forma de anotación, todas estas operaciones son asíncronas.

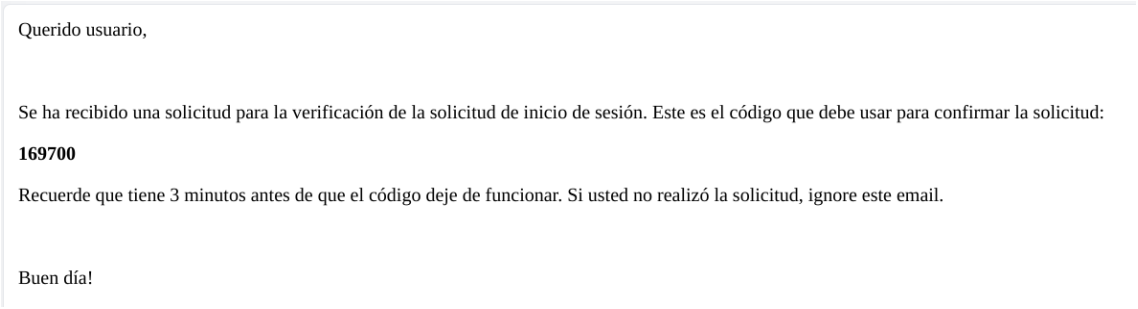
12.4.2 Autenticación de usuario e inicio de sesión La autenticación está basada en tokens JWT junto con el factor de doble autenticación usando códigos OTP de 6 digitos enviados al correo del usuario. Toda el contenido para el flujo de la autenticación está en el archivo authentication, donde además se usan las sesiones de la base de datos y Redis que fueron mencionadas en la sección 9.4.1

El usuario inicialmente escribe sus credenciales (usuario y contraseña) a través de una ruta POST, estos datos se verifican con la operación del CRUD encargada de validar si las credenciales ingresadas son correctas, comparando las ingresadas con las que están registradas en la base de datos. En caso de serlo, entonces se le envía un código de 6 digitos al correo del usuario.

Para hacer el envío de este correo se hace uso del servicio de Redis, que almacena el correo electrónico del usuario, fecha de expiración del código y el código en cuestión en la memoria caché. El usuario tiene un tiempo máximo de 2 minutos para ingresar el código y poder ingresar a la página. Pasados los 2 minutos la información de ese usuario se elimina automáticamente de la memoria caché.

Redis se encarga de almacenar la información del usuario y el correo, pero no

envía el correo. Para el envío del correo se hace uso de un servidor SMTP previamente configurado con las credenciales del servidor y el correo remitente. Este correo tiene la estructura mostrada en la Figura 6. Por otra parte, como fue mencionado en la sección 9.2.8, el código OTP es generado por medio del empleo de la librería Secrets.


Querido usuario,

Se ha recibido una solicitud para la verificación de la solicitud de inicio de sesión. Este es el código que debe usar para confirmar la solicitud:

169700

Recuerde que tiene 3 minutos antes de que el código deje de funcionar. Si usted no realizó la solicitud, ignore este email.

Buen día!

Figura 16. Ejemplo del correo enviado al usuario para doble autenticación

Posteriormente, una ruta POST se encarga verificar si el código que ingresa el usuario es correcto. En caso de ser inválido o haya expirado, el usuario debe de volver a ingresar sesión. Si es correcto, se procede con la creación del token del usuario validado por factor de doble autenticación.

La estructura definida para los token JWT está definida por 3 partes principales: encabezado, payload y firma:

- El encabezado contiene el algoritmo de firma y el tipo de token que en este caso siempre sería JWT. Se codifica en Base64-URL.
- El payload almacena un elemento llamado reclamaciones que pueden ser privadas, públicas o registradas, consiste en los datos a transmitir. Se codifica en Base64-URL.
- La firma se encarga de verificar que el token sea válido. Consiste en la concatenación del encabezado y payload codificados con un punto, después se hace una firma con una llave secreta con el algoritmo especificado en el encabezado [16].

En la Fig. 17, se encuentra un token JWT. La parte roja representaría el header codificado, la parte morada el payload codificado y la parte azul tendría la llave secreta correspondiente al algoritmo del encabezado.

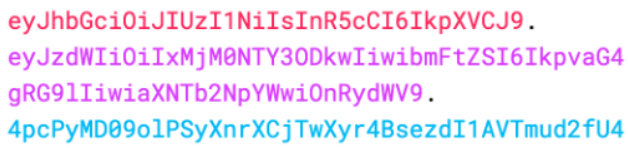


Figura 17. Ejemplo de cómo se ve un token JWT [16]

El encabezado elegido para la implementación de la aplicación consiste en el algoritmo HS256, eficiente y bastante usado para JWT que se enfoca en la autenticación. Para el payload, primero se deben de conocer los 3 tipos de reclamaciones previamente mencionados:

- Las reclamaciones privadas son tipos de datos personalizados que también creados por desarrolladores, tiene como objetivo compartir información entre las partes que hayan acordado su uso. No obstante, no son de uso libre como las públicas.
- Las reclamaciones públicas son tipos de datos creados por los desarrolladores para que sean usadas libremente por cualquier persona. JWT recomienda que estén definidas en el IANA JSON Web Token Registry para evitar riesgo de colisiones. En este sentido, las colisiones se refieren a que 2 reclamaciones sean llamadas igual pero con usos distintos.
- Las reclamaciones registradas son tipos de datos datos predefinidos y estandarizados por estándares de JWT. Tienen un significado conocido. [16]

Para la implementación de los token JWT en la aplicación, se hizo uso de reclamaciones privadas porque eran las más adecuadas a usar en la aplicación, dado que se manejan datos sensibles que no están registrados y tampoco públicos. Los datos que contienen las reclamaciones son el nombre de usuario, identificación del usuario y el tiempo de expiración del token, siendo este de 120 minutos.

Finalmente, para la firma se tiene una variable de entorno que contiene la llave secreta de 32 bytes. Es de alta importancia que la llave secreta sea de entorno, si dicha variable se escribiera en el código fuente podrían surgir riesgos de seguridad. Donde una persona no autorizada podría replicar un token, comprometiendo la integridad de los datos.

