UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA

INFORMÁTICA Y CIENCIAS DE LA COMPUTACIÓN

SOFTWARE PARA APOYO AL CUIDADOR INFORMAL

POR

Alan Fidel Cotal Palma

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero Civil Informático

Patrocinante:

Geoffrey Hecht

Profesores guía:

Pamela Guevara

Claudia Sáez

Agosto 2023

Concepción (Chile)

Resumen

Los adultos mayores que presentan una enfermedad neurodegenerativa como la demencia, requieren ser cuidados todo el tiempo. En ocasiones esta labor la cumple una persona que no tiene los conocimientos necesarios para esto, por lo que se les dificulta efectuar esta tarea. Por lo tanto se ha creado una aplicación web multiplataforma llamada "*CareForMe"*, la cual brinda apoyo a los cuidadores informales de manera que puedan encontrar información relevante y test neuropsiquiátricos, tanto para el autocuidado como para el cuidado del adulto mayor. Además de tener a disposición los test para que el médico o personal de salud correspondiente pueda evaluar al cuidador.

En base a esto, la presente memoria de título tiene como objetivo desarrollar mejoras a esta aplicación en los ámbitos visuales como funcionales, agregando nuevas características con el objetivo de que la herramienta sea más efectiva y completa. Con esta actualización del software, el cuidador podrá tener una atención más personalizada de parte del personal de salud, ya que ahora existirá una relación entre ellos dentro de la plataforma, lo que al médico le permitirá ver consultas, registro de eventos anormales y resúmenes gráficos del paciente.

Summary

Older adults suffering from neurodegenerative diseases such as dementia require constant care. At times, this task falls on individuals without the necessary knowledge, making it challenging for them to carry out. As a result, a multiplatform web application named "CareForMe" has been developed to offer assistance to informal caregivers in finding relevant information and neuropsychiatric tests for both self-care and care of the elderly. Additionally, the tests are available for the corresponding physicians or health professionals to evaluate the caregiver.

Based on this, the current dissertation aims to enhance the visual and functional elements of this application by incorporating new features to make the tool more effective and comprehensive. This software update will enable caregivers to receive more personalized attention from healthcare professionals, as a relationship between them will be established within the platform. This will allow physicians to view consultations, record abnormal events, and graphical summaries of the patient.

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

1. Introducción

La demencia es un trastorno que afecta la memoria, el pensamiento y la razón de las personas. Es por esto, que los cuidadores informales o no profesionales enfrentan muchos desafíos en el desempeño de esta tarea, ya que son los responsables del cuidado y apoyo de estos pacientes en el ámbito domiciliario.

Uno de los problemas que enfrentan es el agotamiento físico y mental/emocional. Las personas que padecen de demencia requieren atención, supervisión y asistencia constante con las actividades básicas del día a día. Además, según el estudio de Villalobos [1], otro problema común es la falta de apoyo y el acceso a recursos educativos para los cuidadores informales en Chile.

La falta de información y orientación adecuada es una barrera para el trabajo de los cuidadores y puede tener un impacto negativo en ellos mismos y en los pacientes. Un estudio realizado en Chile por Sandoval [2] analizó la calidad de vida y necesidades de cuidadores informales de personas con demencia. Donde se descubrió que experimentaban altos niveles de estrés y sobrecarga de trabajo, lo que afectaba de forma negativa en su calidad de vida. Además, también se enfatizó la necesidad de apoyo psicológico para enfrentar de mejor manera esta labor.

Por lo descrito anteriormente, *Diego Muñoz* ha creado un Software Web que denominó *CareForMe* [3], la cual proporciona apoyo a los cuidadores en base a información recabada por *Constanza Ortiz* en su memoria de título [4]. Además, permite distintas funcionalidades que se describen brevemente a continuación:

- Permite visualizar distintos contenidos relacionados a los cuidados del adulto mayor y asimismo del cuidador.
- Permite realizar dos test llamados “NPI-Q” y “Zarit”, estos están enfocados en evaluar al adulto mayor y al cuidador, respectivamente.
- Se permite registrar en la web como “cuidador” o “personal de salud”.
- El personal de salud puede ver la información de todos los cuidadores y los test realizados.

En la presente memoria de título se presenta un proyecto para mejorar y añadir funcionalidades extras a este software. La idea es proporcionar a los cuidadores una herramienta más efectiva y completa para hacer frente a sus responsabilidades. Además, se agregaron nuevas funcionalidades para el personal de la salud a cargo.

Para desarrollar este proyecto, se emplearon técnicas de diseño de software [5], centrándose en la experiencia del usuario y la usabilidad. Además, se desarrolló en base a *CareForMe*, escalando el código ya creado en Python y utilizando el framework Django.
Este proyecto tiene como objetivo mejorar la calidad de vida de los cuidadores y de los adultos mayores a su cargo, ofreciendo una herramienta más efectiva y completa para la gestión de los cuidados.

1.1. Objetivos

Los objetivos del proyecto se definieron en conjunto con el cliente (*Pamela Guevara)*, dónde en principio el autor recibió información sobre las nuevas funcionalidades que requería la plataforma. Luego, se llegó a un consenso con ideas nuevas de parte del memorista como el registro de los datos del paciente, registro de eventos o síntomas, gráficos de datos y sección de atención específica para el personal de salud.
De esta manera se lograron definir las nuevas funcionalidades y a continuación se muestran los objetivos para el proyecto.

1.1.1. Objetivo General

El propósito de este proyecto consiste en crear una nueva versión de un software creado previamente (CareForMe) [3], al cual se le añadirán nuevas funcionalidades con el fin de agregar valor, tanto a nivel visual como funcional.

1.1.2. Objetivo Específicos

- **Registrar los datos del paciente**: Se implementará un sistema de registro para que los cuidadores puedan ingresar los datos de la persona a su cargo.
- **Enlazar agente Cuidador con Profesional de la salud**: Se permitirá a los cuidadores enlazar sus registros con los profesionales de la salud que están a cargo del tratamiento del

paciente, para que puedan acceder a la información relevante en tiempo real y coordinar mejor la atención.

- **Agregar sección de preguntas**: Se incluirá una sección para que los cuidadores publiquen dudas sobre algún tema en particular y que estas puedan ser respondidas por el personal de salud, además de ponerlas a disposición de los demás cuidadores.
- **Mejora de interfaz con base bibliográfica**: Se mejorará la interfaz de usuario para que los cuidadores puedan acceder a la información relevante de manera más rápida y sencilla, utilizando una base bibliográfica actualizada [6].
- **Agregar sección de registro de síntomas o eventos del paciente:** Se incluirá una sección para el registro de síntomas y eventos relevantes, de manera que los cuidadores puedan llevar un registro completo de la evolución de la persona a su cargo.
- **Análisis de los datos y reportes automatizados**: Se realizará un análisis de los datos registrados para generar reportes automatizados que brinden una visión general de la condición del paciente y permitan una mejor toma de decisiones en la atención.
- **Sección de consulta para el personal de salud a la hora de la evaluación del paciente**: Se creará una sección específica para el personal de salud, donde puedan acceder a los registros de los tests y los eventos registrados por el cuidador, para obtener una visión completa de la evolución del paciente.
- **Crear pruebas automatizadas del software**: A través de pruebas unitarias para los componentes funcionales del software, se verificará el correcto funcionamiento.

2. Discusión Bibliográfica

A continuación se describe la base bibliográfica con la que se ha realizado el desarrollo de la nueva versión de *CareForMe*, la cual se basa principalmente en los principios de diseño web. Con respecto a la motivación y la base bibliográfica del proyecto inicial, se puede encontrar en el informe de *Diego Muñoz* [3], aunque a continuación se resumirá una parte de esta, con el fin de evidenciar algunas plataformas similares.

La primera plataforma se llama *YoTeCuido*. Es un app creada por AFACO [26], disponible en IOS y Android. Tiene por objetivo brindar ayuda a cuidadores y afectados por Alzheimer,

ofreciendo información sobre la enfermedad, resolviendo dudas por cada tema. Además, ofrece rutinas de ejercicio físico y recursos como información de centros de Alzheimer en el país (España), glosarios y manuales. Lo más destacado de la plataforma, es su navegación rápida y su interfaz que se puede apreciar en la ***Figura 2.1***. Pero carece de ilustraciones, interactividad y solo está disponible para dispositivos móviles.

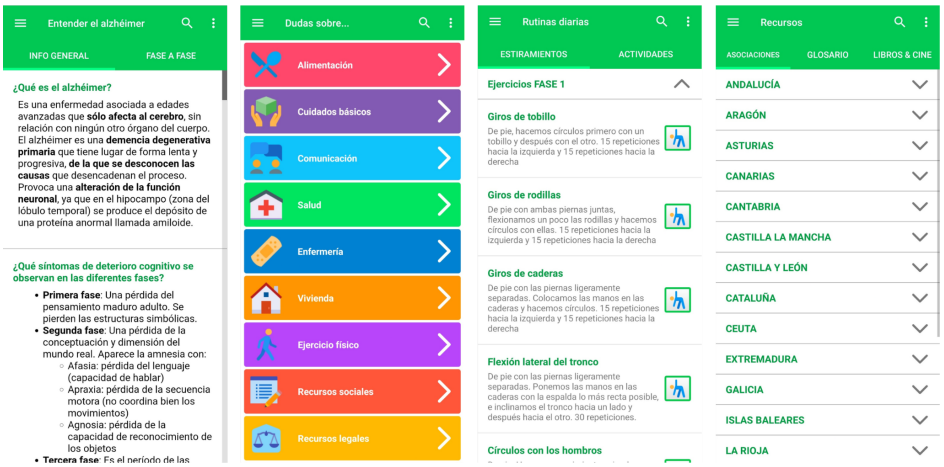

Figura 2.1: App Android *YoTeCuido*

Otra opción es *SerCuidador* [27], una plataforma desarrollada por la Cruz Roja Española para cuidadores no profesionales de personas dependientes. Disponible en web, iOS y Android. Ofrece guías, infografías y contenido de profesionales en cuidado de adultos mayores. También posee blogs y testimonios. Uno de los puntos positivos es que incluye una evaluación de sobrecarga para cuidadores y una interfaz web organizada (Ver ***Figura 2.2***). No obstante, la interfaz para dispositivos móviles tiene problemas de rendimiento y diseño desactualizado (Ver ***Figura 2.3***).

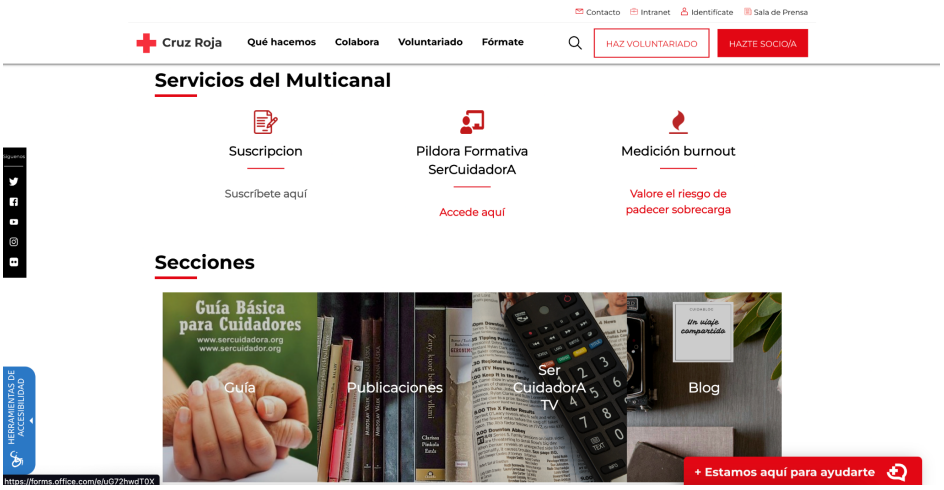

Figura 2.2: Sitio web *SerCuidador*

Figura 2.3: App Android *SerCuidador*

Ambas plataformas descritas anteriormente comparten la función de proporcionar información sobre autocuidado y cuidados para adultos mayores o personas dependientes. Algunas se destacan por su interfaz o funcionalidades, de esta manera influyen en la creación de *CareForMe* y en su actualización.

Por otra parte, una de las principales bases sobre el diseño de interfaces de usuario (UI) efectiva es escrita por Jakob Nielsen [7] , donde presenta diez heurísticas como guía

proporcionando un conjunto sólido de principios para diseñar interfaces de usuario intuitivas, eficientes y fáciles de usar, con el objetivo de mejorar la experiencia del usuario y maximizar la usabilidad de los sistemas. En la ***Tabla 2.1*** se puede ver un resumen de estas heurísticas.

