



GREENLIFE NUTRITION

Desarrollo de una web para la gestión personalizada de dietas y clientes

Nelson Blanco Charro

IES San Andrés

Oscar Hernando Tejedor

CFGS Desarrollo de Aplicaciones Web

Dpto. Informática

Mayo de 2023

1 Tabla de contenido

1	Tabla de contenido	1
1	Estudio del problema y análisis del sistema	2
1.1	Introducción	2
1.2	Finalidad	3
1.3	Requisitos	6
2	Recursos	7
2.1	Recursos hardware	7
2.2	Recursos software	8
3	Planificación	9
3.1	Planificación temporal	9
3.2	Planificación económica	12
4	Desarrollo y pruebas	14
4.1	Diseño e implementación del sistema	14
4.2	Pruebas del sistema	23
5	Descripción de la aplicación	25
6	Conclusiones finales	31
6.1	Grado de cumplimiento de los requisitos fijados	31
6.2	Propuesta de mejora o ampliaciones futuras	32
7	Guías	34
7.1	Guía de instalación	34
7.2	Guía de uso	36
8	Referencias bibliográficas	40

1 Estudio del problema y análisis del sistema

1.1 Introducción

En la actualidad, el desarrollo de aplicaciones web se ha convertido en una herramienta fundamental para diversas áreas profesionales, entre ellas la nutrición. Por esta razón, en el presente Trabajo de Fin de Grado se presenta el desarrollo de una página web para una nutricionista real, la cual permitirá mostrar información detallada de dietas a los clientes, tales como pesos, platos recomendados y recomendaciones adicionales.

El proyecto ha sido desarrollado utilizando el framework Laravel, el cual es uno de los más utilizados para el desarrollo de aplicaciones web. Este framework ofrece una gran cantidad de funcionalidades, entre ellas la conexión directa a una base de datos MySQL para el almacenamiento de información. Esta conexión permitirá a la nutricionista almacenar información de sus clientes y dietas de manera segura y eficiente.

Además, en el presente proyecto se ha creado una API externa en Typescript, que permitirá obtener información nutricional de los platos recomendados. Esto mejorará significativamente la experiencia de los clientes en la página web, ya que podrán obtener información más detallada sobre los platos que se les recomiendan. La utilización de Typescript permitirá también la creación de un código más robusto y eficiente, ofreciendo una mayor estabilidad y mejor rendimiento.

El diseño de la página ha sido desarrollado utilizando TailwindCSS, un framework de CSS que permite la creación de diseños modernos y responsivos. Esto permitirá a los clientes de la nutricionista navegar por la página de manera amigable e intuitiva, y a la nutricionista presentar su información de manera clara y concisa.

En cuanto a la utilización del proyecto, se espera que la nutricionista lo utilice como una herramienta para presentar información a sus clientes de manera digital, lo que permitirá una mayor eficiencia y rapidez en el acceso a la información. La página web se ha desarrollado pensando en las necesidades de la nutricionista y de sus clientes, lo que permitirá una experiencia de usuario óptima.

En resumen, el presente proyecto ha sido desarrollado con tecnologías modernas y actuales, tales como Laravel, MySQL, Typescript y TailwindCSS, y se ha enfocado en satisfacer las necesidades de una nutricionista real y sus clientes. La creación de una página web para la presentación de información nutricional permite una mayor eficiencia y rapidez en el acceso a la información, lo que permitirá a la nutricionista una mejor gestión de sus clientes y dietas. La utilización de tecnologías modernas y actuales garantiza la estabilidad y seguridad del proyecto, lo que permitirá una mayor satisfacción por parte de los usuarios y una mayor eficiencia en el manejo de la información.

1.2 Finalidad

La finalidad del proyecto presentado en el Trabajo de Fin de Grado es la creación de una página web simple, efectiva y atractiva para la presentación de información nutricional a los clientes de una nutricionista. Se busca que la página web sea capaz de mostrar información detallada de las dietas y recomendaciones a los clientes, así como permitir la modificación de los datos de los clientes de manera fácil y sencilla.

La privacidad de los clientes es una cuestión fundamental en el desarrollo de la página web del proyecto presentado. En este sentido, se busca que la página web sea capaz de mostrar información a los clientes sin que aparezcan sus nombres. Es importante destacar que es una petición explícita de la nutricionista, quien ha expresado la necesidad de mantener en todo momento la privacidad y seguridad de la información de sus clientes. Por lo tanto, se ha decidido no incluir un formulario de registro para clientes, sino que se dará de alta a los nuevos clientes desde el panel de administración de la nutricionista.

En consecuencia, el desarrollo de la página web se realizará bajo un enfoque riguroso en la protección de la privacidad y seguridad de la información de los clientes. De esta manera, se busca satisfacer plenamente las necesidades y requerimientos de la nutricionista, garantizando al mismo tiempo la confidencialidad y privacidad de la información de sus clientes en todo momento.

La finalidad del proyecto presentado no solo se enfoca en satisfacer las necesidades y requerimientos de la nutricionista en cuanto a la gestión de la información de sus clientes, sino también en contribuir al crecimiento y expansión de su negocio.

En este sentido, se espera que la página web diseñada sea capaz de atraer nuevos clientes. Para lograr este objetivo, se ha puesto especial atención en el diseño, la usabilidad y la accesibilidad de la página web, con el objetivo de que sea atractiva y fácil de navegar para cualquier visitante.

Además, se ha prestado especial atención en la presentación de la información de las dietas, platos y recomendaciones nutricionales, buscando que sea clara, sencilla y efectiva, de manera que cualquier persona pueda entenderla. Todo esto con el fin de que los potenciales clientes puedan obtener información valiosa sobre los servicios de la nutricionista, lo que les podría motivar a recomendar sus servicios a conocidos. En consecuencia, el diseño y la funcionalidad de la página web del proyecto presentado en el Trabajo de Fin de Grado están pensados no solo para satisfacer las necesidades actuales de la nutricionista, sino también para contribuir a la expansión de su negocio, atrayendo nuevos clientes y manteniendo una imagen sólida y confiable en el mercado.

En el proyecto presentado se ha puesto especial atención en mejorar la eficiencia de la gestión de los clientes y dietas por parte de la nutricionista. En este sentido, se busca simplificar su trabajo actual a través del uso de herramientas tecnológicas modernas que permitan una mayor automatización y optimización de los procesos.

La página web diseñada en Laravel y conectada a una base de datos MySQL, permitirá a la nutricionista acceder y modificar fácilmente la información de sus clientes, así como crear y modificar las dietas de manera sencilla y efectiva.

Por otro lado, el diseño de la página web también busca reflejar la mentalidad y ética de la nutricionista. Se ha prestado especial atención en la presentación de la información de manera clara y concisa, con el objetivo de transmitir los valores de la nutricionista y su compromiso con el bienestar y la salud de sus clientes.

Además, el diseño atractivo y la usabilidad de la página web permitirán una mayor conexión con los clientes y una mejor presentación del trabajo de la nutricionista. Los clientes podrán acceder a la información de sus dietas y platos recomendados de manera fácil y rápida, lo que les permitirá tener una mejor comprensión de los servicios ofrecidos y una mayor confianza en la nutricionista.

La accesibilidad es un aspecto fundamental en el desarrollo de aplicaciones web modernas, ya que permite a todas las personas, independientemente de sus capacidades físicas o cognitivas, acceder a la información y funcionalidades disponibles. En el caso de la página web desarrollada para la nutricionista, se busca garantizar que cualquier persona pueda utilizarla de manera efectiva, sin importar su edad, género, patología o cualquier otra condición.

Para lograr este objetivo, se seguirán las pautas y estándares de accesibilidad web establecidos por la World Wide Web Consortium (W3C), incluyendo la aplicación de etiquetas HTML semánticas y atributos accesibles, el uso adecuado de contrastes de color, el soporte para dispositivos de asistencia como lectores de pantalla y teclados especiales, entre otros.

El proyecto presentado en el Trabajo de Fin de Grado no solo busca ofrecer soluciones innovadoras y eficientes para la gestión de clientes y dietas de la nutricionista, sino también permitir el aprendizaje de nuevas tecnologías y el perfeccionamiento de conocimientos previos en tecnologías fundamentales en el desarrollo de aplicaciones web.

Entre las nuevas tecnologías que se busca incorporar al proyecto se encuentran Tailwind y Typescript. Tailwind es un framework de CSS que permite la creación de diseños atractivos y personalizados con una mayor facilidad y rapidez. Por otro lado, Typescript es un lenguaje de programación que permite agregar tipado estático a Javascript, lo que mejora la eficiencia en el desarrollo de aplicaciones web.

Asimismo, se espera perfeccionar los conocimientos previos en tecnologías como Laravel, Javascript y CSS aprendidos en el ciclo. Laravel es un framework de PHP que facilita la creación de aplicaciones web eficientes y escalables, mientras que Javascript es un lenguaje de programación fundamental en el desarrollo de aplicaciones web. CSS, por su parte, es el lenguaje de diseño utilizado para dar estilo y personalidad a las páginas web.

El aprendizaje y uso de estas tecnologías permitirá una mayor versatilidad en el desarrollo de aplicaciones web, lo que se traducirá en una mayor eficiencia y estabilidad en el desarrollo del proyecto.

En resumen, el objetivo principal del proyecto es crear una página web que satisfaga las necesidades de la nutricionista y sus clientes, permitiendo una gestión eficiente de la información nutricional y una presentación atractiva y accesible. La web se diseñará para mostrar información detallada de las dietas, platos y recomendaciones a los clientes, lo que permitirá una mejor gestión de su información nutricional y una mayor conexión con la nutricionista. La privacidad de los clientes es una prioridad para la nutricionista, por lo que se ha solicitado expresamente que la página web sea capaz de mostrar información a los clientes sin que aparezcan sus nombres. Asimismo, se busca simplificar el trabajo actual de la nutricionista, permitiéndole una mayor eficiencia en la gestión de sus clientes y dietas.