En este apartado existe una función encargada de obtener la información del usuario a partir del payload del token JWT. Esta función decodifica el token usando el token, la llave secreta (variable de entorno SECRET_KEY) y el algoritmo de codificación (HS256). Cuando se decodifica se extrae el nombre de usuario e identificación del usuario. Si es válido devuelve los datos previamente mencionados, si es inválido, esto implica que el token expiró y devuelve error de autorización, en dicho caso, el usuario debe de volver a iniciar sesión. Esta función es usada a lo largo de los archivos tanto para obtener la información del usuario, como para verificar que el token siga activo y tengo validez. Ver Figura 18

```python
@router.get("/token/get_user_info")
async def get_current_user( token : Annotated[ str, Depends(oauth2_bearer) ] ):
    """
    Obtener la información del usuario a partir del payload del token JWT.

    :param token: Cadena que representa el token JWT, obtenida a través de la autenticación de OAuth2.

    :return: Un diccionario que contiene el nombre de usuario y el ID del usuario extraídos del token JWT.
    """
    try:
        payload =jwt.decode( token, SECRET_KEY, algorithms=[ALGORITHM] )
        username : str = payload.get('sub')
        user_id : int = payload.get("id")
        if username is None or user_id is None:
            raise HTTPException( status_code = status.HTTP_401_UNAUTHORIZED,
                                 detail = "La sesión ha expirado, por favor vuelva a iniciar sesión" )
        return { 'username' : username, 'id' : user_id }

    # Si el payload no puede ser decodificado, entra aquí
    except JWTError:
        raise HTTPException( status_code = status.HTTP_401_UNAUTHORIZED,
                             detail = "La sesión ha expirado, por favor vuelva a iniciar sesión" )
```

Figura 18. Función get_current_user

12.4.3 Cambio de contraseña El primer proceso para el cambio de contraseña consiste en el uso de token JWT y enlaces mágicos con una duración máxima de 7 minutos, que son enviados al correo electrónico que el usuario escribe en el formulario. Cabe aclarar, que este es uno de los 2 procesos que existen en la aplicación para el cambio de contraseña, la diferencia entre ambos radica en que para este primer proceso el usuario aún no ha ingresado sesión, el segundo proceso es en el apartado del perfil del usuario donde ya tuvo que haber ingresado con sus credenciales previamente. Es decir, el primer proceso se usa cuando el usuario no recuerda su contraseña y el segundo cuando el usuario sí recuerde sus credenciales y desee cambiar su contraseña.

El usuario inicialmente debe de escribir el correo asociado a la cuenta a la que desea restablecer la contraseña. En este caso, fue elegido el correo y no el nombre de usuario por conveniencia. Posterior a esto independientemente si el correo existe o no, se le hace saber al usuario que el correo fue enviado, se hace de esta forma para evitar que por medio de ataques de fuerza bruta alguna persona malintencionada pueda obtener algún correo válido de un usuario. Cabe mencionar que igual a la sección anterior, se usa un servidor SMTP previamente configurado. En caso de que el correo exista en la base de datos, se le envía un mensaje al usuario con las instrucciones necesarias para proseguir con el cambio de contraseña. Ver Figura 19.

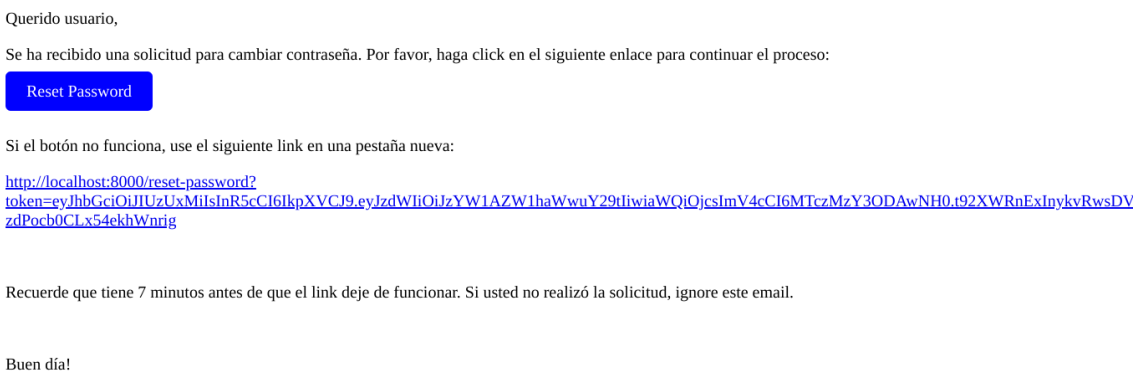


Figura 19. Ejemplo del correo enviado al usuario para restablecer contraseña

En este mensaje se le indica al usuario que dispone de un máximo de 7 minutos para completar el proceso. Para este paso, se hace uso de enlaces mágicos dándole la posibilidad al usuario de escribir el link en una pestaña manualmente o hacer click en un botón que lo redirecciona a la página correspondiente. Ambas opciones pueden ser usadas sin problema, se usan 2 formas distintas en caso de que alguna no funcione o el usuario prefiera más una que otra.

Para garantizar la integridad y duración del enlace, además de que solo funcione para el usuario cuyo correo fue previsto, se hace uso de un token JWT con una duración de 7 minutos. Compuesto por un encabezado con el algoritmo de firma HS256, el payload con reclamaciones privadas que contienen el identificador y correo electrónico del usuario y la firma con otra llave secreta diferente a la de autenticación. Este token funciona como la base para la creación del enlace mágico.

Los enlaces mágicos son enlaces únicos y temporales generados cada vez que un usuario haga una solicitud específica sin necesidad de algún PIN [24]. Estos enlaces funcionan con un fin específico, usualmente son empleados para inicios de sesión sin uso de credenciales aunque en este caso el objetivo es el cambio de

contraseña. Se decidió implementar enlaces mágicos dada su corta vida útil que ayuda a mejorar la seguridad de la aplicación. Entonces, la creación de este enlace funciona por medio del token. Cuando el token expira el link deja de funcionar.

Una vez que el usuario ingresa al link bien sea por medio del botón o de la URL, se verifica la validez del link mediante la decodificación del token JWT, si es válido, accede al formulario para el cambio de contraseña. Solo existe un validador de por medio que es para verificar que la nueva contraseña no sea igual que la anterior. Finalmente, la contraseña se encripta y se realiza el cambio en la base de datos.

12.4.4 Perfil del usuario En esta sección se encuentra el segundo proceso de cambio de contraseña. En este caso el proceso es mucho más simple que el primero dado el detalle de que el usuario ya inició con sus credenciales, doble autenticación y el token de autorización sigue válido. Por lo que ahora se encuentra un formulario similar al anterior con el mismo validador de que la contraseña no sea igual a la anterior antes de realizar los cambios a la contraseña. La nueva contraseña se cifra cuando se aplican los cambios en la base de datos.

12.4.5 Gestión de usuarios y pacientes La aplicación cuenta con 3 distintos tipos de usuarios: administrador, terapeuta y docente. La gestión de usuarios es una sección que solo puede ser accedida por administradores debido a las acciones que se pueden realizar. En esta área un administrador tiene la capacidad para añadir, editar o eliminar otros usuarios, por lo que todo dentro de esta área es exclusivo para administradores. Para la creación de nuevos usuarios se deben de considerar los siguientes datos: nombre, usuario, contraseña, correo electrónico, número de celular y rol. Para la edición se pueden editar todos los datos mencionados en la creación pero sin poder editar la contraseña

De forma similar, dentro de la gestión de pacientes le permite ver al paciente que ha iniciado sesión los pacientes que tenga asignados, permitiendo añadir pacientes nuevos de 2 formas distintas.