Título	Descripción
1. Visibilidad del estado del sistema	La interfaz debe mantener informado al usuario sobre el estado actual del sistema, proporcionando retroalimentación visual clara y comprensible.
2. Sincronía entre el sistema y el mundo real	El diseño de la interfaz debe seguir las convenciones y terminología del mundo real, para que los usuarios puedan relacionarse fácilmente con el sistema.
3. Control y libertad del usuario	Los usuarios deben tener el control total de la interfaz y la libertad de realizar acciones reversibles sin consecuencias graves.
4. Consistencia y estándares	La interfaz debe ser consistente en todo el sistema, siguiendo estándares y convenciones de diseño reconocidos para facilitar la comprensión y el uso intuitivo.
5. Prevención de errores	Es preferible diseñar interfaces que eviten los errores en lugar de depender de la detección y corrección de los mismos. Se deben incluir barreras para prevenir acciones no deseadas.
6. Reconocimiento antes que recuerdo	El diseño de la interfaz debe minimizar la carga de memoria del usuario, presentando información y opciones de manera clara y visible en lugar de requerir que los usuarios recuerden información anteriormente proporcionada.
7. Flexibilidad y eficiencia de uso	La interfaz debe ser flexible y permitir que los usuarios experimentados realicen acciones de forma rápida y eficiente, proporcionando atajos y opciones para agilizar las tareas comunes.
8. Diseño estético y minimalista	Una interfaz visualmente atractiva y libre de elementos innecesarios mejora la experiencia del usuario al facilitar la comprensión y el enfoque en la tarea principal.
9. Ayuda y documentación	Siempre que sea necesario, se deben proporcionar ayudas contextuales y documentación clara para que los usuarios puedan comprender y resolver problemas en la interfaz.
10. Reconocimiento, diagnóstico y recuperación de errores	La interfaz debe ser diseñada para que los usuarios puedan reconocer, diagnosticar y recuperarse de errores de forma rápida y sencilla, ofreciendo mensajes de error claros y opciones de solución.

Tabla 2.1: "10 Usability Heuristics for User Interface Design" by Nielsen [7]

Otro punto a considerar para una mejor experiencia de usuario, es el usuario que tiene por objetivo el software, que en este proyecto se considera a las personas de edad adulta intermedia, por lo que se podría deducir que son usuarios que normalmente cuentan con menor conocimiento/manejo de software. Un artículo que podría ayudar a lidiar con esto es el realizado por *Ali Darejeh* y *Dalbir Singh* [8], donde realizan una revisión de cómo se podría aumentar la usabilidad del software para usuarios con menor conocimiento computacional. En la ***Tabla 2.2*** se puede apreciar un resumen que podría ayudar a mejorar la experiencia de este tipo de usuario específico.

Problema	Solución de diseño UI
Falta de conocimiento computacional	● Colocar una guía de usuario básica y ayuda en el software. ● Reducción de terminología.
Cambios cognitivos	● Reducir el desorden en la pantalla. ● Rutas de navegación claras y sencillas. ● Usar funciones similares para realizar diferentes tareas.
Dificultad para memorizar	● Colocar textos descriptivos y guías de herramientas.

Tabla 2.2: Usuarios mayores, problemas y soluciones [8].

También existen otros principios como pautas universales para guiar la creación del diseño del software. Entendiendo estas directrices es posible crear interfaces digitales más atractivas y cumplir con los propósitos del software en cuestión. Por ejemplo, estos principios podrían resumir en grandes rasgos las principales pautas de diseño [9]:

- **Alineación**: Se refiere a la colocación y disposición de los elementos de manera ordenada, logrando una estructura coherente y una jerarquía visual clara.
- **Consistencia**: Utilización de elementos visuales de manera recurrente a lo largo del diseño, estableciendo una coherencia entre las pantallas y unificando la apariencia visual.
- **Jerarquía**: Organización visual de los elementos de acuerdo a su importancia.

- **Contraste**: Diferenciación entre los elementos en función de sus características visuales, como el tamaño, color, forma, etc. Esto permite destacar los elementos importantes y mejorar la legibilidad y comprensión de la información.
- **Espacio Negativo**: Utilización intencional del espacio en blanco, mejorando la legibilidad y destacando elementos importantes.
- **Escala**: Proporción de los elementos y distribución de estos en la pantalla. Permitiendo determinar la importancia y el nivel de atención que requiere.

3. Análisis

Esta sección tiene como objetivo analizar el Software CareForMe en su versión actualizada, tanto en el ámbito de sus requerimientos funcionales como en los no funcionales. Esto permite una mejor comprensión en las características que debe tener el software y en general una visión menos técnica.

3.1. Descripción del Software

CareForMe es un software web dirigido principalmente a cuidadores informales con el objetivo de brindar información relevante sobre temas de cuidado de personas mayores con demencia. Además la plataforma permite distintas funcionalidades extras para apoyar en la labor de los cuidados en el día a día.

Esta plataforma en su versión inicial permite a los usuarios cuidadores ver información relevante sobre distintos temas asociados a su labor, igualmente estos pueden registrar sus datos y realizar test neuropsiquiátricos con el fin de medir la sobrecarga. Además, posibilita el acceso de usuarios de tipo personal de salud con el objetivo de acceder a la información (historial de tests, datos personales) de todos los cuidadores registrados en la plataforma.

Ahora en su nueva versión, además de las funcionalidades descritas anteriormente, esta permitirá la asociación personal entre un usuario cuidador y usuario personal de salud. De esta manera, el cuidador podrá realizar preguntas, registrar datos del paciente y eventos anormales (cambios en el sueño, alimentación, caídas, etc).

Del mismo modo, el personal de salud podrá acceder a las preguntas correspondientes al cuidador asociado para ser respondidas. Igualmente, este podrá acceder a la información del cuidador, donde además tendrá la posibilidad de ver esta información en gráficos.

Esto les brindará a los usuarios una plataforma más completa y atención personalizada.

La plataforma web podrá ser accedida a través de cualquier dispositivo con un navegador web, aunque se recomienda el uso desde una computadora.

3.2. Requerimientos

Para la descripción de los requerimientos del software es necesario especificar los tres distintos usuarios con los que cuenta la plataforma. A continuación se describirán brevemente:

- **Cuidador:** Es aquel actor principal al cual está enfocado el software, es necesario que se registre.
- **Personal de Salud:** Es aquel actor que se relaciona con el actor cuidador para visualizar su información. Es necesario que se registre.
- **Administrador:** Es aquel actor encargado de administrar y actualizar el contenido de la plataforma. Es creado en el momento del desarrollo del software.

3.2.1. Requerimientos Funcionales

La plataforma permitirá...	Cuidador	Personal de salud	Administrador
Registrarse con los datos del usuario	☑	☑	☐
Acceder con el nombre de usuario y contraseña	☑	☑	☑
Cambiar la contraseña	☑	☑	☑
Recuperar la contraseña con el correo electrónico en caso de olvidarla	☑	☑	☐
Visualizar contenido con respecto a los cuidados necesarios para el paciente con filtrado por palabra clave.	☑	☐	☐

La plataforma permitirá...	Cuidador	Personal de salud	Administrador
Registrar datos del paciente.	☑	☐	☐
Editar el perfil propio y del paciente.	☑	☐	☐
Realizar preguntas al personal de salud asociado.	☑	☐	☐
Visualizar las preguntas y respuestas correspondientes.	☑	☑	☐
Responder las preguntas hechas por el cuidador asociado.	☐	☑	☐
Registrar eventos o situaciones anormales del paciente.	☑	☐	☐
Visualizar tabla con los eventos registrados.	☑	☑	☐
Realizar test neuropsiquiátricos (NPI-Q y ZARIT).	☑	☐	☐
Visualizar tabla de tests realizados.	☑	☑	☐
Enlazar con uno o más usuarios cuidadores para visualizar su información.	☐	☑	☐
Visualizar detalles de los tests realizados y eventos registrados.	☐	☑	☐
Visualizar gráficos históricos de los tests realizados y eventos registrados.	☐	☑	☐
Actualizar contenido disponible.	☐	☐	☑

Tabla 3.1: Requerimientos funcionales y usuarios correspondientes

Casos de Uso:

Para representar los requerimientos funcionales y una ilustración desde la perspectiva del "actor" o usuario, se puede utilizar un diagrama de casos de uso (UCD), ya que provee un resumen del sistema con fácil comprensión y además permite obtener retroalimentación por parte del cliente o los usuarios finales [10]. En la ***Figura 3.1*** se puede apreciar el diagrama de casos de uso actualizado para la versión de este software.

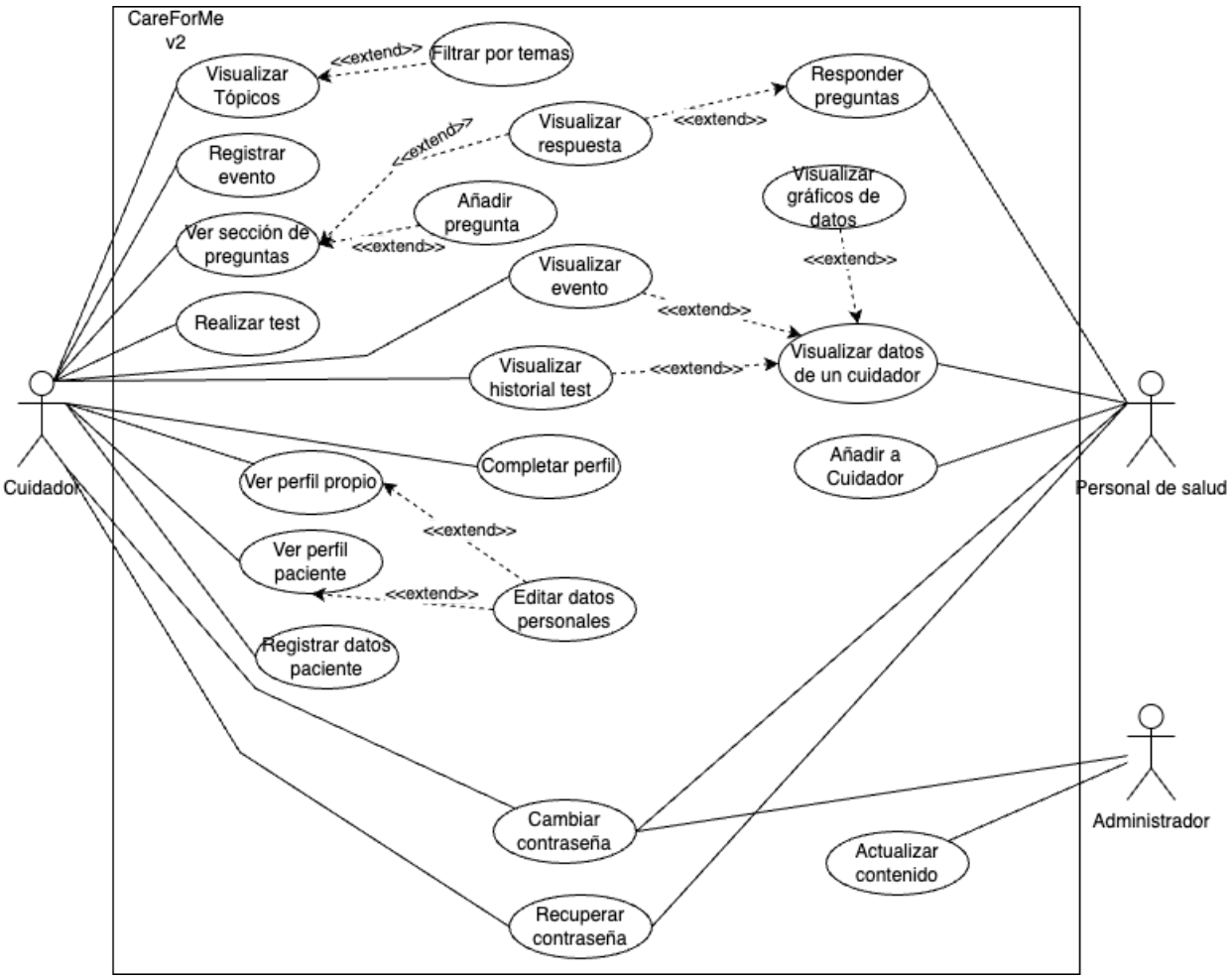

Figura 3.1: Diagrama de casos de uso CareForMe versión actualizada

3.2.2. Requerimientos No Funcionales

- **Rendimiento:** El sistema debe ser capaz de soportar una carga de usuarios en simultáneo sin verse afectado significativamente por esto. Teniendo un tiempo de respuesta inferior a dos segundos.
- **Seguridad:** El software debe cumplir con estándares de seguridad mínimos. Por ejemplo, la autenticación con credenciales únicas. Con el objetivo de proteger los datos de los usuarios.
- **Usabilidad:** El sistema debe estar enfocado en la experiencia de usuario objetivo, con el fin de brindar una navegación intuitiva y sencilla.

- **Portabilidad:** La plataforma debe ser accedida tanto desde un computador como de un smartphone.
- **Consistencia:** El sistema debe mantener la integridad de la información de los usuarios.
- **Escalabilidad:** Como se espera que el sistema reciba actualizaciones durante el tiempo para mejorar y un aumento en su carga de trabajo, debe mantener la capacidad de adaptarse a estos cambios de una manera sencilla.

4. Diseño y Arquitectura

Debido a que la escalabilidad es uno de los puntos fuertes de Django [11], se decidió continuar con el desarrollo en base a la arquitectura base de este framework.

Django se basa en la arquitectura MVC (Model-View-Controller)[12], la cual denomina como MTV (Model-Template-View) donde el equivalente entre estos es que el Controller es el "View" y el View es el "Template". Si bien es algo que podría ser confuso, la explicación breve es solo un cambio en la terminología, ya que para Django la "View" es la función de devolución de una llamada de Python para una URL en particular, debido a que esta se encarga de decidir qué datos son los devueltos y no cómo son mostrados estos datos. Para el caso de "Template", al ser la vista, es quien muestra los datos [11].