El proyecto también tiene como objetivo el aprendizaje de nuevas tecnologías, como Tailwind y Typescript, lo que permitirá una mayor versatilidad en el desarrollo de aplicaciones web. Además, se espera perfeccionar los conocimientos previos en tecnologías como Laravel, Javascript y CSS, lo que permitirá una mayor eficiencia y estabilidad en el desarrollo del proyecto.

1.3 Requisitos

El presente apartado tiene como objetivo detallar los requisitos necesarios para el desarrollo de la aplicación. Estos requisitos se han establecido con el fin de asegurar un funcionamiento óptimo y satisfactorio de la plataforma, así como brindar una experiencia agradable tanto para la nutricionista como para los clientes.

- Se debe implementar un *diseño atractivo* y visualmente agradable en toda la aplicación.
- Se debe crear una *página principal que contenga toda la información* relevante sobre la nutricionista, como su experiencia, formación y servicios ofrecidos.
- La aplicación debe permitir *mostrar a cada cliente su dieta* personalizada, teniendo en cuenta sus requerimientos y objetivos específicos.
- Es necesario incorporar un *gráfico que muestre la evolución del peso de cada cliente* a lo largo del tiempo, brindando una visualización clara de los cambios.
- Se debe implementar un *formulario que permita recopilar información relevante sobre los hábitos personales y alimenticios de los clientes*, con el fin de brindar un contexto adecuado para la nutricionista.
- La aplicación debe contar con un *sistema de autenticación que permita el acceso diferenciado para la nutricionista y el administrador*.
- Se debe habilitar la *posibilidad de registrar y almacenar los pesos de los clientes*, permitiendo actualizarlos de forma regular para llevar un seguimiento preciso.
- La aplicación debe permitir dar de *alta a nuevos clientes*, ingresando su información personal, datos de contacto, pesos iniciales y objetivos de pérdida de peso.
- Se requiere implementar un *algoritmo que calcule el peso teórico* de cada cliente, considerando una pérdida de peso normal y estableciendo metas alcanzables.
- El sistema debe contar con una *interfaz intuitiva y sencilla* que facilite la introducción de datos relevantes de los clientes, como información personal, pesos, dietas y textos adicionales.
- Se debe permitir la *modificación de los datos de los clientes de forma sencilla*, brindando opciones para actualizar información personal, pesos, dietas y otros detalles relevantes.
- La aplicación deberá contar con un *sistema de mensajería* que permita la comunicación entre los clientes y la nutricionista, así como también la recepción de mensajes de usuarios no registrados.
- Se debe implementar una funcionalidad que permita *mostrar a los clientes los valores nutricionales de los platos de su dieta*, proporcionando información detallada sobre los nutrientes y calorías.
- La aplicación deberá contar con un *sistema de registro y seguimiento de errores*, almacenando información relevante sobre las incidencias ocurridas para facilitar su posterior análisis y corrección.
- Se requiere implementar un *sistema de gestión de clientes que garantice la confidencialidad y protección* de los datos personales.
- La aplicación web debe contar con un *diseño responsive*, adaptándose de forma óptima a diferentes dispositivos y tamaños de pantalla.

2 Recursos

2.1 Recursos hardware

En la sección de recursos de hardware, se detallan los dispositivos y herramientas tecnológicas utilizados durante el desarrollo del Trabajo de Fin de Grado.

Para el **desarrollo** de la aplicación, uno de los recursos más importantes es el ordenador, que se utiliza para la programación y el diseño de la aplicación web. En este caso, se ha seleccionado el Asus Tuf Dash F15, un equipo con procesador Intel Core i7, 16GB de memoria RAM, una tarjeta gráfica NVIDIA GeForce RTX 3070, y un disco duro de 1TB SSD.

El teclado mecánico HyperX Alloy Origins Core y el ratón inalámbrico Logitech Mx Masters 3S fueron utilizados para mejorar la comodidad y eficiencia en la entrada de datos y el uso de herramientas de desarrollo. Estos dispositivos tienen una alta capacidad de respuesta y personalización, lo que permitió una mayor velocidad y precisión en el trabajo.

La pantalla externa Samsung S34J552, con una resolución de 3440x1440, proporciona una amplia área de trabajo y una alta calidad de imagen para aumentar el espacio de trabajo disponible y mejorar la visualización de los diseños y elementos de la interfaz de usuario de la aplicación.

Finalmente, se utilizó una variedad de dispositivos móviles, como el Xiaomi Poco M4 Pro 5G, el iPhone 12 Pro Max y el Huawei Pro 30, para probar y validar la compatibilidad y funcionalidad de las aplicaciones web desarrolladas en diferentes plataformas y sistemas operativos móviles.

Para el **despliegue** de la aplicación, se requiere un servidor web que será responsable de alojar y servir los archivos y recursos de la aplicación de manera eficiente. Se ha elegido el servicio *render.com* que permite el despliegue fácil y rápido de aplicaciones de forma gratuita. También ofrece una opción de escalabilidad que resulta especialmente útil en situaciones donde hay un aumento significativo en el tráfico de la aplicación. Este servicio brinda opciones flexibles y precios competitivos para adaptarse a las necesidades específicas de nuestro proyecto.

Por último, para poder **ejecutar** la página web correctamente, es necesario contar con una conexión a internet estable. La conexión a internet permitirá la comunicación entre el navegador web del usuario y el servidor que aloja la aplicación. Es importante destacar que la página web es compatible con el 98% de los navegadores actuales y con los sistemas operativos más utilizados en el mercado. Esto implica que la aplicación funciona correctamente en una amplia gama de navegadores populares, como *Google Chrome*, *Mozilla Firefox*, *Microsoft Edge*, *Safari*, entre otros. Asimismo, la página web debe ser accesible desde diferentes sistemas operativos, como *Windows*, *macOS*, *Linux*, *iOS* y *Android*.

2.2 Recursos software

En ésta sección detallaremos los programas y herramientas informáticas utilizados en el proceso de desarrollo del proyecto.

En primer lugar, una herramienta de gran importancia para el desarrollo de aplicaciones web es Visual Studio Code, un editor de código fuente que permite una gran eficiencia en la escritura de código, depuración y gestión de proyectos.

Asimismo, para la creación de servidores locales se ha utilizado Xampp, que incluye Apache y MariaDB, permitiendo la creación de un servidor web y base de datos locales para el desarrollo y pruebas del proyecto.

Para la comprobación de la funcionalidad de la página en distintos navegadores, se han utilizado Firefox, Chrome y Edge, que permiten verificar la compatibilidad del proyecto en los navegadores más utilizados.

Otro software utilizado en el proyecto es WPS Office, una suite ofimática que permite la creación de documentos y presentaciones necesarios para el trabajo de fin de grado.

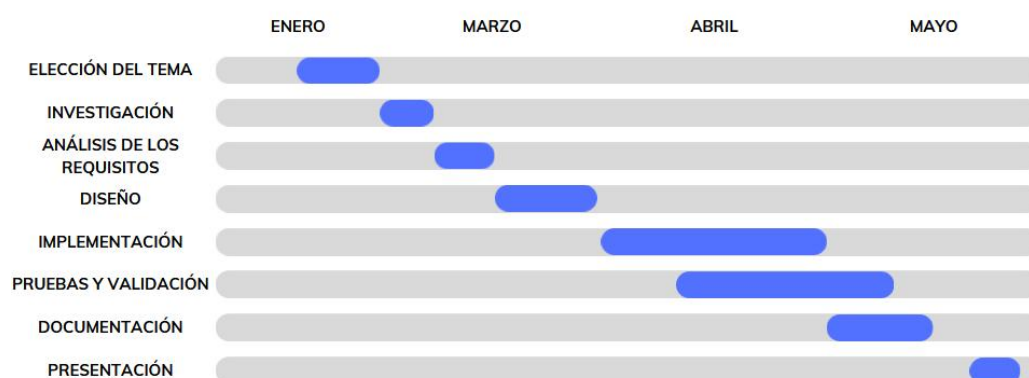
En cuanto a los frameworks, se ha utilizado Laravel, un framework de desarrollo web de código abierto en PHP que permite la creación de aplicaciones web robustas y escalables. También se ha utilizado Typescript, un lenguaje de programación que permite la escritura de código más seguro y eficiente.

En cuanto al sistema operativo, se ha utilizado Windows 11, que ofrece una amplia compatibilidad con el software necesario para el desarrollo de aplicaciones web.

3 Planificación

3.1 Planificación temporal

La planificación temporal es una etapa esencial en cualquier proyecto, y el Trabajo de Fin de Grado no es la excepción. Esta sección tiene como objetivo establecer un cronograma para el desarrollo del proyecto, definiendo las fases, tareas y tiempos necesarios para su realización. La planificación temporal permite organizar las actividades de forma eficiente, fijar metas y objetivos, y establecer un plazo realista para la entrega del trabajo final.



Elección del tema (15 de enero - 31 de enero)

Durante las primeras semanas del proyecto, se llevó a cabo una reunión con la nutricionista para acordar los requisitos iniciales que se deben cumplir para que el proyecto sea satisfactorio para ambas partes. En esta reunión, se discutió y estableció el alcance del proyecto, los objetivos a alcanzar, las funcionalidades necesarias y los plazos para la entrega del proyecto.

Una vez que se definieron los requisitos, se presentó la propuesta al tutor para su aprobación y para asegurarnos de que el trabajo esté dentro de los objetivos y las expectativas del proyecto. De esta manera, podemos asegurarnos de cumplir con los objetivos del proyecto y entregarlo en el plazo establecido.