- Permite agregar a un paciente que ya existe a la base de datos por medio del identificador del paciente. Es decir, solo se necesita el ID para asignarlo.
- Agregar a un paciente que aún no existe en la base de datos por medio de hacer el registro pertinente. Para este caso, se deben de registrar estos datos: identificador, nombre, edad, nivel escolar, diagnóstico, género y el identificador del usuario al que se le va a asignar el paciente.

A forma de anotación. Tanto para la creación de usuarios nuevos como de pacientes nuevos, se hace uso de esquemas que cuentan con validadores para

asegurar que los datos que se reciben sean válidos.

Los validadores hechos para garantizar la validez de los datos al momento de crear un usuario nuevos son los siguientes:

1. Nombre: Debe tener una longitud entre 1 y 50 caracteres.
2. Correo electrónico: Debe de tener un @.
3. Nombre de usuario: Debe tener una longitud entre 4 y 13 caracteres. Dichos caracteres solo pueden ser letras mayúsculas y minúsculas, números y guiones bajos.
4. Contraseña: Debe de contener entre 4 y 20 caracteres.
5. Rol: Solo puede ser admin, terapeuta o docente.

Ver Anexos para ver pantallazos de las partes críticas de la aplicación.

13. VALIDACIÓN DEL CORRECTO FUNCIONAMIENTO DE UNA APLICACIÓN QUE INTEGRA EL CONJUNTO DE DATOS PROVENIENTES DE APLICACIONES DE TERAPIA INTERACTIVAS E INDEPENDIENTES

13.1 OBJETIVOS

El objetivo principal del plan de pruebas es garantizar que la plataforma web que permite visualizar y gestionar los resultados del conjunto de datos provenientes de aplicaciones de terapia, cumpla con los requisitos funcionales a través de aspectos como:

- **Funcionalidad:** Asegurar que todas las funcionalidades previstas operen correctamente bajo diferentes escenarios de uso.
- **Seguridad:** Probar la integridad del sistema en cuanto a la gestión segura de credenciales y datos sensibles.
- **Compatibilidad:** Validar que la plataforma sea accesible y funcional.

13.2 ENFOQUE

El enfoque del plan de pruebas para la aplicación consiste en una descripción de cómo se ejecutarán las pruebas, qué pruebas se harán, metodologías y herramientas para asegurar la calidad de la aplicación. El enfoque cubrirá aspectos tanto funcionales como no funcionales de la aplicación.

1. **Pruebas unitarias:** Se centrarán en la validación de componentes individuales, asegurando que cada módulo o funcionalidad del sistema opere de manera correcta.
2. **Pruebas de sistema:** Asegurará que el sistema completo, desde la autenticación hasta la visualización de los resultados, cumpla con los requisitos funcionales y no funcionales. Se evaluará el sistema en su totalidad desde la perspectiva del usuario final.
3. **Pruebas no funcionales:**
    - **Pruebas de seguridad:** Verificarán que la plataforma protege adecuadamente la información sensible del usuario.

- **Pruebas de rendimiento:** Evaluará el tiempo de respuesta del protocolo, para garantizar que se ejecuta en un tiempo razonable.

4. **Técnicas de prueba:**

   - **Pruebas basadas en casos de uso:** Los casos de prueba estarán diseñados a partir de los principales flujos de usuario, asegurando que las funciones críticas, como la autenticación, la ejecución del ETL o la visualización de resultados, se comporten correctamente.
   - **Pruebas de Caja Negra:** Estas pruebas se enfocarán en las entradas y salidas de la aplicación, garantizando que cumpla con los resultados esperados para diferentes escenarios de usuario.

5. **Métricas de éxito:**

   - **Cobertura de pruebas:** El porcentaje de casos de prueba cubiertos en comparación con los requisitos de la aplicación. La meta es lograr una cobertura mínima del 70 % en los módulos críticos.
   - **Tasa de defectos:** Se espera que las pruebas unitarias detecten la mayor parte de los errores, reduciendo el número de defectos en las pruebas de sistema.
   - **Tiempo de respuesta del sistema:** Durante las pruebas de rendimiento, la aplicación deberá responder en menos de 2 segundos bajo cargas normales.

6. **Cobertura de pruebas:** El enfoque se centrará en asegurar una cobertura de pruebas, priorizando las áreas críticas y asegurando que tanto los flujos funcionales como los aspectos no funcionales (rendimiento y seguridad) sean debidamente verificados antes de la entrega final.

13.3 ALCANCE

El alcance del plan de pruebas de la aplicación define las áreas del sistema que serán evaluadas y establece los límites de las actividades de prueba, considerando tanto las funcionalidades como los aspectos no cubiertos. Este enfoque garantiza que las áreas más importantes reciban la atención necesaria, mientras que se establecen claramente las áreas excluidas.

13.4 ÁREAS INCLUIDAS EN EL ALCANCE

- **Autenticación de usuarios**

  Consiste en la verificación completa de la autenticación, que incluye:

  - Ingreso de credenciales válidas e inválidas.
  - Comportamiento con campos vacíos o incompletos.
  - Manejo de tiempos de espera y respuesta.
  - Verificación de seguridad para asegurar que las credenciales están protegidas.
  - Verificar el correcto funcionamiento de la doble autenticación.

  Se realizarán pruebas funcionales y de seguridad para esta parte.

- **Reseteo de contraseña**

  Consiste en la verificación completa del proceso de olvido de contraseña, que incluye:

  - Comportamiento con campos vacíos o incompletos.
  - Comportamiento con correos existentes e inexistentes.
  - Manejo de tiempos de espera y respuesta.
  - Verificación del correcto envío del correo de recuperación al email especificado.

  Se realizarán pruebas funcionales.

- **Perfil de usuario**

  Verificar el correcto despliegue de la información del usuario logueado, que incluye:

  - Correcta visualización del nombre de usuario, nombre, email y rol del usuario.
  - Verificación del correcto proceso del cambio de contraseña

  Se realizarán pruebas funcionales.

- **Despliegue de datos**

  Verificar el correcto despliegue de la información de los pacientes y funcionamiento del ETL, que incluye:

- Correcta visualización del ID, edad, patología y nivel escolar del paciente.
- Poder visualizar gráficamente el resultado de las partidas.
- Verificar que se puede filtrar por terapia y dificultad.
- Verificar que el ETL funciona y responde dentro de los límites establecidos.
- Manejo de tiempos de espera y respuesta.

Se realizarán pruebas funcionales y de rendimiento.

- **Gestión de usuarios**

  Verificar el correcto manejo de la gestión de usuarios, que incluye:

  - Corroborar que solo pueda ser accesible por usuarios que sean administradores.
  - Verificar que es posible añadir, editar y borrar usuarios.
  - Comportamiento con campos vacíos o incompletos.

  Se realizarán pruebas funcionales.

- **Gestión de pacientes**

  Verificar el correcto manejo de la gestión de pacientes asignados al usuario logueado, que incluye:

  - Corroborar que solo pueda acceder a los pacientes asignados.
  - Verificar que es posible añadir o deasignar pacientes.
  - Comportamiento con campos vacíos o incompletos.

  Se realizarán pruebas funcionales.