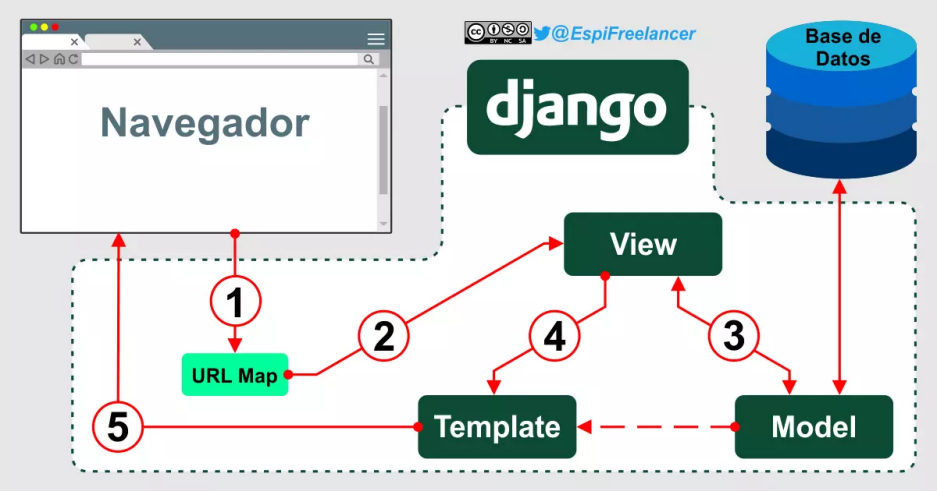

Figura 4.1: Diagrama arquitectura Django [13]

A continuación se describirán brevemente cómo se utilizan los componentes claves de la arquitectura que se puede apreciar en la ***Figura 4.1***:

- **URLs**: El sistema basado en URLs asocia las solicitudes de los usuarios con las vistas correspondientes. Estas se configuran en un archivo de enrutamiento, donde se mapea cada URL a la vista que le corresponde. Esto permite tener rutas limpias, facilitando la navegación tanto dentro de la aplicación como en el código.
- **View**: Las vistas son responsables de procesar las solicitudes de los usuarios y generar las respuestas. Se utilizó las vistas basadas en clases, que permite reutilizar la funcionalidad común y facilitar la organización del código. Estas interactúan con los modelos para recuperar y manipular los datos que sean necesarios, para luego renderizar las plantillas correspondientes y presentar la información al usuario.
- **Model**: Se utilizan los modelos Django para definir la estructura y la lógica de los datos. Estos representan las entidades del dominio y definen las tablas de la base de datos, así como las relaciones y restricciones entre ellas.
- **Template**: Con el fin de separar la lógica del código, las plantillas permiten definir la estructura y el diseño de las páginas web, así como insertar datos dinámicos provenientes de las vistas. Esto permite una mejora en la mantenibilidad y legibilidad del código, además de facilitar la personalización y adaptación de la interfaz de usuario.

Además, cabe mencionar que en la ***Figura 4.1*** los números determinan el orden en el que ocurren las peticiones entre los componentes. Por ejemplo, desde el 1 donde el usuario accede a una url, hasta el 5 donde el sistema despliega la respuesta al usuario en el navegador.

5. Desarrollo

Para el desarrollo incremental del proyecto y la organización, se decidió aplicar la metodología ágil llamada Scrum [14], pero de una manera un poco más flexible. De tal manera que se adoptaron Sprints de cuatro semanas, donde en cada una de ellas se realizó una reunión con el profesor patrocinante *Geoffrey Hecht*, con el fin de ir mostrando los avances correspondientes. Luego, al final de cada Sprint se efectuó una reunión con el

cliente, con el fin de obtener la retroalimentación necesaria para efectuar cambios en los objetivos de cada uno de los siguientes Sprints.

En cuanto al desarrollo del código como tal, se tomó el repositorio base de la plataforma [15] para aplicar las actualizaciones y desarrollar las nuevas funcionalidades. El nuevo código se puede encontrar en el GitHub *acotal22* [16].

5.1. Tecnologías Utilizadas

En este apartado, se listan las principales tecnologías y herramientas utilizadas durante todo el desarrollo del proyecto, con el fin de detallar y explicar brevemente la relevancia de cada una de ellas en este caso concreto:

- **Django**:
  - ¿Qué es?: Es un framework de desarrollo web de alto nivel que se basa en Python. Permite construir aplicaciones web robustas y escalables.
  - Relevancia: Al ser el framework utilizado en el desarrollo base de la plataforma, fue relevante para escalar este código de una manera más sencilla. Además permitió realizar un cambio en la base de datos sin mayor complicaciones (de SQLite a MySQL para el despliegue en Heroku).
- **Bootstrap**:
  - ¿Qué es?: Es una biblioteca de front-end que proporciona un conjunto de componentes y herramientas para el desarrollo de interfaces web que pueden ser responsivas y visualmente atractivas.
  - Relevancia: Debido a la facilidad para la utilización de la biblioteca, principalmente permitió acelerar el proceso de desarrollo de cada una de las interfaces.

- **Heroku**:
  - ¿Qué es?: Es una plataforma basada en la nube que permite la implementación, gestión y escalabilidad de aplicaciones web. También permite un entorno de despliegue sencillo y automatizado, facilitando el despliegue de una aplicación.

    - Relevancia: Permitió el despliegue de la plataforma en una versión de prueba.
- **GitHub**:
    - ¿Qué es?: Es una plataforma de alojamiento de repositorios de código fuente basada en Git. Es ampliamente utilizado para el control de versiones, colaboración y seguimiento de problemas.
    - Relevancia: Al estar alojado el código fuente base de la plataforma, se utilizó para realizar una copia del repositorio en uno propio. Asimismo, realizar el control de versiones e ir subiendo los avances progresivos del código.
- **Jira:**
    - ¿Qué es?: Herramienta para la gestión de proyectos, permitiendo planificar, organizar y asignar tareas, realizar seguimientos del progreso y facilitando la comunicación entre el equipo de trabajo.
    - Relevancia: En este caso solo se utilizó la función de crear y organizar estas tareas para cada Sprint del desarrollo.
- **MySQL Workbench:**
    - ¿Qué es?: Es una herramienta de administración y diseño de base de datos MySQL. Proporciona una interfaz gráfica para administrar esquemas, crear y modificar tablas.
    - Relevancia: Principalmente fue utilizada para crear el esquema relacional de la base de datos.
- **DB Browser for SQLite:**
    - ¿Qué es?: Es una herramienta de administración y diseño de bases de datos SQLite. Permite crear, modificar y consultar bases de datos SQLite de manera visual.
    - Relevancia: Utilizada durante el desarrollo del proyecto Django para visualizar los datos y las tablas, así como también modificar, eliminar o agregar datos con el fin de realizar pruebas.
- **Figma**:
    - ¿Qué es?: Es un editor de gráficos e interfaces, que permite la generación de prototipos, principalmente en aplicaciones web.

- Relevancia: Se utilizó en el principio del desarrollo, con el fin de proporcionar una idea o prototipo de los cambios en la interfaz de una manera más rápida. Así obteniendo una retroalimentación para realizar las correcciones pertinentes.

- **Robot Framework:**
  - ¿Qué es?: Es un framework para la automatización de pruebas de aceptación o validación. Se caracteriza por su legibilidad, modularidad y facilidad de uso.
  - Relevancia: Con el fin de realizar pruebas de una manera más eficiente y automatizada, se utilizó este framework para validar el correcto funcionamiento de la plataforma.

5.2. Base de Datos

Debido a que Django soporta distintas bases de datos, tales como MariaDB, MySQL, SQLite, etc [17] y el cambio entre estas es algo relativamente sencillo de realizar, se decidió continuar el desarrollo del proyecto con la base de datos relacional SQLite en su versión 3. Sin embargo, en el despliegue de la aplicación fue cambiada por MySQL, debido a que Heroku no permite los cambios de manera persistente en una base de datos SQLite para entornos de producción [18].

Para representar los cambios que se realizaron en la base de datos, se modificó el diagrama entidad-relación del proyecto base. Este diagrama permite visualizar una representación de la estructura de la base de datos y cómo se relacionan las entidades. Permitiendo ayudar a comprender mejor la organización y el funcionamiento de la base de datos [19].
El diagrama de la ***Figura 5.1*** fue realizado con la herramienta MySQL Workbench de forma automática desde la base de datos en Heroku, posteriormente modificado para dejar las entidades relevantes (dejando de lado las que crea Django por defecto) para el diagrama.

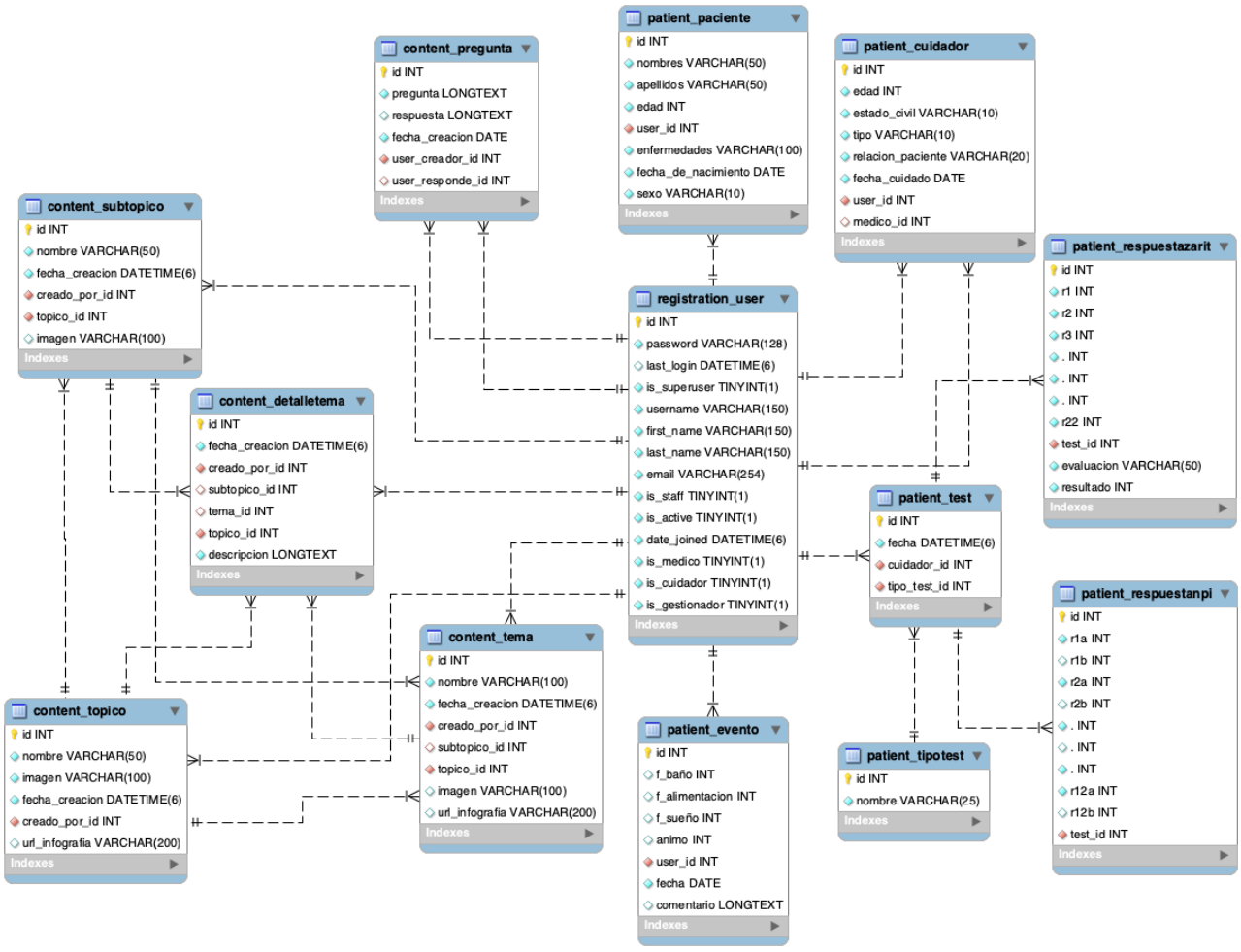

Figura 5.1: Diagrama ER base de datos

Cabe señalar que los nombres de las entidades se dejaron por defecto a cómo se organizan en Django, para que así sea más fácil comprender y realizar la comparación con el código fuente. También mencionar que en las entidades "*patient_respuestazarit"* y *"patient_respuestasnpi"* hay puntos en los atributos con el fin de reducir gráficamente las entidades, pero estos corresponden a la continuación de los datos.

Los cambios principales realizados en la estructura de la base de datos se pueden ver reflejados en las entidades "*content_pregunta*", "*patient_paciente*" y "*patient_evento*". En cuanto a las relaciones, ahora la entidad "*patient_cuidador*" posee una relación adicional con

la entidad "*registration_user*" pero con el tipo de usuario "*médico*". A continuación se describirán las entidades y sus relaciones correspondientes:

- **content_tema:** Entidad encargada de almacenar los nombres de los temas. Puede ser editada solo por el usuario administrador.
    - Relaciones:
        - creado_por_id: Hace referencia al usuario que ha creado el tema.
        - subtopico_id: Se relaciona con los subtópicos que pertenecen al tema.
        - topico_id: Se relaciona con los tópicos que pertenecen al tema.
- **content_detalletema:** Es la entidad encargada de almacenar los detalles de los tópicos, subtópicos o el tema relacionados. Puede ser editada solo por el usuario administrador.
    - Relaciones:
        - creado_por_id: Hace referencia al usuario que ha creado el detalletema.
        - subtopico_id: Hace referencia a los subtópicos que tienen el detalle de este mismo.
        - tema_id: Hace referencia a los temas que pertenece el detalle.
        - topico_id: Se relaciona con los tópicos y los detalles de estos mismos.
- **content_topico:** Encargada de la información de un tópico en concreto. Puede ser editada solo por el usuario administrador.
    - Relaciones:
        - creado_por_id: Relaciona al usuario que ha creado el tópico.