Investigación (1 de marzo - 7 de marzo)

Una vez que el tema del Trabajo de Fin de Grado fue aprobado por el tutor, se inició una fase de investigación exhaustiva para determinar las tecnologías más adecuadas para satisfacer los requisitos del proyecto. Se optó por Laravel para el desarrollo de la página web y una API Rest en Typescript para mostrar la información nutricional de los platos de la dieta. La selección de estas tecnologías se basó en su capacidad para cumplir con los requerimientos planteados y su compatibilidad con las necesidades específicas del proyecto.

Además de la investigación tecnológica, también se llevó a cabo un análisis de los conceptos básicos de nutrición humana y se investigó otras páginas web de nutricionistas para tener una idea más clara de cómo se estructuran y presentan los planes de alimentación. La finalidad de esta etapa fue garantizar que el trabajo final sea una solución robusta y completa que cumpla con los objetivos planteados por la nutricionista.

Análisis de los requisitos (8 de marzo - 15 de marzo)

La fase de análisis de requisitos es una de las más importantes del Trabajo de Fin de Grado, ya que es en este momento en el que se establecen las bases para el resto del proyecto. En este caso, se llevaron a cabo varias reuniones con la nutricionista para presentar la propuesta técnica y de diseño visual y así poder ajustar los requisitos a las necesidades de ambas partes.

En lo que respecta a la parte técnica, se estableció detalladamente cómo sería la interacción del usuario con la página web, qué datos se mostrarían y cómo accedería la nutricionista a dichos datos para su consulta y modificación. Además, se trabajó en cómo se integraría la API con la página web, para que todo funcionara de manera fluida y coherente.

En cuanto a la parte de diseño visual, se presentó inicialmente un wireframe básico de la página web para que la nutricionista pudiera tener una idea general de la estructura y la disposición de los distintos apartados. Posteriormente, se diseñó un mockup más detallado y atractivo visualmente, en el que se incluyeron imágenes y colores para que la nutricionista pudiera hacerse una idea más clara del aspecto final de la página.

Este proceso de diseño visual no solo ayudó a que la nutricionista comprendiera mejor la estructura de la página web, sino que también contribuyó a que pudiera aportar sus propias ideas y sugerencias.

Toda la fase de análisis de requisitos fue crucial para generar una base sólida para el desarrollo del proyecto y asegurarse de que se cumplían todos los requerimientos de la nutricionista

Diseño y planificación (16 de marzo - 31 de marzo)

Durante la fase de diseño y planificación del Trabajo de Fin de Grado, se establecieron tareas, plazos y recursos para llevar a cabo el proyecto con éxito en su totalidad. Entre ellas, se encontraba la generación de las distintas páginas que conformarían la aplicación web, la creación de la API necesaria para mostrar la información nutricional, la generación de la lógica necesaria para llevar a cabo las diferentes operaciones, así como la creación de la comunicación necesaria entre la página web y la API.

Para poder llevar a cabo estas tareas, se establecieron plazos concretos y se definió un calendario de trabajo, que permitió ajustar las diferentes tareas a fechas

específicas. En primer lugar, se estableció el diseño general de la aplicación web, utilizando datos de prueba para poder visualizar el resultado final de forma más clara. Posteriormente, se generó la base de datos que almacenaría toda la información nutricional y los datos de los clientes, y se conectó la página web con los datos de la base de datos para que pudieran ser accedidos por el usuario. Además, se planificó la generación de tests para poder llevar un control del progreso del proyecto.

Además, se definieron los recursos necesarios para llevar a cabo el proyecto de forma efectiva. Entre ellos, se encontraba la obtención de una librería específica para mostrar los pesos en un gráfico, lo que permitió mejorar la experiencia de usuario y facilitar la interpretación de los datos nutricionales. También se evaluaron distintas herramientas que permitirían optimizar el proceso de desarrollo, seleccionando finalmente aquellas que mejor se ajustaban a las necesidades del proyecto.

Para poder cumplir con los plazos establecidos, se realizó una planificación detallada de las distintas tareas, asegurando que cada una de ellas se pudiera completar en un tiempo determinado y que se pudieran llevar a cabo en el orden adecuado. Además, se evaluó la complejidad de cada tarea para poder establecer una carga de trabajo equilibrada y asegurar que el proyecto pudiera avanzar de manera constante.

Implementación (1 de abril - 30 de abril)

Durante la fase de implementación, se llevó a cabo la programación de la página web y de la API, siguiendo los plazos establecidos en la fase de planificación o, en los casos donde fue necesario, se ajustaron para garantizar la finalización del proyecto en el tiempo previsto.

Es importante llevar un seguimiento riguroso de las tareas asignadas y realizar ajustes necesarios para asegurarse de cumplir con los plazos establecidos.

En esta fase, se llevarán a cabo las tareas definidas en la planificación, como la creación de las distintas páginas, la generación de la lógica, la comunicación entre la página web y la API, y se comenzaron a realizar los tests.

Pruebas y validación (15 de abril - 10 de mayo)

En la fase de implementación no solo se desarrolló la página web y la API, sino que también se empezaron a realizar pruebas para garantizar su correcto funcionamiento. La realización de pruebas en esta fase es fundamental para asegurar la calidad del software desarrollado y detectar errores a tiempo sin tener que realizar grandes cambios en el código.

En este caso, se utilizaron diferentes tipos de tests, como los test unitarios con PHPUnit que ofrece el framework Laravel. Estos tests consisten en probar las funciones individuales del código, verificando que devuelvan los valores esperados y que detecten errores en caso de recibir valores incorrectos.

Además, también se utilizaron tests de integración para verificar que los diferentes componentes del sistema funcionen correctamente cuando se integran. Estos tests comprueban que las partes individuales del sistema trabajan de manera correcta en conjunto.

Finalmente, se realizaron los llamados tests "*end to end*" que comprueban todo el flujo del sistema de manera completa, por ejemplo, desde que se inicia una acción en la página web hasta que se recibe la respuesta correcta por parte de la API. Esto permite detectar errores en todo el proceso y garantizar la calidad del software.

Documentación final

Durante la fase de documentación, se creó un documento Word que detalla todo el Trabajo de Fin de Grado, incluyendo la descripción del proyecto, la justificación, los objetivos, la metodología, la arquitectura, la implementación, las pruebas, los resultados obtenidos, las conclusiones finales, la guía de uso y la guía de instalación. Este documento es esencial para demostrar la validez y la calidad del proyecto realizado, y servirá como guía para futuros trabajos en el mismo ámbito.

También se preparó una presentación en PowerPoint para mostrar durante la defensa del Trabajo de Fin de Grado. Esta presentación incluyó los aspectos más destacados del proyecto, así como una visión general de los resultados obtenidos y las conclusiones extraídas.

3.2 Planificación económica

Para la realización de este proyecto se ha llevado a cabo una planificación económica que ha permitido estimar y detallar los diferentes costes involucrados en el desarrollo del mismo. En esta planificación se han tenido en cuenta todos los recursos necesarios, tanto humanos como materiales, y se ha asignado un presupuesto a cada uno de ellos.

En primer lugar, los recursos humanos son un componente fundamental del proyecto. El tiempo dedicado por el alumno se ha estimado en 120 horas a un coste de 14 euros por hora, lo que supone un total de 1.680 euros. Por otro lado, el tutor ha dedicado 30 horas a un coste de 18 euros por hora, lo que equivale a un total de 540 euros.

Concepto	Horas
Análisis	10
Diseño	15
Implementación	50
Pruebas	37
Investigación	5
Aprendizaje de nuevas tecnologías	3
TOTAL	120

En cuanto a los recursos de hardware, se ha utilizado un ordenador y periféricos para llevar a cabo el proyecto, cuyo coste se estima en aproximadamente 300 euros, teniendo en cuenta la parte proporcional de la vida útil del equipo que ha sido dedicada al proyecto.

En cuanto a los recursos de software, se han utilizado herramientas gratuitas, lo que ha permitido ahorrar en esta partida.

Por último, se han considerado otros gastos como materiales de oficina, electricidad y acceso a internet, cuyo coste se ha estimado en 150 euros.

Por otro lado, también se ha contemplado el coste de posibles imprevistos o cambios en el proyecto. Es importante tener un margen para poder hacer frente a situaciones que puedan surgir, como por ejemplo la necesidad de adquirir algún componente adicional para el ordenador, la reparación de algún equipo, etc. Por eso se ha incrementado el coste con un 10% del presupuesto total para estos casos.

En resumen, el coste total del proyecto, incluyendo recursos humanos, hardware, software y otros gastos, asciende a 2.970 euros. La planificación económica es importante para garantizar la viabilidad del proyecto y asegurar que se dispone de los recursos necesarios para llevarlo a cabo en términos de costes. Es importante llevar un control exhaustivo de los gastos en todo momento para poder hacer ajustes en caso de ser necesario.

A continuación, se presenta una tabla detallada con los costes desglosados del proyecto:

Concepto	Coste
Trabajo del alumno	1.680 €
Trabajo del tutor	540 €
Hardware	300 €
Software	0 €
Materiales de oficina y suministros	150 €
Otros gastos	300 €
TOTAL	2.970 €

4 Desarrollo y pruebas

En el presente apartado se detallará el proceso de desarrollo del proyecto, desde la planificación hasta la implementación final. En primer lugar, se describirá la estructura y el diseño de la base de datos utilizada en la aplicación web, explicando su funcionalidad y relación con otras entidades. A continuación, se destacará el código de programación más significativo e innovador utilizado en el proyecto, detallando su utilidad y aporte a la solución del problema planteado.

Además, se abordará el manejo de errores en la aplicación, detallando los métodos utilizados para detectar y solucionar los fallos que se han producido durante el desarrollo del proyecto, y se explicará cómo se han aplicado estos métodos para garantizar la calidad del producto final.