- **Cobertura:** Validar la seguridad de la aplicación, que incluye:

  - Protección de credenciales y datos sensibles durante la autenticación.
  - Cifrado de contraseñas.

13.5 ÁREAS EXLUIDAS EN EL ALCANCE

- **Envío de datos proveniente de las aplicaciones de terapia interactivas e independientes:** Durante el desarrollo del ETL, fue necesario modificar ambas aplicaciones para poder hacer el envío de datos a un Bucket S3 de AWS. Esto se realizó para poder probar el correcto funcionamiento del ETL y visualización de resultados, no obstante, el envío de datos no será incluido en las pruebas.

- **Compatibilidad con navegadores o dispositivos móviles:** Las pruebas no se centrarán en versiones de navegadores web que ya no tienen soporte oficial, tampoco se validará el funcionamiento en dispositivos móviles.

13.6 ENTREGABLES DE PRUEBAS

1. Documento que describe en detalle la estrategia de pruebas implementada para la aplicación, incluyendo las conclusiones y resultados obtenidos.
2. Resultados de la pruebas.

13.7 DEPENDENCIAS

1. **Acceso a entornos de prueba:** Las pruebas requieren un entorno de prueba bien configurado. Asegurarse de que los recursos para el entorno de prueba estén preconfigurados y listos antes del inicio de las pruebas.
2. **Disponibilidad de datos de prueba:** Se requieren datos representativos para realizar las pruebas. Se puede considerar el generar datos de prueba automáticamente o manualmente.

13.8 ENTORNO DE PRUEBAS

1. **Configuración del hardware:**

   Asegurar que los servidores necesarios para el uso de la aplicación estén disponibles y configurados adecuadamente.

2. **Configuración de navegadores:**

   Para asegurar la compatibilidad, el entorno de pruebas debe incluir Brave en sus última versión para verificar la compatibilidad con la aplicación.

3. **Entorno de prueba local:** Para la implementación de las pruebas, se usará un entorno de pruebas local.
4. **Simulación de condiciones reales** Es necesario considerar un número considerable de datos para evaluar el correcto funcionamiento del ETL bajo un número alto de datos.

13.9 INFORME DE PRUEBAS

Para realizar las pruebas, se utilizaron las siguientes herramientas:

- **Selenium:** Empleado para evaluar funcionalidades que requerían una visualización detallada del flujo paso a paso, facilitando la detección de posibles errores en la interacción del usuario con la interfaz.
- **Postman:** Utilizado para verificar el correcto funcionamiento de los endpoints, asegurando que las principales funcionalidades de la aplicación respondieran según lo esperado.

La siguiente imagen. Ver figura20, muestra el reporte de las pruebas de una manera rápida siguiendo la estructura dada en [25].

Reporte de pruebas

Funciones	Descripción	Porcentaje exitoso	Pruebas pendientes	Prioridad	Observaciones
Autenticación de usuarios	Verificar la correcta autenticación de un nuevo usuario, evaluando el funcionamiento del 2FA y comprobar cómo se comporta el sistema con campos inválidos y/o vacíos.	100%	Pruebas de carga y estrés. Pruebas de seguridas más extensas	Alta	Todo el proceso de la autenticación, desde la verificación de credenciales hasta el ingreso del OTP, toma aproximadamente 3 segundos la primer vez, estos datos se almacenan en caché y los ingresos de sesión posteriores tardarían aproximádamente 0.6 segundos.
Reseteo de contraseña	Verificar el proceso de olvido de contraseña, se evalúa el correcto envío del email, contraseñada generada válida y manejo con emails existentes, inexistentes y campos vacíos.	100%	Pruebas de carga, estrés y usabilidad.	Alta	El proceso tuvo un tiempo promedio de 1.2 segundos
Perfil de usuario	Verifica la correcta visualización de los datos relacionados al usuario (nombre, nombre de usuario, email y rol). Además de comprobar que el cambio de contraseña es exitoso, manejando campos vacíos y contraseñas no iguales.	100%	Pruebas de carga, estrés y usabilidad.	Alta	Se debe de verificar que la contraseña del usuario no sea visible.
Despliegue de datos	Verifica el correcto filtro para elegir entre terapias y niveles de dificultad. Adicionalmente, se debe de asegurar que la ejecución del ETL funciona en un tiempo razonable.	100%	Pruebas de carga, estrés y usabilidad.	Alta	Se evaluó el funcionamiento del ETL con una carga de 70 datos de aplicaciones de terapia, logrando un tiempo de 48.5 segundos para todo el proceso de ETL.
Gestión de usuarios	Verifica la gestión de usuarios (añadir, eliminar o editar usuarios), además de verificar que estas acciones solo están disponibles para usuarios de tipo administrador.	100%	Pruebas de carga, estrés y usabilidad.	Alta	En las acciones de añadir y editar usuario se comprobó el manejo de campos inválidos o vacíos.
Gestión de pacientes	Verifica la gestión de pacientes (añadir pacientes nuevos o existentes y deasignar pacientes)	100%	Pruebas de carga, estrés y usabilidad.	Alta	En las acciones de añadir pacientes nuevos se comprobó el manejo de campos inválidos o vacíos; la acción de añadir pacientes existentes funciona por medio de ID y se muestra el nombre de los pacientes con un ID similar al que se haya escrito de momento (es decir, si un usuario escribe "12" le aparecerán todos los pacientes sin asignar cuyo ID inicie por "12")

Figura 20. Informe simple del resultado de las pruebas

Fueron seguidos estos pasos para cada sección:

1. **Autenticación de usuarios:** Abrir el sitio web, ingresar credenciales en los campos correspondientes, hacer clic en el botón de login, observar el resultado y rediccionar la página. Los resultados esperados consistían en el acceso correcto

con credenciales válidas, redirección al panel de inicio y mensaje de error visible para credenciales inválidas o campos vacíos.

2. **Reseteo de contraseña:** Ingresar a la página para el reseteo de contraseña, ingresar el correo electrónico en el campo correspondiente, entrar en la bandeja del correo y obtener la contraseña generada. Los resultados esperados fueron que el cambio de contraseña debe ser exitoso. Se tiene la siguiente observación: en caso de que el correo de la entrada no esté registrado, no debe de haber un mensaje de 'este correo no existe' para evitar problemas de seguridad.

3. **Perfil de usuario:** Ingresar a la pestaña de perfil de usuario, revisar los datos desplegados al lado izquierdo de la pantalla, hacer click en el botón de cambiar contraseña y llenar el formulario para el cambio de contraseña. Se comprobó que los resultados esperados fueran que el nombre, nombre de usuario, correo y rol del usuario sean desplegados correctamente y que el botón de cambio de contraseña fuera exitoso con mensajes de error pertinentes en caso de haber. A forma de nota: la contraseña no debe de estar desplegada en la información del usuario.