- **content_subtopico**: Entidad que almacena los subtópicos de la plataforma. Puede ser editada solo por el usuario administrador.
    - Relaciones:
        - creado_por_id: Usuario que ha creado el subtópico.
        - topico_id: Hace referencia al tópico que pertenece.

- **content_pregunta:** Entidad encargada de almacenar las preguntas que realiza el usuario *cuidador*, así como también la respuesta respectiva hecha por el usuario *personal de salud*.
    - Relaciones:
        - user_creador_id: Usuario que crea la pregunta.
        - user_responde_id: Usuario que crea la respuesta a la pregunta.
- **registration_user**: Entidad que almacena la información de los usuarios que son registrados en la plataforma. Un dato importante es que tiene atributos para identificar el tipo de usuario al que corresponde (is_medico, is_cuidador, is_gestionador, etc).

- **patient_paciente:** Almacena la información propia de un usuario de tipo paciente. Solo el usuario cuidador puede editar esta entidad.
    - Relaciones:
        - user_id: Hace referencia al usuario de tipo cuidador al que pertenece el paciente.

- **patient_cuidador**: Almacena la información propia de un usuario de tipo cuidador.
    - Relaciones:
        - user_id: Esta relación permite enlazar con el usuario de la plataforma por el cual será identificado con las credenciales, correo, etc.
        - medico_id: Relaciona el usuario cuidador con un usuario de tipo médico (*personal de salud*).
- **patient_tipotest:** Entidad para identificar el tipo de test a realizar.
- **patient_test:** Almacena los test realizados por el usuario *cuidador*.
    - Relaciones:
        - cuidador_id: Referencia al cuidador que ha realizado este test.
        - tipo_test_id: Determina si el test es de tipo NPI-Q o ZARIT.

- **patient_respuestazarit, patient_respuestanpi:** Estas entidades son las encargadas de guardar las respuestas de los test correspondientes.

    - Relaciones:
        - test_id: Referencia a la entidad test correspondiente.
- **patient_evento:** Es la entidad que almacena la información de un evento ocurrido al paciente. Solo puede ser editada por el usuario cuidador.
    - Relaciones:
        - user_id: Hace referencia al usuario cuidador que ha registrado el evento.

5.3. CareForMe V2

El propósito de esta sección es especificar en mayor detalle los cambios realizados a la plataforma CareForMe, desde la estructura de los archivos específica de este proyecto Django hasta los detalles de interfaz y nuevas funcionalidades. Además, con esto se busca detallar el desarrollo del software con el fin de que sea entendido para un posible desarrollo a futuro teniendo en cuenta estos cambios.

5.3.1. Estructura de Archivos en Proyectos Django

Para explicar más detalladamente la nueva plataforma, se comenzará por describir la estructura de archivos de Django para este caso en particular.

Como bien se sabe, en Django el proyecto se puede dividir en distintas "Apps" [20] con el fin de delegar las responsabilidades y de esta forma tener un proyecto escalable en cuanto a añadir nuevas funcionalidades o quitar algunas de ellas.

Por esta razón, en la ***Figura 5.2*** se puede apreciar la estructura de los archivos principales que tiene el proyecto, con el fin de facilitar la comprensión de las aplicaciones y sus respectivas funcionalidades.

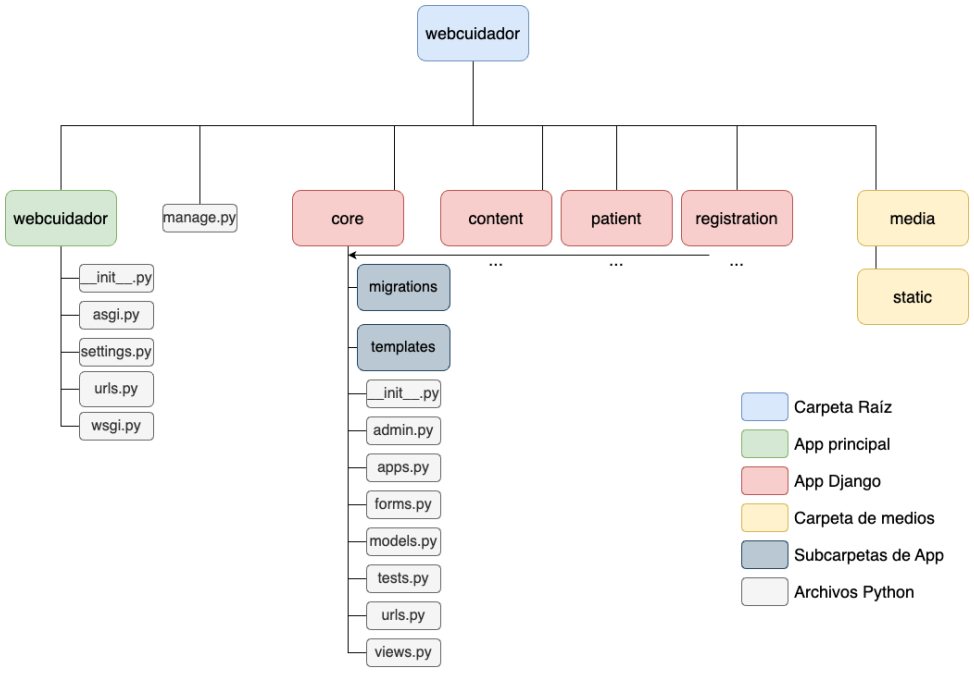

Figura 5.2: Estructura de archivos del proyecto

En términos generales cada una de las aplicaciones posee las subcarpetas y archivos python que se visualizan en la ***Figura 5.2***, donde además se pueden observar archivos principales que son la base de la arquitectura MVT, como "*models.py", "views.py",* y la carpeta "*templates"* posee los archivos html que son las plantillas de la arquitectura.

Dicho esto, a continuación se explicarán brevemente las aplicaciones y sus funcionalidades principales en el proyecto:

- **Core:** Es la aplicación que como su nombre indica, es el núcleo o base de las otras aplicaciones. Ya que las demás aplicaciones heredan características de esta. También es la encargada de manejar las vistas estáticas como la portada o la barra de navegación.
- **Content:** Es una de las aplicaciones más importantes o con más funcionalidades del proyecto. Principalmente se encarga de manipular todo el contenido del usuario *cuidador*, tal como los tópicos, subtópicos, consejos, preguntas y tests. Para el caso del usuario *personal de salud*, se responsabiliza en la lista de los usuarios cuidadores, preguntas, informe gráfico y tests realizados.
- **Patient:** Su función principal radica en las funcionalidades para el usuario cuidador, tales como el registro y edición del perfil propio y del paciente, añadir eventos, tests realizados y la realización de los tests.

- **Registration:** Como su nombre lo dice, se encarga de administrar los registros de usuarios, así como también los inicios de sesión, cambio de contraseña y recuperación de contraseña.

5.3.2. Actualización de Interfaz

Con el objetivo de mejorar la experiencia de usuario a través de una interfaz más atractiva, intuitiva y simple, se han realizado cambios en la interfaz del software. A través de una revisión del estado del arte (se describe en el apartado **2**) y un enfoque centrado en el usuario final, se han introducido nuevas funcionalidades y optimizado la navegación para mejorar la usabilidad en general.

En este apartado se presentará en detalle los cambios más relevantes en el ámbito de la interfaz, destacando los beneficios que aportan a los usuarios y al sistema en su conjunto.

- **Página de: Bienvenida**

Esta interfaz corresponde a la primera vista una vez que se ingresa a la plataforma, por lo que mantener un mensaje de bienvenida y además una breve descripción es relevante.

En este caso, se quitó un botón (registrarse) con el fin de reducir las opciones y mantener la atención en lo relevante. También se decidió colocar una imagen de fondo para hacer una representación sobre el objetivo principal de la plataforma.

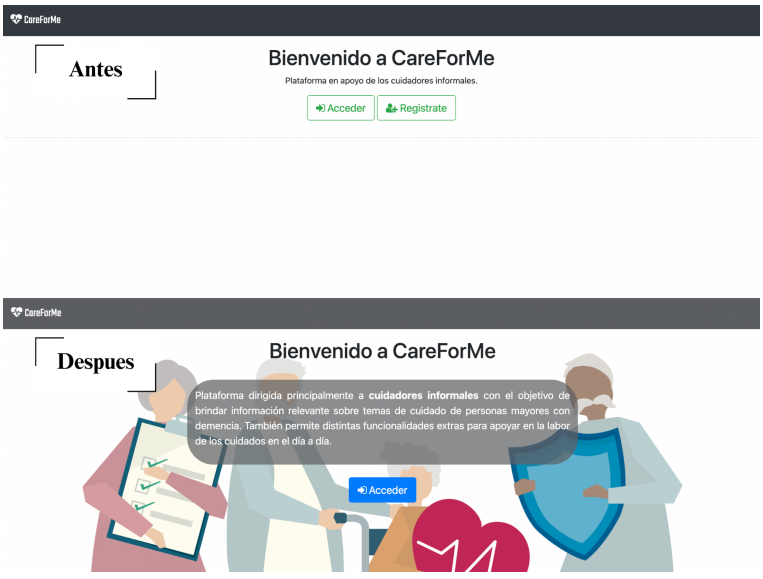

Figura 5.3: Antes y después de la página de Bienvenida

- **Página de: Acceso, Registro, Editar Perfil y Cambiar Contraseña**

  En estas páginas se encontró un problema relevante en común con respecto al contraste de los campos de los formularios, ya que estos no tenían diferencia en los colores con el fondo, por lo que podría dificultar la visualización o no mantener la atención en los elementos importantes. Para solucionar esto, se agregó un fondo oscuro en los campos de ingreso, de esta manera se logra un **contraste** y **jerarquía** en los elementos.

  Además, en la pantalla de **Acceso** se agregaron dos funcionalidades importantes que se encuentran marcadas en rojo en la ***Figura 5.4***, la primera permite visualizar la contraseña y la otra permite recuperar la contraseña en caso de ser olvidada.

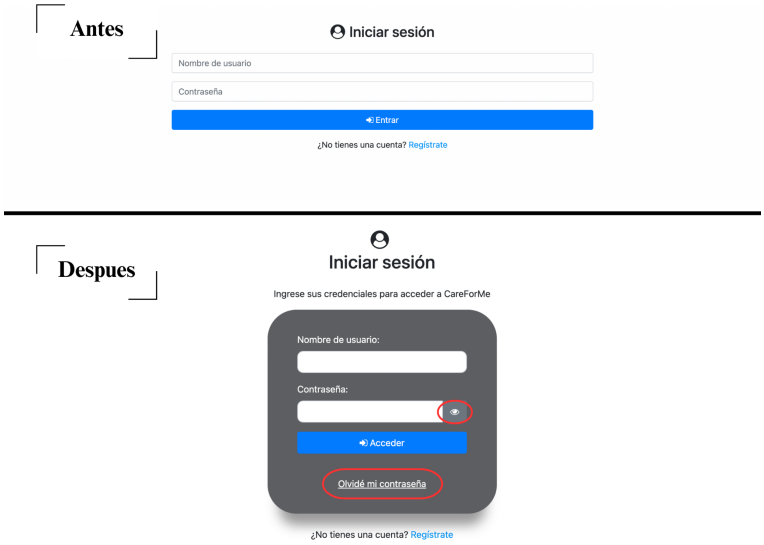

Figura 5.4: Antes y después de la página de Acceso

Para la pantalla de **Registro** también se encuentra el problema del exceso de texto, lo cual puede afectar en el orden de los elementos en la pantalla y la sobrecarga de estos. Por lo mismo, se emplea la solución que se puede observar en las ***Figuras 5.5 y 5.7*** , la cual corresponde a los textos de ayuda colocados en un elemento que solo es visible en el caso de pasar el ratón sobre el ícono, de esta forma se reduce el texto inicial en la pantalla.

Figura 5.5: Antes y después de la página de Registro

Para el manejo de los errores en los formularios, se puede observar que se cambiaron a un elemento que aparece en el campo correspondiente con el fin de informar de manera más precisa. Cabe destacar que esto fue aplicado a todos los errores de este tipo en todas las pantallas.