También se describirá la implementación de la API desarrollada, detallando sus funcionalidades y los procedimientos llevados a cabo para integrarla en la aplicación web.

Finalmente, se abordarán los problemas que surgieron durante el desarrollo del proyecto, explicando cómo se afrontaron y resolvieron, así como las lecciones aprendidas de cara a futuros proyectos.

4.1 *Diseño e implementación del sistema*

Base de datos

Para la gestión de los datos en la base de datos de este proyecto, se ha optado por una estrategia en la que las tablas han sido creadas de manera separada, sin conexión entre ellas ni uso de claves foráneas. Esta elección se debe a la intención de simplificar al máximo los datos que aparecen repetidos en la base de datos y evitar posibles conflictos o redundancias de información. Además, en caso de ser necesario modificar el diseño de alguna tabla en el futuro, se evita que dicha modificación afecte a otras tablas. Para establecer las relaciones de datos entre las tablas, se ha utilizado código en Laravel, lo que ha permitido una conexión adecuada y una gestión óptima de la información en la base de datos.

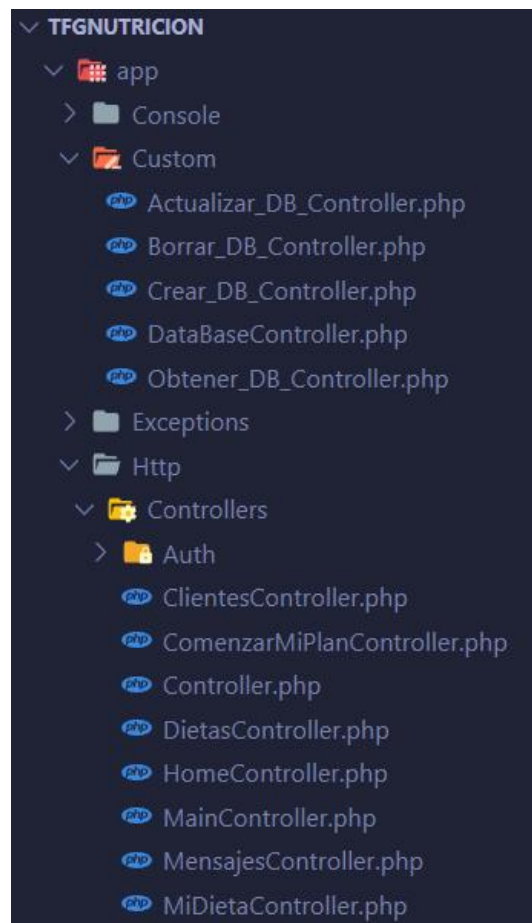
Esta estructura permite una mayor flexibilidad en el desarrollo y la posible modificación de la base de datos en el futuro, garantizando un sistema escalable y robusto. A continuación, se detallan las diferentes tablas y sus campos:

- ✧ ***contactos_externos*** se utiliza para almacenar información de contacto para personas que no son clientes. Contiene campos como nombre, teléfono, correo electrónico, fecha y mensaje.
- ✧ ***contactos_internos*** se utiliza para almacenar información de contacto para los clientes. Contiene campos como el ID del cliente, fecha y mensaje.
- ✧ ***datos_iniciales_clientes*** se utiliza para almacenar las respuestas de las preguntas de los nuevos clientes. Contiene campos como el ID del cliente, fecha, pregunta y respuesta.

- ✧ **clientes** se utiliza para almacenar los datos de los clientes. Contiene campos como el ID del cliente, nombre y apellidos, teléfono, correo electrónico, dirección, fecha de inicio, peso inicial, peso final, pérdida de peso y semanas necesarias para la pérdida de peso.
- ✧ **pesos** se utiliza para almacenar los pesos de los clientes. Contiene campos como el ID del cliente, fecha, peso, peso teórico y nota o pasos de los clientes.
- ✧ **platos** se utiliza para almacenar los platos de la dieta de los clientes. Contiene campos como el ID del cliente, acción (desayuno, comida, cena, etc.) y cada uno de los platos.
- ✧ **textos_clientes** se utiliza para almacenar los textos de recomendaciones de cada dieta de los clientes. Contiene campos como el ID del cliente, si es un texto general o específico, el texto y texto extra.
- ✧ **users** se utiliza para almacenar la información de inicio de sesión de los usuarios de la plataforma. Contiene campos como el nombre de usuario, correo electrónico, contraseña y rol (nutricionista o administrador).

Código de programación significativo

En este apartado se presentan algunas de las funciones y procedimientos más importantes que se han utilizado en el desarrollo del proyecto. Estos fragmentos de código han sido seleccionados por su complejidad y su importancia en el funcionamiento del sistema. A través de su análisis, se pretende mostrar el enfoque y la metodología de trabajo empleados en el desarrollo del proyecto.



Para la estructura general del código se ha optado por una estructura organizada y modular, en la que cada página del sitio web está separada en un controlador propio, permitiendo una mejor organización del código.

Además, se han separado las funciones de comunicación con la base de datos en clases independientes, que contienen los métodos necesarios para realizar operaciones como crear, actualizar y eliminar datos. De esta forma, cada controlador puede instanciar la clase correspondiente y hacer uso de sus métodos de manera clara y sencilla.

Esta metodología permite una mejor organización del código, al dividirlo en bloques más específicos, lo que facilita su lectura y comprensión. Además, esta estructura es muy fácil de mantener y escalar en el futuro.


```

$funcion_no_existe_conexion_db = new DataBaseController;
$no_existe_conexion_db = $funcion_no_existe_conexion_db->comprobar_no_existe_conexion_db();
if ($no_existe_conexion_db) {
    return view('error');
}

```

El control de la conexión a la base de datos es un aspecto crucial en cualquier aplicación. Es por eso que se ha implementado un fragmento de código al comienzo de cualquier comunicación con la base de datos para garantizar que la conexión sea exitosa. Este fragmento de código es esencial porque si la conexión no se establece correctamente, cualquier operación posterior que se intente realizar en la base de datos fallará.

En lugar de controlar la conexión de la base de datos en el bloque *catch* de los bloques *try-catch*, se ha decidido ponerlo antes incluso de las funciones para tener un control completo de la conexión. De esta manera, se asegura que se pueda tener una conexión a la base de datos confiable antes de ejecutar cualquier otra operación.

Además, se ha tomado en cuenta que el manejo de errores es una parte importante del control de la conexión de la base de datos. En el caso de que la conexión no se establezca correctamente, se han implementado mecanismos para manejar esta situación de manera adecuada y proporcionar al usuario una respuesta clara y precisa sobre el estado de la conexión.

```

function borrar_cliente($id)
{
    $borrado = false;

    try {
        DB::beginTransaction();
        Contacto_Interno::where('id_cliente', $id)->delete();
        Dato_Inicial_Cliente::where('id_cliente', $id)->delete();
        Cliente::where('id_cliente', $id)->delete();
        Peso::where('id_cliente', $id)->delete();
        Plato::where('id_cliente', $id)->delete();
        Texto_Cliente::where('id_cliente', $id)->delete();
        DB::commit();
        $borrado = true;
    } catch (\Exception $e) {
        DB::rollBack();
    }

    return $borrado;
}

```

Una función de gran relevancia dentro del código es la función ***borrar_cliente***, encargada de eliminar todos los datos relacionados con un cliente en particular de todas las tablas de la base de datos. Para garantizar la integridad de los datos y la seguridad de la operación, se ha utilizado una transacción en esta función.

La transacción asegura que todos los cambios realizados en las tablas sean confirmados antes de ser guardados permanentemente en la base de datos. En caso de que algún error ocurra en el proceso, todos los cambios realizados en las tablas se deshacen, devolviendo la base de datos a su estado anterior. Esto significa que sólo

se borrarán los datos del cliente cuando se verifique que se han eliminado correctamente de todas las tablas.

Dado que se trata de una operación que afecta a muchas tablas distintas, y que no existe una conexión entre ellas, se ha optado por la utilización de una transacción para tener un control exhaustivo de los datos y asegurar que se realice una operación segura y eficiente.

La función de *borrar_cliente* es fundamental para mantener la integridad de la base de datos y garantizar que los datos sean precisos y actualizados. Gracias al uso de la transacción, se logra una mayor seguridad y confiabilidad en la manipulación de los datos.

```
public function actualizar_cliente($datos_cliente, $peso_cliente, $id_cliente, $platos, $textos_clientes)
{
    $actualizado = false;

    try {
        DB::beginTransaction();

        $this->actualizar_datos_cliente($datos_cliente);
        $this->actualizar_pesos_cliente($peso_cliente);
        $this->actualizar_platos_cliente($id_cliente, $platos);
        $this->actualizar_textos_clientes($id_cliente, $textos_clientes);

        DB::commit();
        $actualizado = true;
    } catch (\Exception $e) {
        DB::rollBack();
    }

    return $actualizado;
}
```

La función ***actualizar_cliente*** tiene como objetivo actualizar los datos de un cliente en las distintas tablas de la base de datos. En este caso, se ha optado por una estructura más modular y escalable, en la que cada tabla tiene su propia función para actualizar los datos. De esta forma, se evita tener una función demasiado grande y compleja que tenga que manejar la lógica de todas las tablas.

Al igual que en la función anterior, se utiliza una transacción para garantizar la integridad de los datos en caso de que ocurra algún error durante el proceso de actualización. La transacción asegura que todas las actualizaciones se realicen correctamente y, en caso de que una de las llamadas a funciones lance un error, se deshagan todos los cambios realizados en las tablas anteriores.