4. **Despliegue de datos:** Ingresar a la pestaña de datos, usar el ETL dando click en el botón de ejecutar ETL, filtrar pacientes, revisar los datos del paciente filtrado, filtrar terapias y niveles de dificultad y revisar los gráficos de los resultados. Los resultados esperados fueron que el ETL debía funcionar correctamente, las terapias, pacientes y niveles de dificultad deben de poder filtrarse exitósamente y los gráficos deben de funcionar mostrando la información correcta.

5. **Gestión de usuarios:** Ingresar a la pestaña de gestión de usuario, revisar la lista de usuarios, editar un usuario y llenar su formulario, crear un usuario y llenar su formulario, borrar usuario. Los resultados esperados consisten en que las acciones de crear, editar y borrar usuarios funcionen correctamente con mensajes de error en caso de ser necesarios, además de que los usuarios deben de listarse correctamente. Esta pestaña solo debe de ser visible para usuarios que sean administradores.

6. **Gestión de pacintes:** Ingresar a la pestaña de asignar paciente, revisar los pacientes listados, crear un paciente nuevo, deasignar un paciente existente y asignar un paciente existente. Los resultados esperados consistían en corroborar que las acciones de crear, asignar y deasignar pacientes funcionen correctamente y que los pacientes sean listados correctamente. Un usuario solo puede gestionar los pacientes que tiene asignados.

Para ver una información más detallada respecto a los resultados de las pruebas, revisar el siguiente hipervínculo: **Informe detallado del resultado de las pruebas**

14. CONCLUSIONES

Esta aplicación, diseñada principalmente con NextJS, FastAPI, Python y PostgreSQL permite la gestión de usuarios, pacientes, visualización de resultados de los pacientes y extracción de datos. Espera ser el primer paso para una aplicación con más funcionalidades para apoyar al Instituto de Niños de Ciegos y Sordos del Valle del Cauca con su labor.

Uno de los puntos clave del desarrollo de este proyecto fue el ETL, desde su elección hasta su implementación. Se logró implementar de tal forma que fue posible extraer 70 archivos con un tiempo de 48 segundos, además de asegurar la integridad de los datos extraídos.

Adicionalmente, a pesar de que no fueron realizadas pruebas de forma extensiva, sí fueron realizadas pruebas para verificar que los puntos críticos de la aplicación operaran correctamente y, en el caso particular del ETL, que además funcionara dentro de un margen de tiempo considerable,

Por otro lado, fueron encontrados desafíos durante el desarrollo de la aplicación, particularmente con el desarrollo del ETL. Debido a que las aplicaciones de terapia interactivas e independientes exportaban los datos únicamente de forma local. No obstante, este y demás desafíos fueron superados satisfactoriamente.

A pesar de esto, fue posible cumplir con los objetivos previstos y, pese a que es una realidad que existen bastantes funcionalidades posibles que se pueden añadir, se espera que la aplicación tenga la capacidad de apoyar a los terapeutas y docentes (usuarios) en el sentido de tener una forma más simple y accesible a los datos con su correspondiente visualización, idealmente, se estima que de esta forma sea más eficiente para los usuarios ayudar a los niños para mejorar o detectar su condición.

Finalmente, durante este proyecto fue puesto en práctica gran parte del conocimiento aprendido durante la carrera de Ingeniería de Sistemas y Computación. Logrando concretar la aplicación gracias a los conocimientos aprendidos y apoyo del director y codirector de tesis. Aunque hayan existido elementos no estudiados directamente en la carrera, sirvieron como cimiento para poder aprender y emplear el nuevo conocimiento aprendido.

15. TRABAJO FUTURO

En trabajos futuros, se plantea mejorar la funcionalidad de la aplicación web, incorporando características como un calendario o un sistema de registro para planificar y visualizar las terapias programadas diariamente. Esto permitiría un mayor control y organización de los datos que se ingresan en la base de datos, optimizando la gestión y el seguimiento de la información.

Además, para optimizar los procesos de extracción, transformación y carga (ETL), se propone explorar y aprovechar servicios avanzados de AWS. En particular, el uso de herramientas como AWS Data Pipelines [26], podría automatizar y mejorar la eficiencia de estos procesos, asegurando una mayor escalabilidad y confiabilidad en el manejo de los datos.

En términos de seguridad, se puede considerar el uso de AWS Secrets Manager para almacenar las credenciales de acceso al usuario del Bucket S3.

Se puede evaluar la adición de más terapias, como se mencionó en el alcance este proyecto solo incluye 2. Se debe de hacer el uso correcto de los componentes ya diseñados para así disponer de las visualizaciones de los datos dependiendo de la aplicación de terapia interactiva que se quiera ver. Adicionalmente, se debe de añadir la nueva terapia en la tabla Therapies y se debe de añadir cada métrica en la tabla Metrics.

Por otra parte, en caso de que se añada una o más terapias nuevas al sistema, se debe de también de realizar cambios al ETL y asegurarse que las terapias se suben a AWS en un formato correcto. Este formato debe de contener el ID del paciente que jugó la partida y su fecha dentro del nombre del archivo. De manera adicional, se debe de modificar el proceso de extracción y transformación para poder recibir los nuevos datos, donde se debe de asegurar que los nuevos sean transformados de tal forma que estén todas las columnas de la tabla variableResults.

En el sentido general de los códigos, se aseguró usar buenas prácticas y documentación para que el código sea legible en caso de que se modifique en un futuro.

Asimismo, se podría implementar la doble autenticación con otro servicio, de esta forma, el usuario podría elegir cuál desea usar o emplear uno en caso de que el otro falle. Una posibilidad sería mediante el uso de aplicaciones móviles como Authy o Microsoft Authenticator.

Finalmente, en caso de el Instituto tenga un servidor propio y los datos de las aplicaciones interactivas ahora se almacenen en dicho servidor, entonces se debe de modificar la extracción para recibir los datos del servidor.

REFERENCIAS

[1] L. J. E. Dewi, I. N. S. W. Wijaya y K. A. Seputra, «Web-based Buleleng regency agriculture product information system development,» *Journal of Physics: Conference Series*, vol. 1810, n.º 1, pág. 012 029, mar. de 2021.

[2] K. Tabi, A. Randhawa, F. Choi et al., «Mobile Apps for Medication Management: Review and Analysis,» *JMIR Mhealth Uhealth*, vol. 7, n.º 9, e13608, 2019.

[3] M. Schmidt, S. A. J. Schmidt, K. Adelborg et al., «The Danish health care system and epidemiological research: from health care contacts to database records,» *Clinical Epidemiology*, vol. 11, págs. 563-591, 2019. DOI: 10.2147/CLEP.S179083.

[4] «ÇiegosyÏnstituto para niños ciegos y sordos.» (2024), dirección: https://ciegosysordos.org.co/ (visitado 01-01-2024).

[5] J.-C. M. et al., «Using Software Product Lines to Support Language Rehabilitation Therapies: An Experience Report,» en *2018 ICAI Workshops (ICAIW)*, Bogota, Colombia, 2018, págs. 1-6. DOI: 10.1109/ICAIW.2018.8554992.