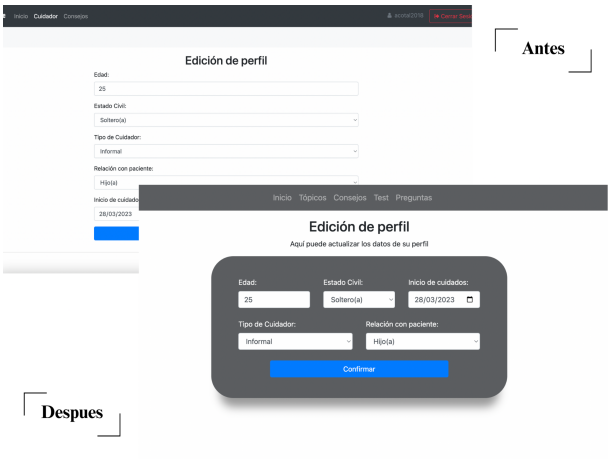

Figura 5.6: Antes y después de la página de Editar Perfil

Figura 5.7: Antes y después de la página de Cambiar Contraseña

- **Página de: Inicio, Tópicos y Consejos**

Para estas páginas se encontró un problema de **navegación**, debido a que por ejemplo en la ***Figura 5.8*** se pueden observar cómo los enlaces a otras pantallas están ubicados a la izquierda de la barra de navegación (elemento con fondo color gris en la parte superior de la pantalla) y con letra pequeña. Asimismo, el nombre del usuario posee un botón a su derecha que podría llamar la atención aún más que los otros elementos de la barra de navegación.
Pensando en los usuarios objetivos, se dedujo que lo anterior podría dificultar la navegación correcta o la facilidad para lograr ingresar a las distintas secciones.

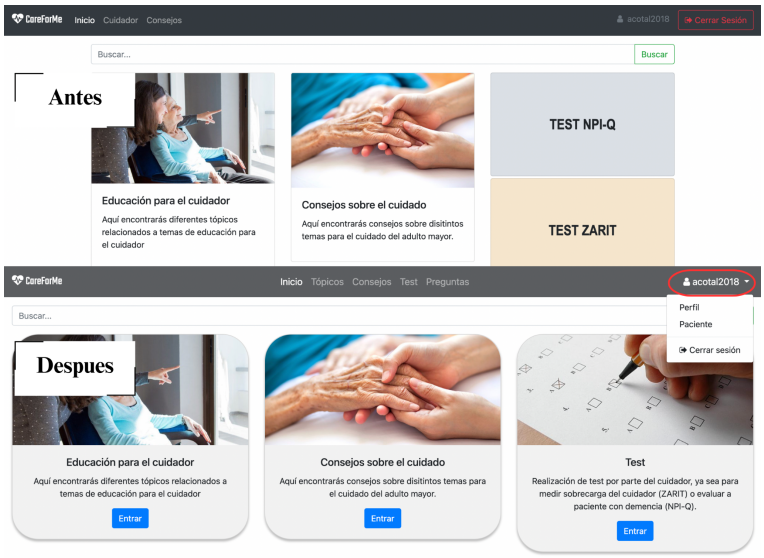

Figura 5.8: Antes y después de la página de Inicio

Ahora, como se puede observar en la ***Figura 5.8***, los enlaces se encuentran en el centro de la barra de navegación y con una letra más grande, de esta manera posee una **alineación** más clara y facilita el cambio entre las distintas secciones. Además, al nombre de usuario se le da una funcionalidad con el fin de quitar el botón de "cerrar sesión" de la barra y agregar el acceso a los perfiles (propio y del paciente). Incluso para mantener la **consistencia** y **escala** en la interfaz, se agregó un apartado específico para los test, quedando tres elementos (Educación para el cuidador, Consejos sobre el cuidado y Test) en la pantalla de **Inicio**, los cuales ahora ocupan todo el ancho de la pantalla y cada uno cuenta con su texto descriptivo.

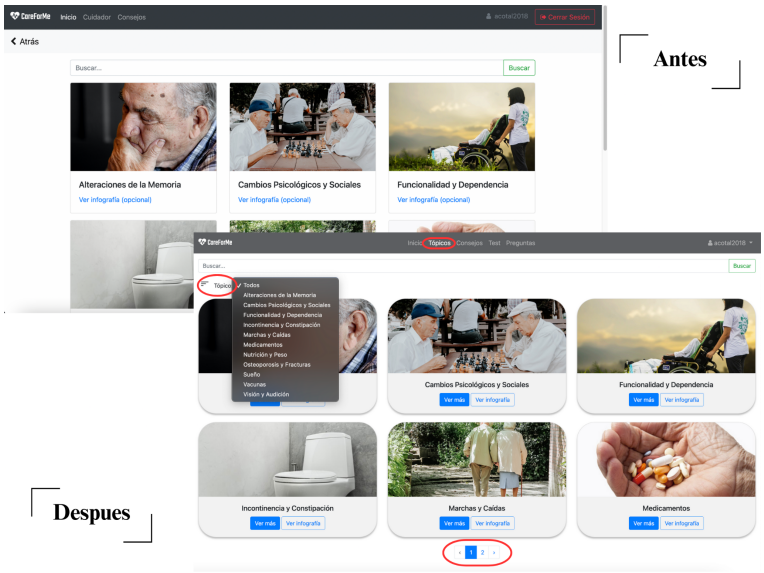

Figura 5.9: Antes y después de la página de Tópicos

Igualmente, para mejorar la **navegación** en las páginas de **Tópicos** (***Figura 5.9***) y **Consejos** (***Figura 5.10***) se añadió la paginación, ya que esta permite mostrar menos elementos por página y así poder navegar menos "hacia abajo", pero en la primera también se añadió un filtro por cada tema con el mismo fin.

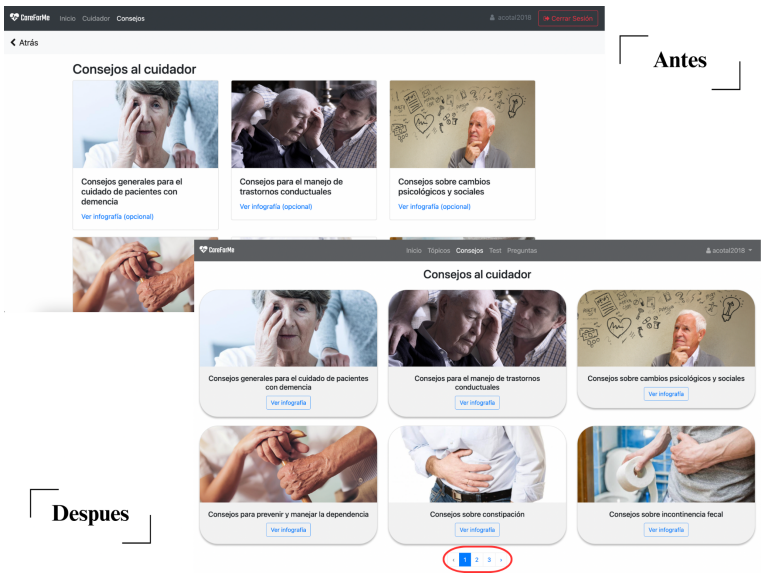

Figura 5.10: Antes y después de la página de Consejos

- **Página de: Perfil**

  En esta página el principal problema radica en que los elementos no poseen una **alineación** correcta, además de que hay funcionalidades que no tienen **coherencia** con el elemento principal (botones de realizar test).

La solución a lo anterior, que se puede apreciar en la ***Figura 5.11***, se basa en los principios de **alineación**, **consistencia** y **jerarquía**, ya que los datos del usuario son alineados en un elemento consistente con los demás elementos en la pantalla (por su forma). Además, al quitar la tabla de "test realizados" permite ordenar los elementos de manera más jerarquizada, pudiendo acceder a esta haciendo click en el elemento específico.

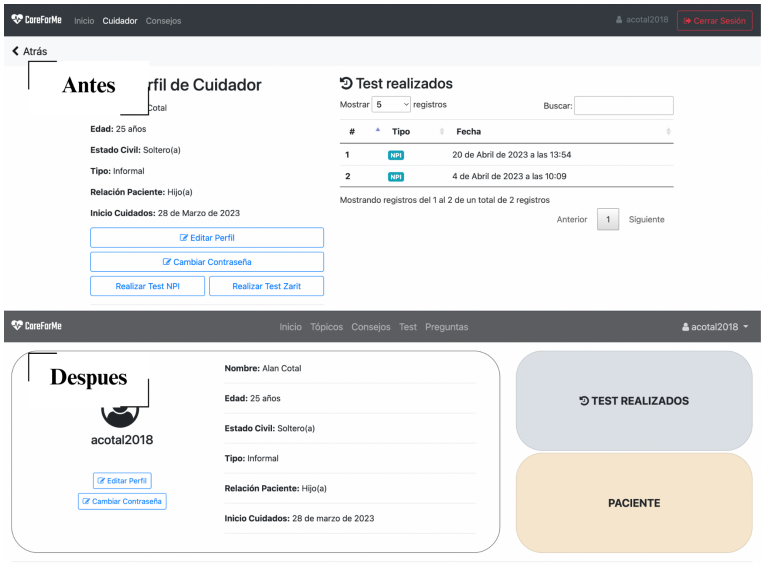

Figura 5.11: Antes y después de la página de Perfil

5.3.3. Nuevas Funcionalidades

En este capítulo se brindarán los detalles de las nuevas funcionalidades que satisfacen a los requerimientos del software que se detalló anteriormente en el capítulo **3.2**.

- **Registro de datos del paciente:**

  El **objetivo** de esta nueva funcionalidad es que se almacenen en la plataforma los datos del paciente para que el usuario "médico" tenga mayores detalles del paciente asociado a su cuidador y así poder brindar una atención más personalizada.

  ¿Cómo funciona?. Una vez que el usuario "cuidador" se haya registrado e iniciado sesión en la plataforma, podrá registrar los datos del paciente en la sección "Paciente", que se puede ver en la ***Figura 5.12***, solo si no lo ha registrado con anterioridad.

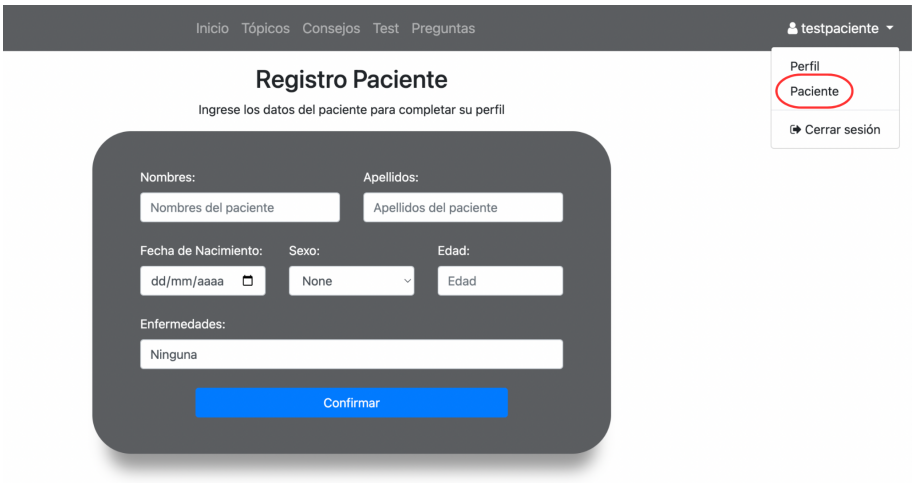

Figura 5.12: Pantalla de registro del paciente

Una vez registrado los datos, estos podrán ser editados en la página del paciente (ver ***Figura 5.16***). Cabe mencionar que la interfaz será igual a la del registro pero los datos ya estarán cada uno en sus respectivos campos.

- **Asociación entre usuario "médico" y "cuidador":**

  El **objetivo** principal de esta característica es proporcionar una atención más personalizada del "médico" hacia el "cuidador", ya que ahora el primero podrá agregar a los cuidadores de la plataforma que no estén a cargo de un médico. De esta manera, solo podrá acceder a la información de los cuidadores que decida mantener en su lista y no la de todos los que estén registrados en la plataforma. Asimismo, no podrá acceder a la información de los cuidadores que estén a cargo de otro médico.

  ¿Cómo funciona?. Cuando el usuario "médico" inicie sesión en la plataforma, lo llevará a la página de inicio, donde podrá observar la lista de cuidadores que tiene a su cargo y un botón en la parte superior derecha de esta tabla (ver ***Figura 5.13***). Este botón le permitirá acceder a otra pantalla donde aparecen los cuidadores disponibles en una tabla con la respectiva información y un botón para realizar la asociación (previa confirmación). Una vez realizado esto, se podrá observar que se encuentra un nuevo cuidador en la lista del inicio y podrá acceder a la información con un botón o bien eliminar de la lista con otro botón.

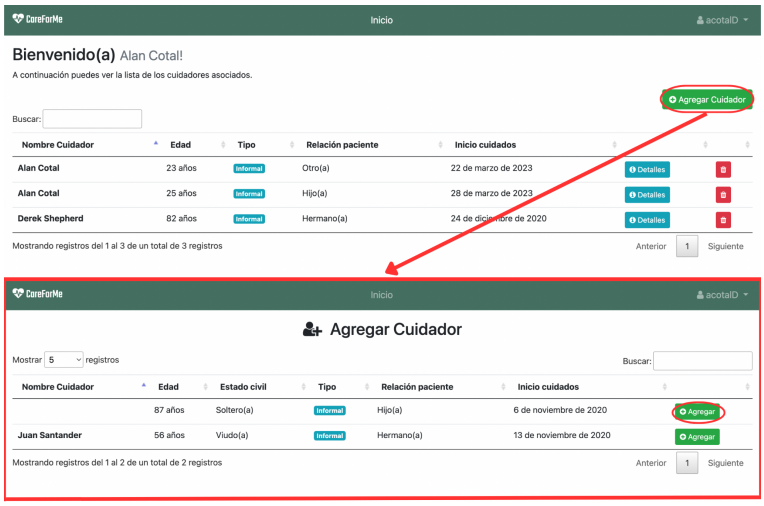

Figura 5.13: Pantallas de acción "Agregar Cuidador"

- **Sección de preguntas:**

  El **objetivo** de esta funcionalidad es proporcionar la oportunidad de que el usuario "cuidador" pueda realizar consultas al "médico" cuando lo requiera, y este último responda a esas consultas. Esto permite que se puedan aclarar dudas sin la necesidad de asistir personalmente a una cita médica.