Esta estrategia de dividir la lógica de actualización en funciones separadas para cada tabla tiene varias ventajas. En primer lugar, facilita la tarea de mantener el código y hacer cambios en el futuro, ya que se puede modificar la lógica de actualización de una tabla sin afectar a las demás. Además, permite una mejor organización del código y una mayor legibilidad, ya que cada función se encarga de una tarea específica y no es necesario examinar un bloque de código largo y complejo.

```
function actualizar_nuevo_peso($nuevo_dato_peso)
{
    $actualizado = false;
    try {
        if ($nuevo_dato_peso['nota_pasos'] > 0) {
            $entradas = Peso::get()->where('id_cliente', $nuevo_dato_peso['id_cliente']);
            $valor_nota_pasos = 5;
            if ($nuevo_dato_peso['nota_pasos'] > 10) {
                $valor_nota_pasos = $nuevo_dato_peso['nota_pasos'];
            }
            foreach ($entradas as $entrada) {
                if ($entrada->nota_pasos == 0 && strtotime($entrada->fecha) < strtotime($nuevo_dato_peso['fecha'])) {
                    Peso::where(['id_cliente' => $entrada->id_cliente, 'fecha' => $entrada->fecha])->update(['nota_pasos' => $valor_nota_pasos]);
                }
            }
            $peso::where(['id_cliente' => $nuevo_dato_peso['id_cliente'], 'fecha' => $nuevo_dato_peso['fecha'])
            ->update([
                'peso' => $nuevo_dato_peso['peso'],
                'nota_pasos' => $nuevo_dato_peso['nota_pasos'],
            ]);
            $actualizado = true;
        } catch (\Throwable $e) {
        }
        return $actualizado;
    }
}
```

La función ***actualizar_nuevo_peso*** resulta interesante por la complejidad de la lógica que se utiliza para actualizar los datos en la tabla *pesos* de la base de datos. En esta tabla, los clientes pueden adjudicarse una nota de 1 a 10 que indica su grado de satisfacción con la pérdida de peso de la semana o pueden registrar el número de pasos que han andado en dicha semana. Sin embargo, si el cliente decide empezar a utilizar esta función en algún punto de su proceso de pérdida de peso, se presenta un problema en la visualización de los datos anteriores.

Para resolver este problema, se ha implementado un fragmento de código que cambia los valores anteriores a 5 si el nuevo valor de "nota_pasos" es menor a 10, o cambia todos los valores al número de pasos andados si el nuevo valor es mayor a 10. De esta manera, se evita que en la gráfica de pesos del cliente aparezcan valores en cero, lo que podría desmotivar al cliente o hacer menos estética la visualización de los datos.

La inclusión de esta lógica permite mejorar la experiencia del usuario al visualizar su progreso en la pérdida de peso, al mismo tiempo que se mantiene la integridad de los datos en la base de datos. Esta solución demuestra la capacidad del código para adaptarse a las necesidades del usuario y proporcionar una experiencia más personalizada y satisfactoria.

```
private function fechas_pesos_teoricos($fecha_inicio, $peso_inicial, $peso_final_1, $peso_final_2, $perdida_peso_1, $semanas_perdida_peso_1, $perdida_peso_2, $semanas_perdida_peso_2, $perdida_peso_final)
{
    $pesos = [];
    $semana = 1;
    $peso_iterativo = $peso_inicial;
    $peso_fin = $peso_final_1;
    if ($peso_final_2 > 0) {
        $peso_fin = $peso_final_2;
    }
    $pesos[strtotime($fecha_inicio)] = intval($peso_iterativo);
    while ($peso_iterativo > $peso_fin) {
        if ($semana < $semanas_perdida_peso_1) {
            $peso_iterativo = $peso_iterativo - $perdida_peso_1 / 1000;
        } elseif ($perdida_peso_2 > 0 && $semanas_perdida_peso_2 > 0 && $semana < ($semanas_perdida_peso_1 + $semanas_perdida_peso_2)) {
            $peso_iterativo = $peso_iterativo - $perdida_peso_2 / 1000;
        } else {
            $peso_iterativo = $peso_iterativo - $perdida_peso_final / 1000;
        }
        $fecha = strtotime($fecha_inicio . "+" . $semana . " week");
        $pesos[$fecha] = round($peso_iterativo, 2);
        $semana++;
    }
    return $pesos;
}
```

La función ***fechas_peso_teoricos*** es una herramienta importante para el seguimiento del progreso del cliente. Su objetivo principal es calcular la cantidad de peso que el cliente debería perder en un determinado período de tiempo, según un rango de pérdida de peso saludable. Para esto, se utiliza un bucle *while* que itera desde el peso actual del cliente hasta el peso deseado, restando las pérdidas de peso semanales.

Es importante tener en cuenta que al principio del proceso de pérdida de peso se produce una mayor pérdida de peso que al final, por lo que esta función también tiene en cuenta este factor para proporcionar una meta de pérdida de peso realista para el cliente. Además, si el cliente llega a su peso final pero desea seguir perdiendo peso, la función también puede calcular una segunda meta de pérdida de peso manteniendo la meta inicial para mantener al cliente motivado.

Esta función es muy útil para los clientes que desean establecer metas realistas y alcanzables para su proceso de pérdida de peso. Al tener una idea clara de cuánto peso deberían perder en un período determinado, pueden ajustar su plan de pérdida de peso y mantenerse motivados al ver su progreso hacia sus objetivos.

Errores

El control de errores es una parte crucial en cualquier proyecto de software, ya que permite identificar y solucionar los posibles errores que puedan surgir durante la ejecución del programa. Sin un adecuado manejo de errores, el programa podría fallar de forma inesperada, generando resultados inesperados o incluso llegando a interrumpir su ejecución.

Por esta razón, en este apartado se abordará la importancia del control de errores en un proyecto, así como las diferentes estrategias y técnicas que se han utilizado para detectar, manejar y solucionar los errores que puedan surgir durante el desarrollo y la ejecución del programa.

Como primera medida de control, se ha procurado mantener al mínimo la cantidad de variables del código, así como limitar el número de parámetros que admiten las funciones. De esta forma, se trabaja con una cantidad de datos mutables reducida, lo que facilita la detección de posibles errores y su posterior corrección.

En segundo lugar, se han utilizado bloques *try-catch* en todo el código, con el objetivo de capturar cualquier error que pueda surgir durante la ejecución. Este tipo de estructuras permiten detectar y controlar los errores que pueden producirse en tiempo de ejecución, evitando que el sistema se detenga por completo y brindando la posibilidad de recuperación ante un fallo.

Asimismo, se ha creado una tabla en la base de datos del sistema para almacenar los errores detectados por los bloques *catch*. De esta manera, se guarda información detallada sobre el error, como su código, mensaje, archivo donde se encuentra la función, función que ha lanzado el error, línea de código donde ha ocurrido, fecha del error y página web donde ha tenido lugar. Esta información es de gran utilidad para identificar el origen del error y tomar las medidas necesarias para solucionarlo de manera efectiva.

La cuarta medida de control consiste en escribir el código en bloques mínimos, que se encargan de realizar una tarea específica y claramente definida. Esto se logra abstrayendo la idea y generando bloques de código que sean simples y eficientes. El objetivo de esta medida es tener un mejor control del ámbito de cada bloque de código y minimizar la aparición de errores.

Otra medida de control importante es la documentación del código. Se ha incluido documentación en el código para explicar el propósito de cada función, los parámetros que acepta y los valores que devuelve. Esto hace que sea más fácil para otros programadores entender el código y usarlo en el futuro.

Además, se han implementado pruebas unitarias para cada función. Las pruebas unitarias son una técnica de programación que se utiliza para probar pequeñas secciones de código, asegurando que funcionen correctamente antes de integrarlas en el código principal. De esta forma, se pueden detectar y corregir errores antes de que afecten al funcionamiento del sistema completo.

API

```
const app = express()
app.use(express.json())
const cors = require('cors');
app.use(cors());
const PORT = process.env.PORT || 3000

// Configuración de Pug para mostrar la home
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'pug');
app.use('/', renderpug);

// Funcionalidad de la API
app.use('/api/v1/', alimentosRouter)
```

La elección de crear una API REST en Typescript para el proyecto ofrece varias ventajas. En primer lugar, permite controlar los datos de nutrientes que los clientes verán en la aplicación, lo que les permite adaptarlos a sus necesidades. Las APIs externas suelen tener platos o alimentos genéricos, pero al generar los nutrientes de los platos especiales que cocina cada persona, se mejora la experiencia del usuario. Por ejemplo, un potaje en Galicia puede tener ingredientes diferentes a los de un potaje en Avila o en Irlanda, y por lo tanto los nutrientes serán distintos.

En segundo lugar, se ha elegido el uso de Typescript debido a que permite tener un control mucho más riguroso del código mientras se desarrolla. Aunque Typescript no funciona en ejecución, sino que transpila a Javascript y es el código generado el que se ejecuta, la programación en Typescript permite manejar tipos, errores, lógica, etc., que no se obtienen con el Javascript. Además, aprender Typescript es una habilidad valiosa, ya que es uno de los superset más utilizados de Javascript.

La API cuenta con una página de inicio que se renderiza mediante *PUG*. *PUG* es un motor de plantillas que se utiliza para generar HTML a partir de archivos de plantilla. Con *PUG*, es posible escribir HTML en un formato más sencillo y conciso, lo que facilita el mantenimiento y la actualización de la página. La página de inicio muestra las funcionalidades disponibles actualmente, que solo incluyen funciones GET, pero se está planificando en un futuro incluir funciones POST, DELETE y PATCH para modificar los datos de la API sin tener que actualizar el archivo de datos.

Otra ventaja de utilizar una API REST es que es fácilmente escalable. Si la aplicación tiene éxito y empieza a recibir un gran número de solicitudes, se puede añadir más potencia de procesamiento para el backend. Por ejemplo, se pueden añadir más servidores o cambiar a un servicio en la nube. Además, una API REST es independiente del lenguaje o la plataforma utilizados en el frontend, lo que permite crear aplicaciones móviles o de escritorio que se comuniquen con el backend.