[6] A. Pérez-Muñoz, P. Ingavélez-Guerra e Y. Robles-Bykbaev, «New approach of serious games in ludic complements created for rehabilitation therapies in children with disabilities using Kinect,» en *2018 IEEE XXV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, Lima, Peru, 2018, págs. 1-4. DOI: 10.1109/INTERCON.2018.8526464.

[7] C. Putnam y J. Cheng, «Therapist-centered requirements: A multi-method approach of requirement gathering to support rehabilitation gaming,» en *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, Karlskrona, Sweden, 2014, págs. 13-22. DOI: 10.1109/RE.2014.6912243.

[8] R. Elmasri y S. Navathe, *Fundamentals of Database Systems*, 7th. 2015.

[9] Oracle. «Oracle Database SQL Language Reference.» (2021), dirección: https://docs.oracle.com/en/database/oracle/oracle-database/21/sqlrf/SQL-Standards.html#GUID-92B38403-6934-4E86-9D9A-E94957ACDDFC (visitado 01-01-2024).

[10] MongoDB. «NoSQL.» (), dirección: https://www.mongodb.com/ (visitado 01-01-2024).

[11] C. Hamilton, M. Lovarini, A. McCluskey, T. Folly de Campos y L. Hassett, «Experiences of therapists using feedback-based technology to improve physical function in rehabilitation settings: a qualitative systematic review,» *Disability and Rehabilitation*, vol. 41, n.º 15, págs. 1739-1750, 2019. DOI: 10.1080/09638288.2018.1446187.

[12] «What is ETL?» (), dirección: https://aws.amazon.com/what-is/etl/ (visitado 01-01-2024).

[13] «FastAPI.» (), dirección: https://fastapi.tiangolo.com/ (visitado 01-01-2024).

[14] «Flask vs FastAPI.» (), dirección: https://www.imaginarycloud.com/blog/flask-vs-fastapi (visitado 01-01-2024).

[15] «JWT Introduction.» (), dirección: https://jwt.io/introduction (visitado 01-01-2024).

[16] «An Introduction to OAuth 2.» (), dirección: https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2 (visitado 01-01-2024).

[17] «SQLAlchemy.» (), dirección: https://www.sqlalchemy.org/ (visitado 01-01-2024).

[18] «Pydantic Documentation.» (), dirección: https://docs.pydantic.dev/latest/why/#customisation (visitado 01-01-2024).

[19] «Redis Overview.» (), dirección: https://www.ibm.com/topics/redis (visitado 01-01-2024).

[20] «Why You Should Use Redis Cache.» (), dirección: https://medium.com/@vndpal/why-you-should-use-redis-cache-2e48bdd2c0be (visitado 01-01-2024).

[21] «What is SMTP?» (), dirección: https://www.cloudflare.com/learning/email-security/what-is-smtp/ (visitado 01-01-2024).

[22] «Advantages and Disadvantages of SMTP.» (), dirección: https://www.ecstuff4u.com/2020/05/advantage-disadvantages-smtp.html (visitado 01-01-2024).

[23] «Python Secrets Module.» (), dirección: https://docs.python.org/3/library/secrets.html (visitado 01-01-2024).

[24] «What is a Magic Link and How Does It Work?» (), dirección: https://help.karbonhq.com/en/articles/6415546-what-is-a-magic-link-and-how-does-it-work (visitado 01-01-2024).

[25] Guru99, *How Test Reports Predict the Success of Your Testing Project*, Accessed: 2025-01-14, 2025. dirección: https://www.guru99.com/how-test-reports-predict-the-success-of-your-testing-project.html.

[26] Amazon Web Services, *AWS Data Pipeline: What is AWS Data Pipeline?* [Accessed: January 15, 2025], 2025. dirección: https://docs.aws.amazon.com/es_es/datapipeline/latest/DeveloperGuide/what-is-datapipeline.html.