  ¿Cómo funciona?. Para realizar consultas, el "cuidador" deberá acceder a la sección **Preguntas**, donde se verá similar a la de la ***Figura 5.14***. En esta pantalla, podrá visualizar todas las preguntas que ha realizado anteriormente, pudiendo filtrarlas por fecha, estado de respuesta o buscar por palabra clave. Para agregar una nueva pregunta, el cuidador deberá presionar el botón "Añadir Pregunta", el cual abrirá un elemento donde podrá ingresar esta pregunta.

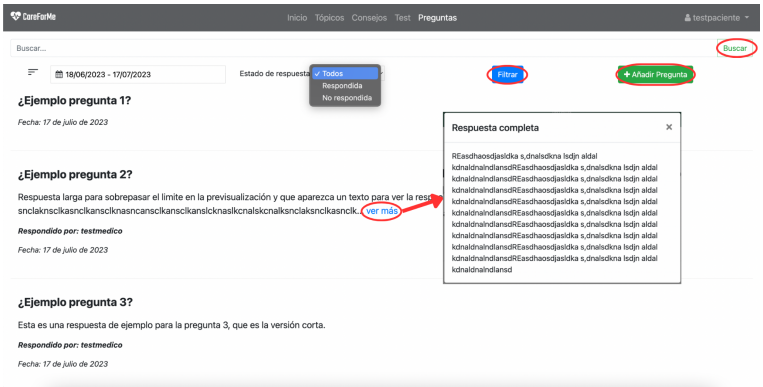

Figura 5.14: Pantalla de sección pregunta desde la vista del cuidador

Para responder las preguntas, el usuario “médico” debe ingresar al apartado **Preguntas** que puede encontrar en el detalle de los cuidadores asociados (ver ***Figura 5.17***). Acá visualizará las preguntas de manera similar (ver ***Figura 5.15***) a la vista del cuidador, pero con la diferencia de que las preguntas que no ha respondido tendrán un botón específico para hacerlo.

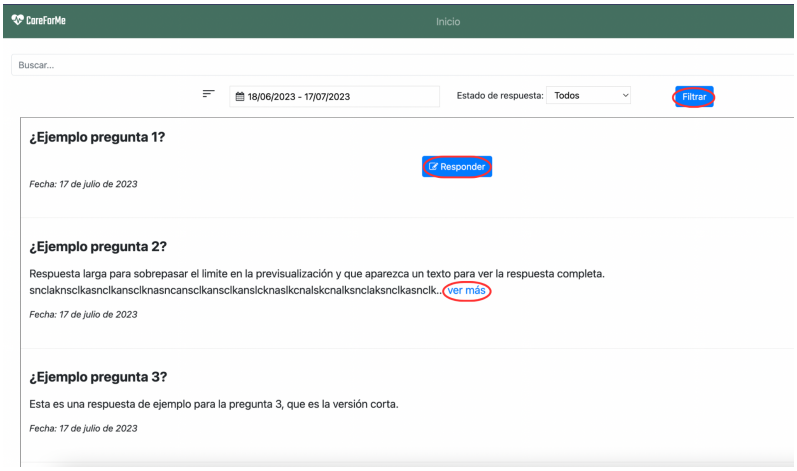

Figura 5.15: Pantalla de sección pregunta desde la vista del médico

- **Registro de eventos:**

  El **objetivo** de esto es llevar un registro a través del tiempo sobre eventos o situaciones no comunes en los pacientes, cómo por ejemplo: aumento en las veces que asiste al baño a hacer sus necesidades, poco apetito, sueño cambiado, estado de ánimo afectado, caída, etc. Esto con el fin de que el médico pueda visualizar estos comportamientos del paciente y pueda obtener información relevante al respecto.

  ¿Cómo funciona?. Para el registro de eventos es necesario que el usuario “cuidador” registre los datos del paciente con anterioridad. Luego, debe ingresar al apartado del paciente (ver ***Figura 5.16***), donde se encontrará los datos y una tabla resumen de los eventos registrados (los puntos verdes significan que ese síntoma fue afectado), los cuales puede visualizar en detalle. Para añadir un evento nuevo, debe presionar el botón “Añadir Evento”, el cual desplegará en pantalla un elemento con el formulario para registrarlo.

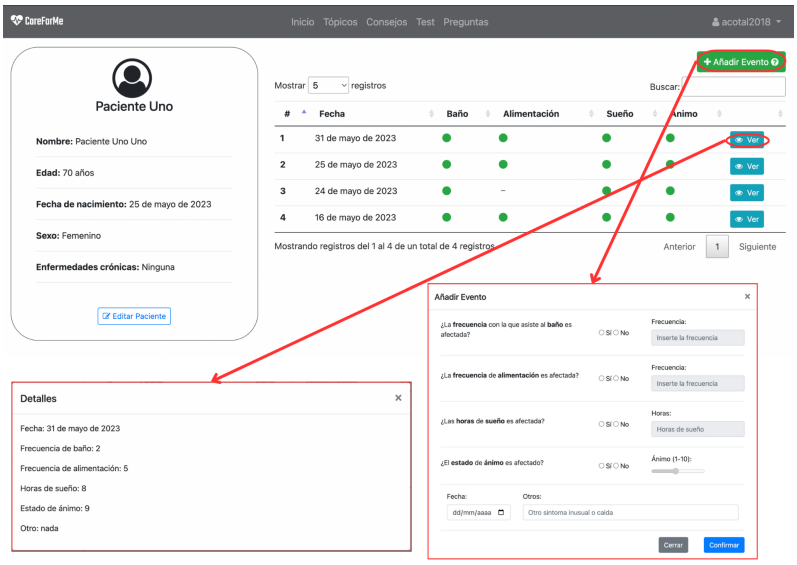

Figura 5.16: Pantalla de Paciente desde la vista del Cuidador

- **Sección de atención e informe gráfico:**

  Este apartado es exclusivo para el usuario "médico", y tiene por **objetivo** brindar una sección donde el usuario pueda ver la información del cuidador y del paciente. Estos corresponden a: los datos personales, sección **preguntas**, **informe gráfico** de los test y eventos, tablas resumen de los test y tabla resumen de los **eventos**.

  ¿Cómo funciona?. Para acceder a este apartado, el médico debe presionar el botón detalles de la lista de cuidadores (ver ***Figura 5.13***). Una vez ingresado al detalle de un cuidador en específico, verá la pantalla como en la ***Figura 5.17***. Estando ya en esta página, el médico tendrá acceso a la información respectiva del cuidador seleccionado.

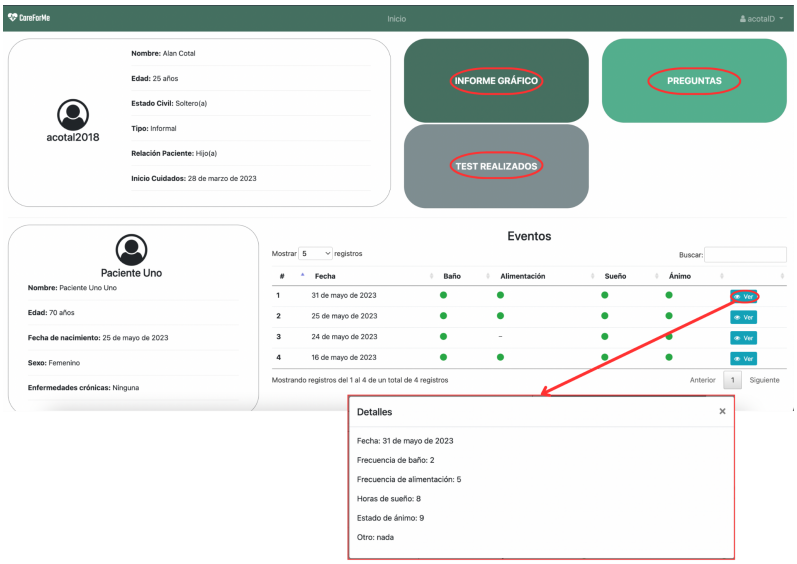

Figura 5.17: Sección de atención/detalles de cuidador

Con respecto a la sección **informe gráfico**, una vez dentro se verá la pantalla como la ***Figura 5.18***, donde podrá observar los gráficos respectivos y filtrar la vista por cada uno de ellos. Además, cada gráfico podrá ser filtrado por fecha o por tipo de leyenda, según corresponda.

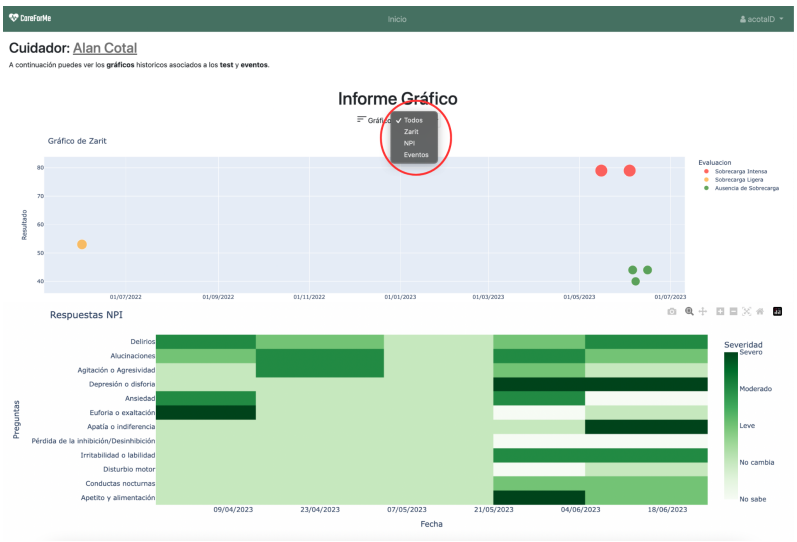

Figura 5.18: Sección de informe gráfico

Para el caso del gráfico del test **zarit**, este se basa en la puntuación que se obtiene al final de realizar el test. El eje *y* corresponde a las fechas y el eje *x* a la puntuación de cada test respectivo.

En cuanto al test **NPI-Q** debido a que se basa en 12 preguntas sobre si "algo" es afectado o no, es un poco más complejo que los otros tipos de gráficos. Para entender esto, hay que tener en cuenta que cada pregunta tiene las opciones de "Severidad" que se puede apreciar en el gráfico "Respuestas NPI" de la ***Figura 5.18***. Por lo tanto, este gráfico de calor representa la severidad de cada pregunta de un test con un determinado color. El eje *y* corresponde a la fecha del respectivo test y el eje *x* corresponde a las preguntas del test.

Figura 5.19: Gráfico de eventos

El gráfico de los **eventos** que se puede observar en la ***Figura 5.19*** es un ejemplo de cómo se podría ver en cuanto a la frecuencia de los distintos eventos o síntomas. Al ser un gráfico de tipo lineal, es posible determinar la evolución de cada uno de los eventos a través del tiempo (eje *y*) con respecto a la frecuencia (eje *x*) de cada uno de estos. Además de filtrar por fecha, también se puede filtrar el gráfico por la variable/evento correspondiente como se muestra en el ejemplo.

El apartado **test realizados** (ver ***Figura 5.17***), consiste en tablas que muestran los test que ha realizado el cuidador. El médico con esto podrá determinar el estado tanto del cuidador como del paciente. Como se puede observar en la ***Figura 5.20***, es posible ver los detalles (respuestas) de un determinado test.

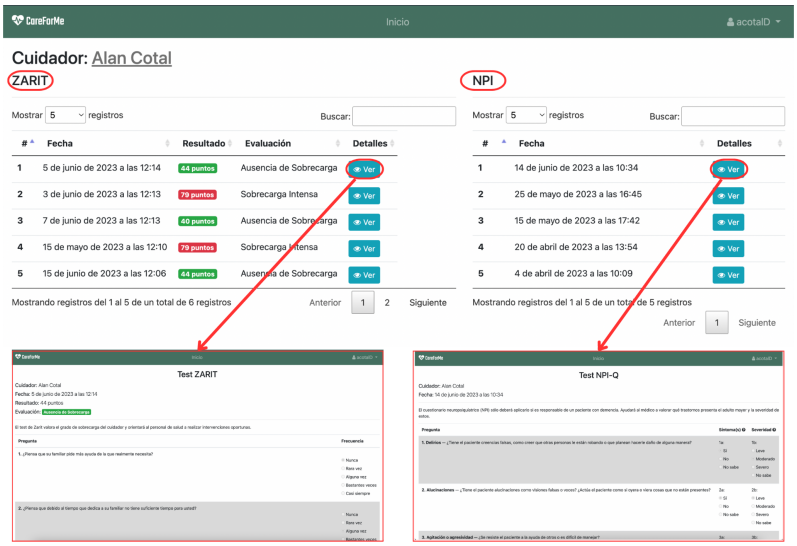

Figura 5.20: Sección test realizados desde la vista del médico

Finalmente, la sección de **preguntas** fue explicada anteriormente en el apartado "**Sección de preguntas**" de este mismo capítulo.