Problemas durante el desarrollo

Durante el desarrollo del proyecto, surgieron varios problemas que retrasaron la finalización de algunas tareas del cronograma inicial. Uno de los principales problemas se presentó al intentar consumir los datos de la API en la página web. En un principio, se pensaba que el problema era de IPs debido a que se trabajaba siempre sobre *localhost*, se realizaron pruebas con APIs externas y se obtenía la información correctamente así que se decidió entonces alojar la API en un servicio de deploy gratuito. En estos servicios de alojamiento se tuvieron problemas al desconocer que primero había que instalar las dependencias (*npm install*) y luego había que transpilar el código Typescript a Javascript. Una vez solucionado este problema de deploy, al volver a intentar consumir la API desde la página web, esta vez de forma externa, tampoco se pudo debido a que no estaba instalado CORS (*Cross-Origin Resource Sharing*), que es un mecanismo de seguridad que impide que una página web acceda a recursos de un dominio diferente. Para resolver este problema, se instaló la dependencia de CORS en la API, lo que permitió consumir los datos sin problema desde la página web.

Otro problema importante que surgió durante el desarrollo fue el diseño de la base de datos. En un principio, se guardaban los datos en forma de JSON en cada columna, lo que hacía que fuese muy complicada la lectura y guardado de los datos. Después de consultarlo con el tutor, se aconsejó que se separase cada dato en una entrada distinta, ya que era más sencillo recuperar todas las entradas buscadas que recuperar una y trabajar con JSON. Esta solución permitió una mejor gestión de los datos y un acceso más eficiente a ellos.

Una complicación que se presentó durante el desarrollo fue el diseño visual general de la página web. Al no tener un conocimiento detallado de diseño de interfaces, se invirtió una cantidad considerable de tiempo en este aspecto. Sin embargo, el uso de TailwindCSS, resultó de gran ayuda para tener unas bases sólidas de un buen diseño visual de la web. Con la ayuda de Tailwind, se logró una apariencia más atractiva y funcional para la página web.

Uno de los desafíos que se experimentó en el proceso de programación fue la alteración de los datos de las distintas tablas de la base de datos. Debido a que las tablas eran independientes, si se realizaba una alteración en una tabla y surgía un problema con otra tabla que no registraba el cambio, la aplicación fallaba. Para solucionar este problema se utilizó el concepto de transacciones, que permitía tener los cambios en *stand by* hasta que todas las tablas fueran alteradas correctamente. En ese momento, se confirmaban los cambios. Este enfoque permitió evitar una gran cantidad de errores en la aplicación.

Otro problema que se encontró durante el desarrollo del proyecto fue con el login en Laravel. Si bien la funcionalidad del login funciona correctamente, si se quisiera ampliar su uso o complejidad en el futuro, habría que cambiar varias cosas. Por ejemplo, si se quisiera tener distintos roles para los usuarios registrados, como un nutricionista o un administrador, la base de datos está preparada para ello y el código de Laravel también está diseñado para mostrar unos datos u otros dependiendo del

rol, sin embargo, los controladores de la autenticación, la recuperación de contraseñas, entre otros, no están correctamente implementados. Por lo tanto, en el futuro se tendría que investigar más a fondo para solucionar este problema. Actualmente, por cuestiones de tiempo y por no tener ese requerimiento, no se profundizó en la implementación de distintos roles de usuario.

Por último, una complicación que se presentó fue a la hora de traspasar los datos de PHP a Javascript. Esto se debió a que cada lenguaje de programación tiene estructuras distintas a la hora de utilizar los datos. Afortunadamente, Laravel cuenta con herramientas que ayudan en la transferencia de datos entre los dos lenguajes de programación. De esta forma, se pudo solucionar este problema de manera efectiva.

En conclusión, durante el desarrollo del proyecto se presentaron varios problemas que retrasaron la finalización de algunas tareas, pero se lograron solucionar con la ayuda de tutorías y la investigación de nuevas herramientas y tecnologías. Estos problemas fueron una oportunidad para aprender y mejorar los conocimientos y habilidades en el desarrollo de futuras aplicaciones web.

4.2 Pruebas del sistema

En el proceso de desarrollo de aplicaciones web, es esencial llevar a cabo pruebas del sistema para garantizar que la aplicación se ejecuta correctamente y cumple con los requisitos especificados. Las pruebas del sistema se realizan en diferentes niveles: pruebas unitarias, de integración y de extremo a extremo (*end to end*).

```
✓ url pagina inicio 0.24s
✓ url pagina mi dieta 0.03s
✓ url pagina comenzar mi plan 0.03s
✓ url pagina clientes 0.05s
✓ url pagina dietas 0.03s
✓ url pagina mensajes 0.09s

Tests: 6 passed (6 assertions)
Duration: 0.59s
```

Las **pruebas unitarias** se centran en comprobar que cada componente individual de la aplicación funciona correctamente. Estas pruebas se llevan a cabo sobre fragmentos de código aislados, como funciones, métodos y clases. El objetivo principal de las pruebas unitarias es detectar errores en la lógica del código antes de la integración en el sistema completo. Esto permite a los desarrolladores detectar y corregir los errores de manera más temprana en el ciclo de desarrollo, lo que reduce significativamente el costo y el tiempo requerido para su resolución.

Para llevar a cabo las pruebas unitarias, se ha utilizado *PHPUnit*, una herramienta de prueba de código abierto para PHP y Laravel. Esta herramienta es ampliamente utilizada en el desarrollo de aplicaciones web y ofrece una amplia variedad de funciones de prueba, como comprobaciones de igualdad, verificaciones de excepciones, entre otras.


```
/** @test */  
public function testObtenerContactoInterno()  
{  
    // Crear un mensaje interno de prueba  
    Contacto_Interno::factory(1)  
        ->create(['id_cliente' => 'aa999']);  
    // Obtener el registro creado  
    $registro = Contacto_Interno::get()  
        ->where('id_cliente', 'aa999')->first();  
    // Verificar que el mensaje haya sido creado  
    $this->assertEquals('aa999', $registro->id_cliente);  
}
```

A modo de ejemplo, se ha desarrollado un test unitario que comprueba el correcto funcionamiento de una función específica. Este test unitario se ejecuta de forma aislada, lo que significa que no interactúa con otras partes de la aplicación. Al ejecutar esta prueba, se comprueba que la función en cuestión se comporta de la manera esperada y que no existen errores o fallos en su lógica.

Las **pruebas de integración** son esenciales para garantizar la calidad de la aplicación en su conjunto. Si bien las pruebas unitarias comprueban la correcta implementación de cada parte del código por separado, las pruebas de integración aseguran que todos los componentes de la aplicación se comporten adecuadamente cuando se utilizan juntos.

Además, las pruebas de integración se enfocan en identificar errores en la comunicación y la interoperabilidad entre los distintos módulos de la aplicación. Esto es especialmente importante en aplicaciones complejas, donde múltiples componentes pueden interactuar de maneras muy diferentes.

Para llevar a cabo las pruebas de integración, se utilizan distintas técnicas y herramientas, como la simulación de distintos escenarios de uso, la generación de datos de prueba y la monitorización del rendimiento y la funcionalidad de la aplicación.

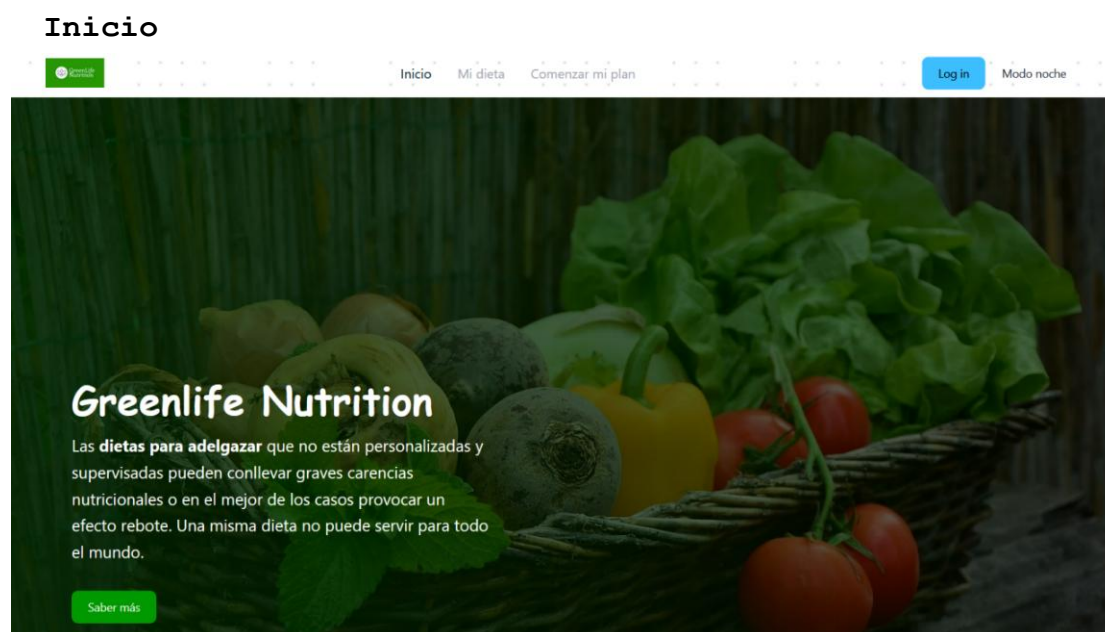
Es importante destacar que las pruebas de integración deben realizarse después de las pruebas unitarias, ya que es necesario asegurarse de que cada componente individual funciona correctamente antes de comprobar cómo funcionan juntos.