16. ANEXOS

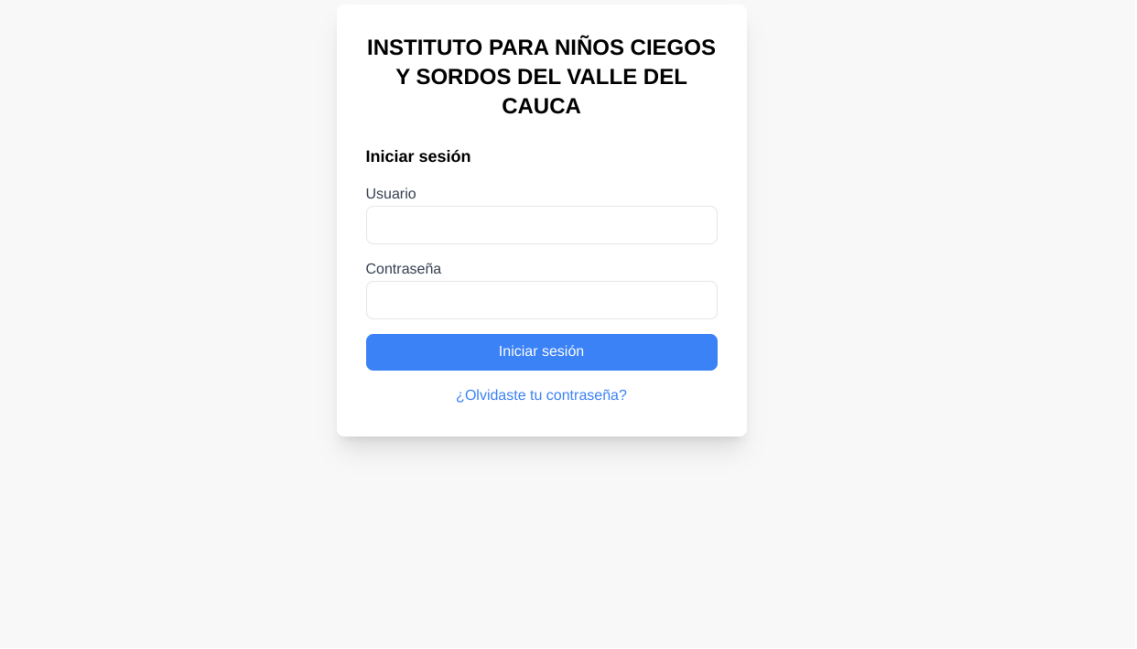


Figura 21. Ingreso de sesión

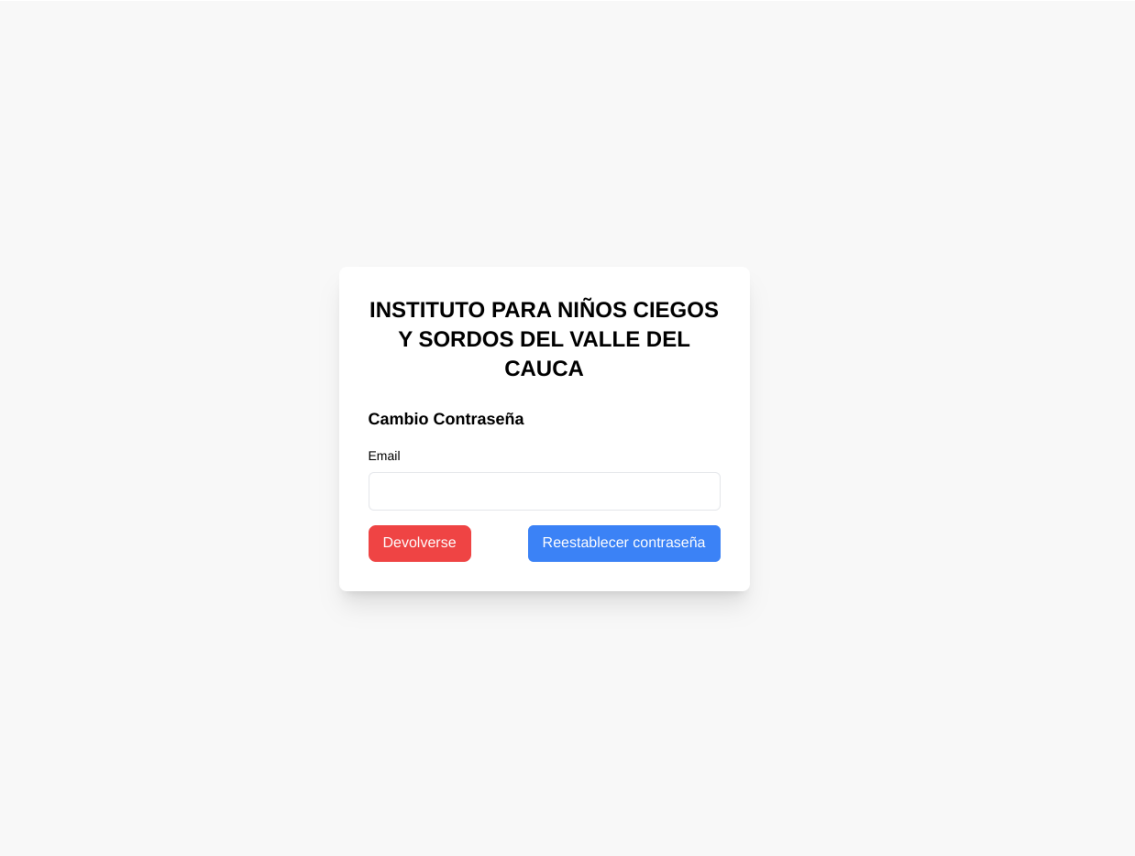


Figura 22. Olvido de contraseña

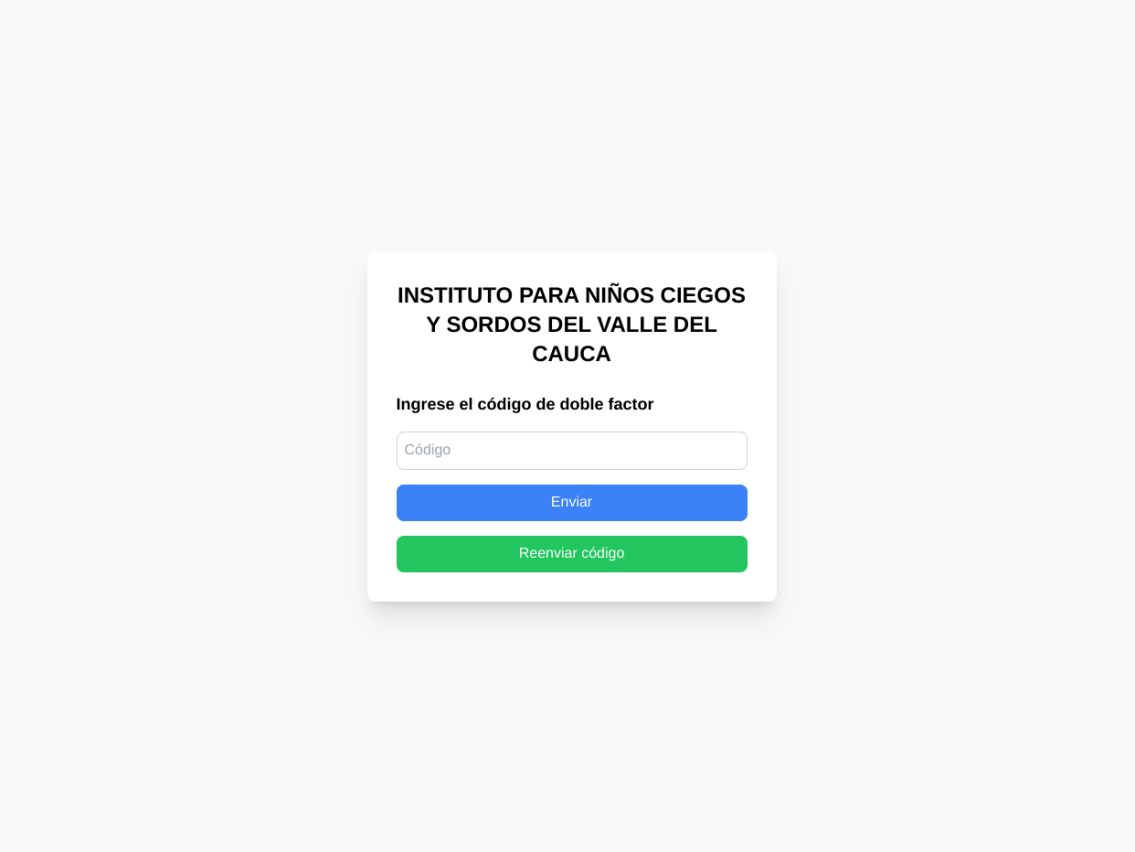


Figura 23. Doble autenticación

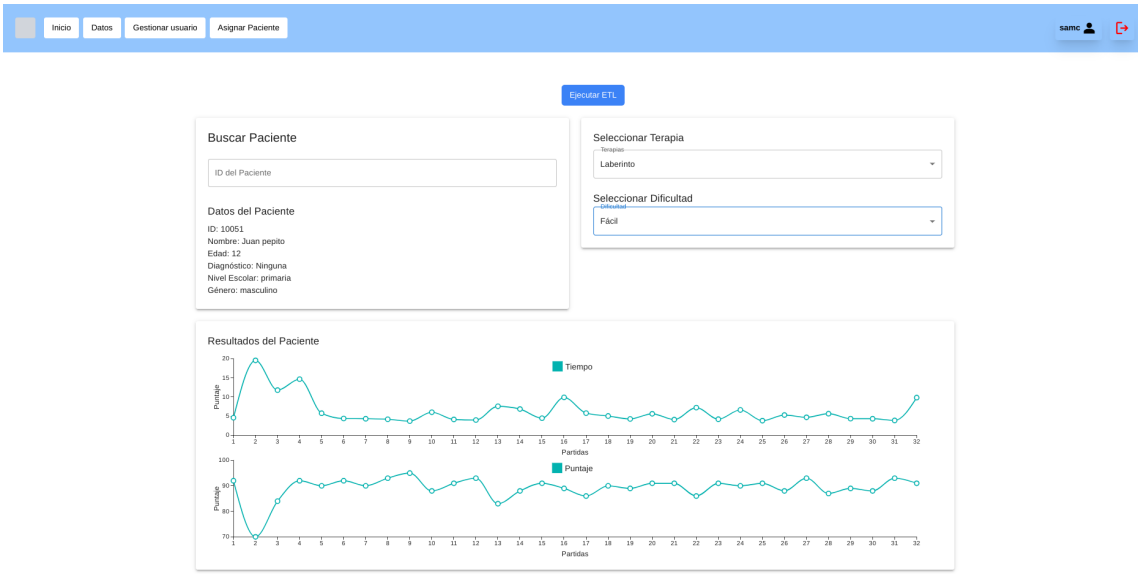