5.4. Despliegue

El despliegue de un proyecto Django implica llevar la aplicación web desde el entorno de desarrollo local a un servidor en la nube. En este caso se decidió por Heroku [21], por la facilidad de subir proyectos de este tipo y el soporte que se tiene de esta tecnología.

Para llevar a cabo esta parte del proyecto fue necesario realizar un cambio importante en el tipo de la base de datos, debido a que Heroku permite una base de datos SQLite pero los cambios en los datos serán efímeros [18], por lo tanto se escaló la base de datos a una MySQL. Utilizando la herramienta "JawsDB" que posee Heroku, se logra que los datos sean persistentes en el tiempo.

Cabe destacar que este despliegue se realizó con el fin de que usuarios pudieran probar la plataforma desde cualquier dispositivo, por lo que no está optimizado para obtener el mayor rendimiento. Entonces, es esperable un mayor tiempo de respuesta cuando la página muestra imágenes o debe procesar los gráficos.

Para acceder a la plataforma puede hacer click aquí (https://careformee-3e87ca750971.herokuapp.com/) . En caso de no funcionar, puede encontrar el link actualizado en el repositorio GitHub del proyecto [16].

5.5. Testeo y Validación

En este apartado se darán a conocer los métodos y resultados de pruebas del software, ya sea para la validación de las funcionalidades hasta la opinión de los usuarios objetivos.

En el ítem **5.5.1** se describe la validación de las funcionalidades con el fin de reducir los errores previa a la validación con los usuarios. Estas son pruebas automatizadas de las principales características del software.

Además, en el punto **5.5.2** se realiza una evaluación heurística de la usabilidad del software. Con el fin de encontrar aspectos mejorables en este ámbito, previamente a la evaluación con usuarios.

Luego en el ítem **5.5.3** se describe con detalle el mecanismo de validación con los usuarios finales, con el fin de obtener una retroalimentación y verificar si se han cumplido los objetivos del proyecto.

5.5.1. Pruebas con Robot Framework

Con el objetivo de validar el correcto funcionamiento de la plataforma, se han realizado pruebas con el entorno de trabajo Robot Framework [23], ya que este permite crear pruebas automatizadas de software web junto con la biblioteca Selenium [22] de una manera rápida y sencilla. Sin embargo, la realización de estas pruebas no deja de lado las "no automatizadas" para las funcionalidades más simples.

En palabras simples, estas pruebas automatizadas ejecutan las acciones de forma automática ingresando a la plataforma y simulando la navegación de distintas posibilidades para validar el correcto funcionamiento o errores que pueden surgir.

Cabe destacar que para llevar a cabo esto, fue necesario crear usuarios de forma manual para utilizarlos en las pruebas automatizadas. Además, se utiliza una copia de la base de datos original, con el objetivo de no modificar esta por error.

A continuación se detallarán las pruebas realizadas o los llamados "Test Cases" desglosados en cuatro archivos *.robot* :

- **Login:** Para este archivo se espera probar los distintos casos relacionados al acceso del usuario.

- Valid Login: Este caso de prueba se encarga de validar el acceso correcto con las credenciales del usuario, de tal manera que al rellenar el formulario de acceso con las credenciales correspondientes, pueda acceder a la página de inicio de la plataforma.
- Invalid Username Test: Como su nombre indica, para esta prueba se espera que al ingresar un nombre de usuario no válido (no registrado), no pueda acceder a la plataforma. Además, se valida que se produce un error en pantalla que deja informado al usuario de esto.
- Invalid Password Test: De manera similar al caso anterior, se ingresa una contraseña no válida para un usuario válido, esperando de igual manera que se informe el error de contraseña en la pantalla.
- Empty Username Test: Tiene el fin de validar que aparezca el error correspondiente cuando se ingresa solo la contraseña en el formulario de acceso y no el nombre de usuario.
- Empty Password Test: Similar al anterior, pero en este caso ingresando un usuario correcto pero dejando en blanco la contraseña, se espera que aparezca el error correspondiente en el campo del formulario.

Name	Documentation	Status	Elapsed
Testing Careforme . Login . Valid Login	Test para la validación de un acceso correcto con las credenciales	PASS	00:00:13.493
Testing Careforme . Login . Invalid Username Test	Test con username incorrecto	PASS	00:00:12.259
Testing Careforme . Login . Invalid Password Test	Contraseña incorrecta	PASS	00:00:10.622
Testing Careforme . Login . Empty Username Test	Campo de username vacio	PASS	00:00:10.373
Testing Careforme . Login . Empty Password Test	Campo de contraseña vacio	PASS	00:00:10.394

Figura 5.21: Resumen de resultados de "Login.robot"

En la ***Figura 5.21*** se puede observar el resultado de cada uno de los casos de prueba para el archivo especificado en este ítem.

- **Signup:** Este archivo tiene por objetivo probar los distintos casos para la funcionalidad de **registro** en la plataforma.
  - Valid Registration Cuidador: Se rellenan los campos del formulario de registro con datos correctos, con el fin de validar un registro exitoso en la plataforma.

- Invalid Registration username: Se ingresa un nombre de usuario con más de 150 caracteres para validar la restricción que posee este dato en el formulario. Se espera que el error específico se visualice en el campo correspondiente.
- Invalid Registration characters: De igual manera, se ingresa un nombre de usuario incorrecto en cuanto a la restricción de caracteres no permitidos, validando que se informe al usuario el error correspondiente.
- Invalid Registration common password: Este caso de prueba se encarga de validar el correcto funcionamiento de la restricción de no crear una contraseña que se denomine como "común", lo que significa que debe tener un mínimo de requisitos. Valida que se despliegue el error correspondiente en pantalla.
- Invalid Registration different password: Encargado de validar que las contraseñas coinciden en los campos correspondientes, desplegando un error pertinente que valida que las contraseñas no son iguales.

Name	Documentation	Status	Elapsed
Testing Careforme . Signup . Valid Registration	Registro valido Cuidador	PASS	00:00:16.184
Testing Careforme . Signup . Invalid Registration username	Username con más de 150 caracteres	PASS	00:00:15.167
Testing Careforme . Signup . Invalid Registration characters	Username incorrecto con caracteres no permitido	PASS	00:00:12.271
Testing Careforme . Signup . Invalid Registration common password	Contraseña común	PASS	00:00:11.349
Testing Careforme . Signup . Invalid Registration different password	Contraseñas no coinciden	PASS	00:00:12.354

Figura 5.22: Resumen de resultados de "Signup.robot"

- **Cuidador:** Para probar las funcionalidades específicas y la navegación para el usuario **cuidador**, se crea este archivo con los distintos casos de prueba correspondientes.
  - Valid Navigation Tópicos: Este caso de prueba valida la correcta navegación tanto de los tópicos como la de los consejos. Se espera que pueda navegar por los distintos tópicos, probando los filtros correspondientes por tema.
  - Valid Answer NPI: Con este caso de prueba se puede confirmar el correcto funcionamiento de la funcionalidad de responder el "Test NPI-Q". Se espera que se agreguen las respuestas correspondientes de manera aleatoria en el formulario, validando con un mensaje en pantalla de que se ha ingresado correctamente el test.

- Valid Answer Zarit: Lo mismo que el anterior, pero en este caso es para el "Test Zarit". Se rellenan las respuestas del formulario de forma aleatoria y se espera el mensaje de confirmación en pantalla una vez que se ha terminado el test.
- Valid Add Question: Con el objetivo de ratificar el correcto funcionamiento de la funcionalidad específica de "agregar pregunta" en la "**sección de preguntas**", se crea este caso de prueba. Se espera que al agregar una pregunta, se despliegue en pantalla un aviso de que ha sido correcta la agregación.
- Valid Add Evento: Para probar el formulario que permite añadir un "Evento" nuevo en la plataforma, se crea este caso de prueba, donde se añadirán respuestas generadas automáticamente de forma aleatoria. Esto permite que se validen los correctos ingresos de nuevos eventos a través de un aviso en pantalla.

Name	Documentation	Status	Elapsed
Testing Careforme . Cuidador . Valid Navigation Topicos	Navegación por tópicos, consejos, etc	PASS	00:00:18.002
Testing Careforme . Cuidador . Valid Answer NPI	Responder un Test NPI-Q	PASS	00:00:17.008
Testing Careforme . Cuidador . Valid Answer Zarit	Responder un Test Zarit	PASS	00:00:16.488
Testing Careforme . Cuidador . Valid Add Question	Agregar una pregunta	PASS	00:00:15.094
Testing Careforme . Cuidador . Valid Add Evento	Agregar un evento	PASS	00:00:18.061

Figura 5.23: Resumen de resultados de "Cuidador.robot"

- **Médico:** En este archivo se podrán encontrar los distintos casos de prueba para las funcionalidades específicas de un usuario de tipo **personal de salud**.
  - Valid Add/Delete Cuidador: Este caso de prueba verifica el funcionamiento de las funcionalidades, tanto de agregar un cuidador a su lista, así como también quitar uno de ellos. Se espera que agregue el cuidador que aparezca primero en la lista de los cuidadores registrados en la plataforma, luego elimina a este mismo cuidador de su propia lista de cuidadores. Esto se valida por medio de avisos correspondientes en pantalla.
  - Valid Add Answer: En la "**sección de preguntas**", también se encuentra la funcionalidad específica del médico que es responder a estas. Por lo tanto, se crea este caso de prueba con el objetivo de ratificar el correcto funcionamiento de esto. Se espera que al ingresar a esta sección, seleccione la primera pregunta sin responder y

agregue una respuesta, luego se valida con un mensaje en pantalla de que se ha realizado de manera exitosa.

Name	Documentation	Status	Elapsed
Testing Careforme . Medico . Valid Add/Delete Cuidador	Agregar el cuidador de prueba y luego eliminar	PASS	00:00:20.607
Testing Careforme . Medico . Valid Add Answer	Agregar una respuesta a una pregunta	PASS	00:00:21.386

Figura 5.24: Resumen de resultado de "Medico.robot"

Cabe señalar que se crearon las pruebas automatizadas de las funcionalidades más relevantes, ya que para el resto de funciones de la plataforma se realizan las pruebas manuales durante el desarrollo del software.

5.5.2. Análisis Heurístico

Uno de los métodos para realizar una evaluación de usabilidad del software es la "Evaluación en base a Heurísticas", en este caso utilizando el marco de las 10 heurísticas de usabilidad establecidas por Nielsen [7] [27]. Estas, representan principios fundamentales para el diseño de interfaces y la mejora de la experiencia del usuario en plataformas digitales.

A través de un análisis minucioso en base a una plantilla realizada por Bart [28], se examinará cómo la plataforma CareForMe se ajusta a cada una de las heurísticas, identificando posibles áreas de mejora y ofreciendo recomendaciones con el objetivo de optimizar la usabilidad y eficiencia del sistema.

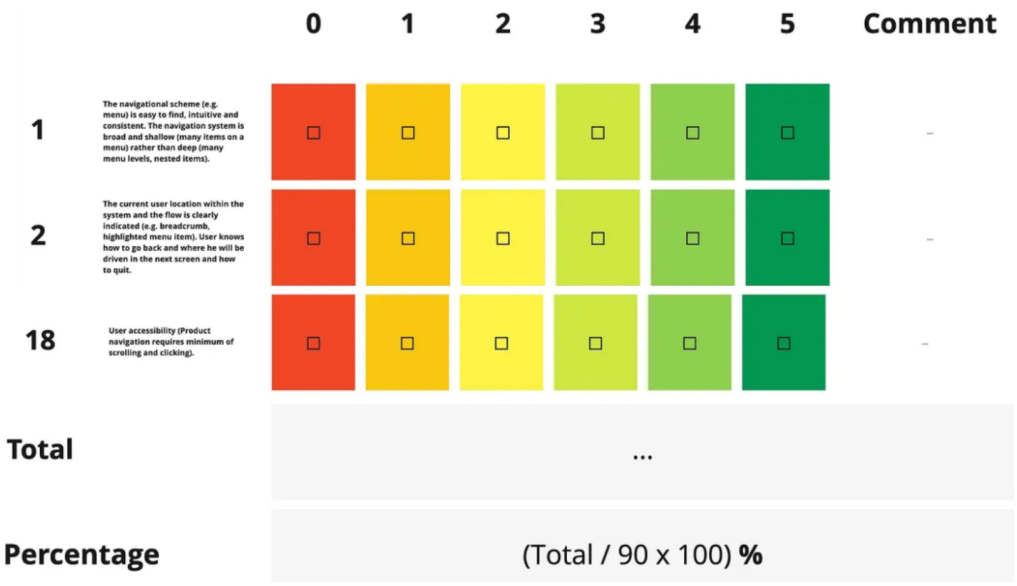

Figura 5.25: Ejemplo de estructura de la plantilla de Bart[28]

Como se puede observar en la ***Figura 5.25***, la plantilla utilizada corresponde a un conjunto de preguntas por cada heurística, que se pueden evaluar con puntajes que van desde 0 puntos hasta 5 puntos. Luego, estos puntajes son sumados y se le aplica la fórmula que se puede ver en la ***Figura 5.25***, de esta manera se obtiene un porcentaje de "eficiencia" por cada heurística a evaluar.

Para este análisis en concreto, se evaluaron las 10 heurísticas de Nielsen. Además, cabe mencionar que esta evaluación fue realizada por:

- Joaquín Barahona (estudiante de último semestre de Ingeniería Civil Informática, Udec).
- Félix Torres (estudiante de último semestre de Ingeniería Civil Informática, Udec).
- Geoffrey Hecht (profesor patrocinante).
- Alan Cotal (autor).