Las **pruebas de extremo a extremo** (*end to end*) son una parte crucial en el proceso de pruebas de software, ya que permiten comprobar la funcionalidad y la usabilidad de la aplicación en un entorno similar al de producción. Estas pruebas se realizan después de las pruebas de integración y antes de la entrega final del software al cliente. El objetivo principal de las pruebas de extremo a extremo es verificar que la aplicación funciona como se espera y cumple con los requisitos del usuario final.

Durante estas pruebas se simulan situaciones reales de uso del usuario final, y se comprueba que la aplicación es capaz de manejarlas adecuadamente. Se prueban todas las funcionalidades de la aplicación, desde el inicio de sesión hasta las funciones más avanzadas. Se verifica que los distintos componentes de la aplicación trabajan correctamente juntos y que el flujo de trabajo es el adecuado.

5 Descripción de la aplicación

La página web constará de seis secciones principales: la página *Inicio*, la página *Comenzar mi plan*, la página *Mi dieta*, la página *Cientes*, la página *Diets* y la página *Mensajes*.




La página *Inicio* es una sección fundamental en la página web de la nutricionista, ya que será la primera sección que vean los usuarios al acceder al sitio. Esta sección debe proporcionar información clara y detallada sobre la nutricionista y sus servicios.

En la página de inicio se incluirá una breve biografía de la nutricionista, donde se describirán su experiencia, formación y enfoque en cuanto a la nutrición. Esta información permitirá a los usuarios conocer mejor la profesionalidad y la calidad de los servicios que ofrece la nutricionista.

Además, se incluirán testimonios de clientes satisfechos para demostrar la efectividad y calidad de los planes de nutrición ofrecidos por la profesional. Otro elemento importante en la página *Inicio* es la lista de precios para los diferentes planes ofrecidos por la nutricionista. Debe ser clara y fácil de entender, y se deben proporcionar detalles sobre los servicios que incluye cada plan y su precio correspondiente.

Finalmente, para que los usuarios puedan contactar fácilmente con la nutricionista, se incluirá un formulario de contacto en la página de inicio. Este formulario permitirá a los usuarios enviar sus preguntas y comentarios directamente a la nutricionista.

Comenzar mi plan

 Inicio Mi dieta Comenzar mi plan Clientes Dietas Mensajes Logout [Modo noche](#)

¿Se ha puesto anteriormente a dieta? ¿Cosas positivas que te aportaron esas dietas?

Responda aquí

¿Tienes el hábito de desayunar regularmente? ¿Motivo? Indica qué desayunos suele hacer.

Responda aquí

¿Tienes el hábito de comer algo a media mañana regularmente? ¿Motivo? Indica qué suele hacerte.

Responda aquí

¿Te gusta picar entre horas? ¿Qué picoteas?

Responda aquí

¿Llegas con ansiedad a alguna de la tomas? ¿A cuál?

Responda aquí

¿Te gustan los alimentos muy salados? ¿Qué tipo de sal usa?

Responda aquí

Enviar

La página *Comenzar mi plan* es una sección crucial donde los clientes proporcionan su información de hábitos para que la nutricionista pueda crear un plan de dieta personalizado para ellos.

Se requerirá que el cliente proporcione información detallada y precisa sobre sus hábitos alimenticios, estilo de vida y preferencias dietéticas para que la nutricionista pueda crear una dieta efectiva y adecuada a sus necesidades individuales. La nutricionista también podría solicitar información adicional, como alergias alimentarias, enfermedades crónicas y medicamentos que toma el cliente para crear un plan de dieta específico para su caso.

Es importante destacar que la sección estará protegida por medidas de seguridad y requerirá un código personal para acceder a las preguntas del formulario. Esto garantizará la privacidad y confidencialidad de los datos del cliente. Una vez que el cliente complete el formulario y envíe sus respuestas, la nutricionista podrá acceder a él para comenzar a trabajar en un plan de dieta específico y adaptado a sus necesidades.

Mi dieta



La sección *Mi dieta* es fundamental para que el cliente pueda hacer seguimiento de su progreso en la pérdida de peso y para que la nutricionista pueda ajustar el plan de dieta personalizado según los resultados obtenidos.

En esta sección se presentará información detallada y personalizada sobre la dieta y recomendaciones nutricionales específicas para el cliente. Además de un gráfico con la evolución de su peso, se incluirán datos sobre el contenido nutricional de cada plato, y se le brindará una lista de opciones para su plan de alimentación.

Es importante destacar que la información proporcionada en esta sección es confidencial y estará protegida por la política de privacidad de la página web. Para acceder a ella, el cliente deberá proporcionar su código de cliente, el cual se le asignó en el momento de su alta en el sistema.

Cientes



En la página *Clientes* la nutricionista tendrá acceso a información confidencial y personal de sus pacientes. Por esta razón, es importante que solo tenga acceso a ella a través de un inicio de sesión seguro y privado.

En esta sección, la nutricionista podrá actualizar la información del cliente, como agregar nuevos pesos y modificar los antiguos. Esta información será útil para evaluar el progreso del paciente y hacer ajustes necesarios en su plan de alimentación para garantizar que se cumplan sus objetivos de pérdida de peso.

Dietas

Inicio Mi dieta Comenzar mi plan Clientes Dietas Mensajes Logout Modo noche

Jose Luis Panadero Gutierrez Borrar cliente

Id Cliente Nombre y apellidos Telefono

j13864 Jose Luis Panadero Gutierrez 926 901 067

Email Direccion

jose.luis@gmail.com Plaza Eduardo, 3, 0º A

Fecha inicio Peso inicial (Kg) Peso final 1 (Kg) Peso final 2 (Kg)

09 / 04 / 2022 98 90 82

Perdida de peso inicial (g) Semanas Perdida de peso despues (g) Semanas Perdida de peso al final (g)

700 4 500 3 400

Mostrar textos

Mostrar platos

La página *Dietas* es una de las secciones más importantes de la plataforma. En esta sección, la nutricionista puede realizar una serie de acciones sobre los datos de los clientes.

En primer lugar, puede dar de alta nuevos clientes en el sistema y tener acceso a las respuestas del cuestionario de la página *Comenzar mi plan*. De esta manera, la nutricionista puede conocer los hábitos alimentarios, las preferencias y las necesidades de cada cliente y crear un plan de alimentación adaptado a sus características.

Además, en esta página, la nutricionista puede modificar los datos del cliente si se produce algún cambio en su situación personal o de salud. También puede crear nuevas dietas o modificar las existentes en función de la evolución de cada cliente. Por ejemplo, si el cliente no ha perdido una cantidad significativa de peso, la nutricionista puede ajustar el plan de alimentación para asegurarse de que sigue siendo efectivo.

Otro aspecto importante de la página es que la nutricionista puede crear, eliminar o modificar las recomendaciones nutricionales personalizadas para cada paciente. Estas recomendaciones pueden incluir sugerencias de alimentos saludables, consejos para controlar el apetito o recomendaciones de ejercicios físicos para complementar el plan de alimentación. De esta manera, la nutricionista puede ofrecer un servicio completo, personalizado y continuo a lo largo del tiempo a cada uno de sus clientes.

Por último, la página *Dietas* permite a la nutricionista borrar a los pacientes que hayan alcanzado su objetivo de pérdida de peso. De esta manera, puede mantener un control sobre los clientes que siguen activos en el programa y enfocarse en ofrecer un servicio completo y de calidad.

Mensajes



[Inicio](#) [Mi dieta](#) [Comenzar mi plan](#) [Clientes](#) [Dietas](#) [Mensajes](#)

[Logout](#) [Modo noche](#)



Jose Luis Panadero Gutierrez

15-03-2023 22:15:38

Hola, tengo una duda sobr

Jose Luis Panadero Gutierrez

j13864





Ariadna Sancho Tercero

25-03-2023 07:05:03

Hola America, estoy sigui



Jose Luis Panadero Gutierrez

03-04-2023 15:32:31

Hola, necesito tu ayuda c



Oliver

04-04-2023 09:57:08

Hola, estoy interesado/a



Gust

28-02-2023 17:45:59

Buen día, estoy buscando

Luna

25-03-2023 21:15:016

Hola, me llamo Luna y me

15-03-2023 22:15:38

Hola, tengo una duda sobre mi dieta. He notado que me siento con menos energía por las mañanas y quería saber si hay algún alimento que pueda incluir en mi desayuno para sentirme mejor. ¿Podrías ayudarme?

La página de *Mensajes* permite una comunicación eficiente y directa entre la nutricionista y sus clientes. Además, también es una herramienta útil para aquellos usuarios externos que deseen hacer consultas o pedir información sobre los servicios de la nutricionista.

En esta sección, la nutricionista tendrá acceso a los mensajes recibidos y podrá responder de manera rápida y eficaz a las preguntas y solicitudes de los clientes y usuarios externos. De esta manera, se puede establecer un canal de comunicación constante y fluido entre la nutricionista y sus clientes, lo que puede mejorar la calidad del servicio y la satisfacción del cliente.

La página también puede ser una herramienta útil para recibir comentarios y sugerencias de los clientes, lo que puede ayudar a mejorar los servicios y ajustar los planes de alimentación para satisfacer mejor las necesidades de los clientes.

6 Conclusiones finales

En este apartado, se presentan las reflexiones y consideraciones finales del proyecto, se resumen los resultados obtenidos y se discuten las implicaciones y limitaciones del mismo. Además, se propondrán posibles mejoras o ampliaciones futuras. En este sentido, en las siguientes secciones se presentan las conclusiones obtenidas en el desarrollo de la aplicación web de seguimiento de dietas de clientes para nutricionistas, así como las posibles líneas de mejora y ampliación para futuras versiones.