Figura 24. Pestaña de datos y visualización de gráficos

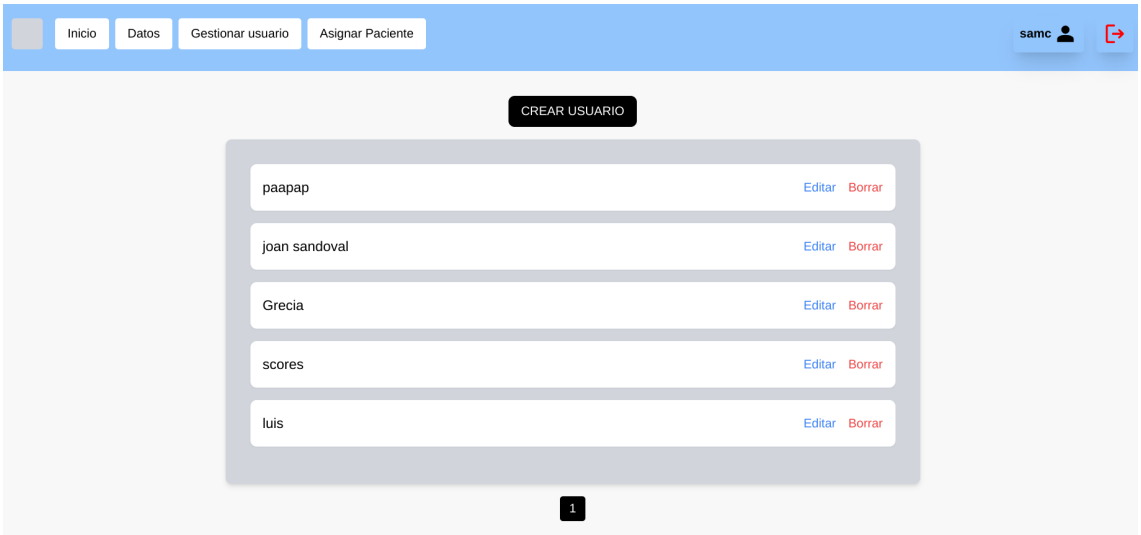


Figura 25. Pestaña de gestión de usuarios

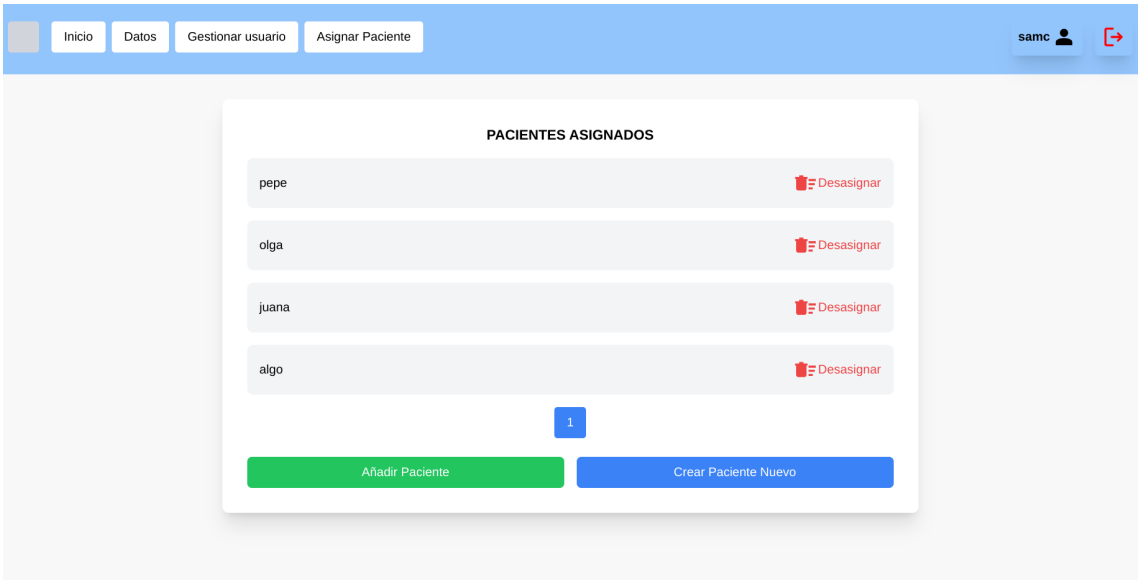


Figura 26. Pestaña para gestión de pacientes

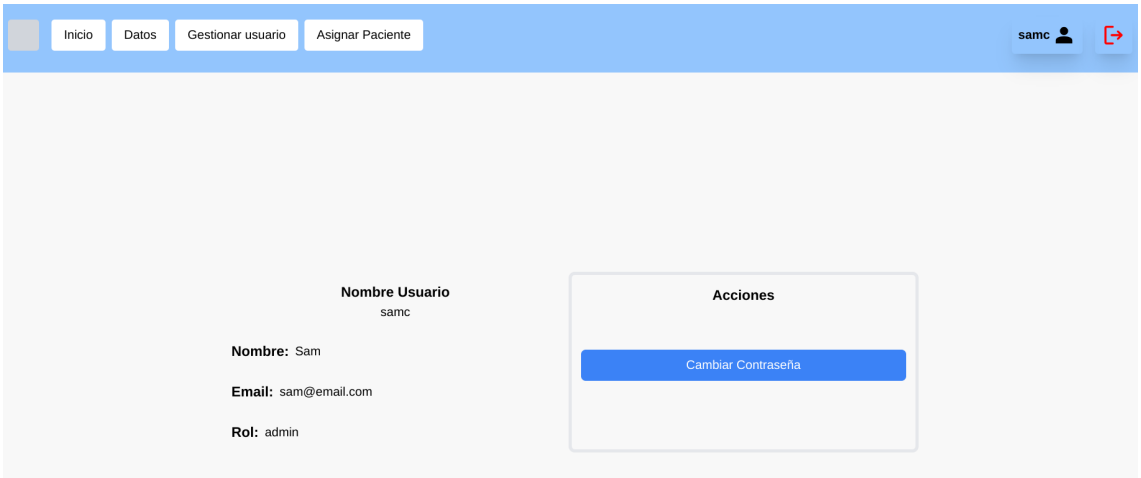


Figura 27. Pestaña de perfil de usuario