Dando como resultado lo siguiente:

	Barahona (%)	Torres (%)	Cotal (%)	Hecht (%)	Promedio (%)
1. Visibilidad del estado del sistema	88	96	80	84	**87**

2. Sincronía entre el sistema y el mundo real	100	95	93	93	**95**
3. Control y libertad del usuario	68	68	76	96	**77**
4. Consistencia y estándares	89	94	92	98	**93**
5. Prevención de errores	78	91	64	87	**80**
6. Reconocimiento antes que recuerdo	88	100	84	92	**91**
7. Flexibilidad y eficiencia de uso	89	94	68	92	**86**
8. Diseño estético y minimalista	96	100	86	100	**96**
9. Reconocimiento, diagnóstico y recuperación de errores	97	90	60	90	**84**
10. Ayuda y documentación	50	86	60	90	**72**

Tabla 5.1: Resultados de evaluación heurística

Como se puede observar en la ***Tabla 5.1*** en los resultados promedios, hay heurísticas con buena valoración, pero hay otras por mejorar. Por lo que también surgieron comentarios al respecto y se detallarán a continuación por cada una de las heurísticas:

1. Visibilidad del estado del sistema:
    - Falta una barra de progreso al completar los test.
3. Control y libertad del usuario:
    - Al rellenar cualquier formulario, las respuestas se pierden al recargar la página.
5. Prevención de errores:
    - El filtro de búsqueda al no encontrar nada, no informa que no se encuentran elementos que coincidan con la palabra/s que se ingresan.
    - Faltan textos de confirmación al entregar un test, volver atrás, cerrar la página, etc.
7. Flexibilidad y eficiencia de uso:
    - No es posible eliminar un test, evento y pregunta, una vez que son ingresadas.
    - Las tablas se contraen desde dispositivos móviles, y hay que desplegar las columnas con un botón.
    - Los gráficos son poco legibles desde dispositivos móviles.
10. Ayuda y documentación:
    - En general no hay textos de ayuda para la usabilidad.

En base a los resultados y los comentarios, se puede tener una evaluación de la usabilidad de manera cuantificada y proceder a un mejoramiento en el software. Dando mayor atención a las heurísticas con menor porcentaje, en este caso la **3**, **5** y **10**.

5.5.3. Pruebas con Usuarios

Con el objetivo de evaluar la experiencia de los usuarios al utilizar la plataforma, se creó una encuesta basada en "System Usability Scale (SUS) [24]", que permite evaluar de una manera rápida y confiable la usabilidad de una herramienta o software en base a diez preguntas con cinco opciones de respuesta, que van desde "totalmente en desacuerdo" a "totalmente de acuerdo". Sin embargo, este método es más confiable en cuanto a la usabilidad pero no en rendimiento, como se concluye en el artículo de Peres [25].

Además de estas diez preguntas bases del SUS, se agregaron cinco preguntas adicionales de manera personalizada para el proyecto en cuestión.

Las preguntas de la encuesta se pueden apreciar en la ***Tabla 5.2***.

	Totalmente en desacuerdo				Totalmente de acuerdo
1. Creo que me gustaría usar esta plataforma con frecuencia	1	2	3	4	5
2. Encontré la plataforma innecesariamente compleja	1	2	3	4	5
3. Pienso que la plataforma era fácil de usar	1	2	3	4	5
4. Creo que necesitaría la asistencia de un experto para poder usar esta plataforma	1	2	3	4	5
5. Encontré que las diversas funciones de esta plataforma estaban bien integradas	1	2	3	4	5
6. Pienso que había demasiada inconsistencia en esta plataforma	1	2	3	4	5
7. La mayoría de la gente aprenderá a usar esta plataforma muy rápidamente	1	2	3	4	5
8. Me pareció que la plataforma era muy incómoda de usar	1	2	3	4	5
9. Me sentí muy confiado/a usando la plataforma	1	2	3	4	5
10. Necesito aprender muchas cosas antes de poder ponerme en marcha con esta plataforma	1	2	3	4	5
Otras					

11. Del 1 al 10. ¿Cómo evaluaría la interfaz/diseño de la plataforma?	
12. ¿Qué funcionalidad de la plataforma ha sido la que más le ha gustado?	
13. ¿Qué funcionalidad de la plataforma usted eliminaría?	
14. ¿Qué otras funcionalidades extras agregarías a la plataforma?	
15. Comentario/opinión en general:	

Tabla 5.2: Preguntas de encuesta a usuarios

Lamentablemente, la aplicación de la encuesta no se pudo llevar a cabo debido a la falta de disponibilidad de usuarios dispuestos a probar la plataforma. Esta situación se debió en gran medida al tiempo acotado para la realización de esto. Aunque, se contó con la ayuda de la doctora *Claudia Sáez* para encontrar a usuarios que pudieran participar en este estudio, pero desafortunadamente no se pudo asegurar la participación mínima necesaria. Sin embargo, durante el desarrollo del proyecto y al final de este, se obtuvo retroalimentación positiva por parte de *Pamela Guevara* y *Claudia Sáez*, mencionando que se implementó correctamente lo solicitado y que esperan poder utilizar la plataforma en un futuro.

6. Discusión y Conclusiones

El propósito del presente proyecto era crear una nueva versión de la plataforma "CareForMe" [3], con el objetivo de brindar información relevante para los cuidados de personas con demencia y apoyar a los cuidadores en esta labor. A lo largo de este documento, se han descrito las funcionalidades de la versión inicial y las mejoras implementadas en esta nueva versión del software, diseñadas para mejorar la experiencia del usuario. En esta sección se discutirán los resultados y conclusiones obtenidos durante el desarrollo del proyecto.

En primer lugar, la incorporación de la asociación personal entre un usuario "cuidador" y un usuario "personal de salud" fue un avance significativo en esta nueva versión. Ya que, esto permite una atención más personalizada y centrada en el cuidador. Además, posibilita que se hayan agregado las nuevas funcionalidades como el registro de eventos del paciente y la sección de preguntas, permitiendo realizar un seguimiento más detallado de la evolución de

los pacientes. Del mismo modo, debido a que el personal de salud tiene acceso a la información del cuidador y visualizar los datos (historial de test neuropsiquiátricos, eventos) en gráficos, facilitan la toma de decisiones más informadas y la identificación de posibles problemas.

Con la nueva versión de *CareForMe* se ha logrado cumplir con los objetivos propuestos al inicio de este documento. Las nuevas funcionalidades le han dado un valor agregado al software, brindando más herramientas a los cuidadores. Aunque, desafortunadamente no se ha logrado evaluar la percepción de los usuarios objetivos, por lo que no se tienen resultados cuantificables en ese ámbito. Pero sí se logró realizar una evaluación heurística de la usabilidad, lo que permite concluir que en términos generales cumple satisfactoriamente en la mayoría de las heurísticas de Nielsen. Además, en esta evaluación se evidencia que hay mejoras por hacer.

En conclusión, el proyecto ha representado una oportunidad para el autor, aplicando sus conocimientos y habilidades en un entorno práctico y significativo. Uno de los puntos más relevantes que destaca sobre la experiencia, es que le ha brindado valiosas lecciones sobre la importancia de la colaboración interdisciplinaria, la adaptabilidad y la dedicación en el logro de los objetivos.

Glosario

Framework: un marco de trabajo o framework es una estructura conceptual y tecnológica que proporciona una base para el desarrollo de aplicaciones, sistemas o proyectos. Incluye herramientas, bibliotecas y convenciones que agilizan el proceso de desarrollo al ofrecer soluciones predefinidas para problemas comunes.

Interfaz: una interfaz se refiere a la forma en que los usuarios interactúan con un sistema, software o dispositivo. Puede ser una interfaz gráfica de usuario (GUI) o una interfaz de línea de comandos (CLI), diseñadas para facilitar la comunicación y la interacción entre el usuario y la tecnología.

NPI-Q: el NPI-Q (Neuropsychiatric Inventory Questionnaire) es un cuestionario utilizado en salud mental y neurología para evaluar síntomas neuropsiquiátricos en pacientes con trastornos cognitivos o neurológicos, abordando áreas como depresión, ansiedad y comportamiento.

Plataforma: una plataforma es un entorno o conjunto de herramientas que permite el desarrollo y ejecución de aplicaciones, servicios o sistemas. Puede ser de software, como un sistema operativo o plataforma en la nube, que brinda recursos para crear y desplegar soluciones tecnológicas.

Software: el software es un conjunto de programas, datos y procesos que instruyen a una computadora para realizar tareas específicas. Incluye sistemas operativos, aplicaciones y utilidades que permiten a los usuarios interactuar con la computadora y llevar a cabo actividades.

Subtópico: un subtópico es un tema específico que se deriva de un tópico más amplio. Sirve para categorizar y organizar información relacionada con aspectos particulares de un tema general.

Tópico: un tópico es un tema principal o área de discusión. Actúa como categoría principal que engloba subtemas o detalles relacionados, facilitando la organización de la información.

UI/UX: UI (Interfaz de Usuario) y UX (Experiencia de Usuario) son conceptos de diseño. UI abarca la apariencia visual de elementos con los que el usuario interactúa, mientras que UX se enfoca en hacer la interacción intuitiva y satisfactoria.

Zarit: la Escala de Sobrecarga del Cuidador de Zarit (Zarit Burden Interview) mide la sobrecarga percibida por los cuidadores de personas con enfermedades crónicas o discapacidades. Evalúa el impacto emocional y físico del cuidado.

Referencias

[1] Villalobos Dintrans, P. (2019). Informal caregivers in Chile: the equity dimension of an invisible burden. Health Policy and Planning, 34(10), 792-799.

[2] Sandoval, F., Tamiya, N., Lloyd‐Sherlock, P., & Noguchi, H. (2019). The relationship between perceived social support and depressive symptoms in informal caregivers of community‐dwelling older persons in Chile. Psychogeriatrics, 19(6), 547-556.

[3] Diego Muñoz (2021). Desarrollo de plataforma web para el apoyo del cuidador informal. Informe de Memoria de Título de Ingeniería Civil Informática, Universidad de Concepción.

[4] Constanza Ortiz E. (2018). Software para el aprendizaje del cuidado al adulto mayor. Informe de Memoria de Título de Ingeniería Civil Biomédica, Universidad de Concepción.

[5] Goree, S., Doosti, B., Crandall, D., & Su, N. M. (2021, May). Investigating the homogenization of web design: A mixed-methods approach. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-14).

[6] Namoun, A., Alshanqiti, A., Chamudi, E., & Rahmon, M. A. (2020, October). Web design scraping: Enabling factors, opportunities and research directions. In 2020 12th International Conference on Information Technology and Electrical Engineering (ICITEE) (pp. 104-109). IEEE.

[7] Nielsen, J. (2005). Ten usability heuristics.

[8] Darejeh, A., & Singh, D. (2013). A review on user interface design principles to increase software usability for users with less computer literacy. Journal of Computer Science, 9(11), 1443.

[9] McClurg-Genevese, J. D. (2005). The principles of design. Digital Web Magazine, 13.

[10] Aleryani, A. Y. (2016). Comparative study between data flow diagram and use case diagram. International Journal of Scientific and Research Publications, 6(3), 124-126.

[11] Django, FAQ:General: https://docs.djangoproject.com/en/4.1/faq/general/#does-django-scale

[12] Bucanek, J. (2009). Model-view-controller pattern. Learn Objective-C for Java Developers, 353-402.

[13] EspiFreelancer (2019). Que es el patrón MTV (Model Template View). https://espifreelancer.com/mtv-django.html

[14] Al-Saqqa, S., Sawalha, S., & AbdelNabi, H. (2020). Agile software development: Methodologies and trends. International Journal of Interactive Mobile Technologies, 14(11).

[15] demm94 GitHub (2021). web-cuidador-informal. https://github.com/demm94/web-cuidador-informal.

[16] acotal22 GitHub (2023). web-cuidador-informal. https://github.com/acotal22/web-cuidador-informal.

[17] Official Documentation Django 4.1. https://docs.djangoproject.com/en/4.1/.

[18] SQLite on Heroku (2022). https://devcenter.heroku.com/articles/sqlite3.

[19] Gregersen, H., & Jensen, C. S. (1999). Temporal entity-relationship models-a survey. IEEE Transactions on knowledge and data engineering, 11(3), 464-497.

[20] Ravindran, A. (2015). Django Design Patterns and Best Practices. Packt Publishing Ltd.

[21] About Heroku Platform. https://www.heroku.com/platform.

[22] Selenium Library Robot Framework. https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html .

[23] Robot Framework. https://robotframework.org/ .

[24] Brooke, John. (1995). SUS: A quick and dirty usability scale. Usability Eval. Ind.. 189.

[25] Peres, S. C., Pham, T., & Phillips, R. (2013, September). Validation of the system usability scale (SUS) SUS in the wild. In Proceedings of the human factors and ergonomics society annual meeting (Vol. 57, No. 1, pp. 192-196). Sage CA: Los Angeles, CA: SAGE Publications.

[26] YoTeCuido, página oficial (2018) http://www.yotecuidoalzheimer.com/

[27] Nielsen, J. (1995). How to conduct a heuristic evaluation. Nielsen Norman Group, 1(1), 8.

[28] Bart Szczepansky (2021). Usability Heristics Template. Medium. https://medium.com/goal-directed-design/usability-heuristics-template-5ad26762de14