6.1 Grado de cumplimiento de los requisitos fijados

En este apartado se analizará en qué medida el proyecto desarrollado cumple con los requisitos y objetivos establecidos al comienzo del mismo. Se examinará cada uno de los requisitos y se determinará si se han cumplido en su totalidad o si existen deficiencias. También se evaluará la calidad y funcionalidad de la aplicación desarrollada en relación a los objetivos propuestos.

Requisito	Grado de cumplimiento	Observaciones
Cumplir todos los requisitos de la nutricionista	100%	Todos las peticiones de la nutricionista se han cumplido
Diseño bonito	70%	Diseño mejorable
Landing page con toda la información para atraer clientes nuevos	100%	Se han cumplido los requisitos técnicos
Mostrar a los clientes su dieta personalizada	70%	Puede mejorar técnica y estéticamente
Mostrar a los clientes un gráfico con su peso	100%	Se muestran de forma sencilla y atractiva
Crear un formulario con preguntas iniciales para los nuevos clientes	100%	Se ha cumplido con el requisito técnico y es sencillo para los clientes
Login de la nutricionista y del administrador web	60%	Debe mejorar en el futuro
Introducción de los pesos de los clientes	100%	Se ha cumplido el requisito técnico
Alta de nuevos clientes	90%	Se ha cumplido el requisito pero puede mejorar estéticamente
Calculo del peso teórico de los clientes	100%	El algoritmo es eficiente y sencillo
Introducción sencilla de los datos, dietas, pesos y textos de los clientes	90%	La introducción de datos es sencilla pero puede mejorar estéticamente
Modificar los datos de los clientes	90%	Método sencillo pero puede mejorar estéticamente

Requisito	Grado de cumplimiento	Observaciones
Mensajería interna y externa independientes	100%	Requisito técnico y estético cumplido
Mostrar a los clientes los valores nutricionales de sus platos	80%	Se ha cumplido el requisito pero debe mejorar para platos mas complejos
Guardado de errores	100%	Persistencia de errores que se puede consultar para solucionarlos
Código limpio	80%	Se ha intentado mantener un código limpio pero puede mejorar en muchos aspectos

6.2 Propuesta de mejora o ampliaciones futuras

En el presente apartado se expondrán posibles propuestas de mejora o ampliaciones futuras para la aplicación desarrollada. Se analizarán posibles funcionalidades adicionales o mejoras en la usabilidad de la aplicación que podrían implementarse para aumentar su valor y satisfacer mejor las necesidades de los usuarios.

En primer lugar, se puede trabajar en **mejorar el diseño estético** de la aplicación, para hacerla más atractiva y moderna. Esto podría lograrse contratando a una diseñadora web que mejore la paleta de colores utilizada y la interfaz de usuario.

Otra posible mejora sería **mejorar la forma en que se muestran los platos** de la dieta a los clientes. En la aplicación actual, los platos se presentan en una lista simple, lo que puede resultar poco atractivo y poco intuitivo para algunos usuarios. Se podría explorar la posibilidad de implementar un sistema de visualización de platos más atractivo y fácil de usar, quizás mediante el uso de imágenes y descripciones detalladas.

También se puede trabajar en la **mejora del login** con distintos roles, en el que cada usuario tenga funciones específicas y separadas según su rol en la aplicación. Por ejemplo, los nutricionistas podrían tener acceso para actualizar los datos de nutrientes de los alimentos de la API, mientras que los administradores podrían tener acceso para gestionar los usuarios y los datos de la base de datos.

Otra posible **mejora sería permitir la creación de platos más complejos** en la dieta. Actualmente, la aplicación solo permite la selección de platos simples como tostadas, pero se podría ampliar la funcionalidad para permitir la selección de platos más elaborados, como tostadas con tomate y aguacate, para ofrecer más opciones a los usuarios.

En cuanto a la calidad del código, se podría trabajar en la **mejora del código** para que sea más cercano a *código limpio*, eliminando redundancias y mejorando la legibilidad del código. Esto puede hacer que el código sea más fácil de entender y mantener en el futuro.

Además, se podría **mejorar la gestión de errores** en la aplicación. Actualmente, los errores están guardados en una tabla en la base de datos, pero solo se pueden ver entrando manualmente en la base de datos. Se podría mejorar esto para que los errores se muestren de forma más accesible y fácilmente identificable para los administradores.

Por último, se podría **implementar la posibilidad de agregar, borrar y modificar nutrientes de los alimentos de la API** desde el rol de administración. Actualmente, solo se permite la visualización de los nutrientes de los alimentos de la API, pero esta funcionalidad podría ampliarse para permitir una mayor interacción con la API y ofrecer más control y flexibilidad a los usuarios.

En conclusión, el proyecto ha sido un gran éxito en términos de cumplimiento de los objetivos fijados, y ha conseguido satisfacer las necesidades de la nutricionista y de los usuarios finales. Sin embargo, siempre hay margen para la mejora y la evolución, y se han identificado diversas áreas en las que se podría trabajar para ampliar y mejorar la funcionalidad de la aplicación. Con las mejoras propuestas, la aplicación podría convertirse en una herramienta aún más valiosa y efectiva para los nutricionistas y sus clientes.

7 Guías

7.1 Guía de instalación

La guía de instalación es un paso fundamental para poder utilizar la página web desarrollada en Laravel. A continuación, se presentan los pasos necesarios para la instalación de la página web.

Requisitos previos

Es necesario contar con un servidor web con soporte para PHP y una base de datos relacional. Se recomienda la instalación de Xampp que dispone de Apache como servidor web y de MariaDB como base de datos relacional.

Descarga del código fuente

El primer paso es descargar o clonar el código fuente de la página web desde el repositorio de Github mediante el siguiente enlace:

<https://github.com/NBCharro/TFGnutricion>

Una vez descargado, se debe descomprimir el archivo en una carpeta.

Configuración del archivo .env

El archivo .env es el archivo de configuración principal de la aplicación. En este archivo se deben especificar los datos de conexión a la base de datos y la URL de la API rest.

Instalación de las dependencias

Para poder utilizar la página web es necesario instalar las dependencias de Laravel y de NodeJS. Para ello, se debe abrir una consola en la carpeta de la aplicación y ejecutar el comando “\$ composer install” y “\$ npm install”. Estos comandos descargarán todas las dependencias de Laravel y de NodeJS necesarias para que la aplicación funcione correctamente.

Configuración de la base de datos

Una vez instaladas las dependencias de Laravel, es necesario configurar la base de datos. Para ello, se debe ejecutar el comando “\$ php artisan migrate”. Este comando creará todas las tablas necesarias en la base de datos para que la aplicación funcione correctamente.

En el enlace se incluyen datos de prueba que pueden ser útiles para aquellos que deseen utilizar la aplicación sin contar con datos de clientes propios. Estos datos se encuentran en el archivo *todasTablasDB.sql* que se encuentra dentro de la carpeta *datos*.

Ejecución de la aplicación

Una vez configurada la base de datos, es posible ejecutar la aplicación.

Para ello, se debe ejecutar el comando “\$ php artisan serve”. Este comando iniciará un servidor web local que permitirá acceder a la página web desde el navegador.

Además, y sin cerrar la terminal anterior se debe ejecutar el comando “\$ npm run dev” que permitirá lanzar Vite para compilar los archivos JavaScript y CSS y generar los archivos necesarios para su correcto funcionamiento en el navegador.

Verificación de la conexión con la API rest

Una vez iniciada la aplicación, es necesario verificar la conexión con la API rest. Para ello, se debe acceder a la sección de la página web que se conecta con la API rest y realizar una prueba de conexión. La página web de la API es:

<https://api-nutricion.onrender.com/>

Ejecutar localmente la API rest

Si se desea utilizar la página web de la API en un ambiente local, es necesario descargarla desde el siguiente enlace:

<https://github.com/NBCharro/apiNutricion>

Una vez descargado el archivo, se debe descomprimir el archivo o clonar el repositorio y acceder a la carpeta. A continuación, se deben ejecutar los siguientes comandos: "`$ npm install`" y "`$ npm run tsc`". El primer comando descargará los paquetes necesarios y el segundo transpilará el código de Typescript en Javascript.

Una vez que se hayan ejecutado estos comandos, se generará la carpeta *build* con el código Javascript. Luego, se debe acceder a la carpeta */build/src* y ejecutar el siguiente comando: "`$ node index.js`" para poner en marcha el servicio de la API en el ambiente local.

Es importante tener en cuenta que se debe modificar la URL que aparece en el archivo */public/midieta/modalAPI.js* para que apunte a la API local y permita el consumo de la misma.

Despliegue en producción

En caso de que se desee desplegar la aplicación en un servidor web en producción, es necesario configurar adecuadamente el entorno, asegurándose de tener las dependencias necesarias instaladas y configurar adecuadamente el servidor web y la base de datos. Además, se debe configurar el archivo de entorno *.env* con los valores correspondientes a la configuración del servidor y la base de datos en producción.

Siguiendo estos sencillos pasos es posible instalar y hacer funcionar la página web desarrollada en Laravel y la API desarrollada en Typescript en un ambiente local. De esta manera, se podrá acceder a la página web y utilizar todas sus funcionalidades. Es importante tener en cuenta que, en caso de presentarse algún problema durante la instalación, se debe revisar cuidadosamente la configuración del archivo *.env* y la conexión con la base de datos y la API rest.

7.2 Guía de uso

En este apartado se presentará una guía de uso detallada para la página web de Laravel, con el objetivo de que los usuarios puedan entender cómo utilizar la plataforma de manera eficiente y sacar el máximo provecho de sus funcionalidades. Se describirán los pasos necesarios para acceder a las páginas, crear y gestionar los clientes, visualizar la información sobre la dieta y los platos, y hacer uso de las herramientas de mensajería.

Página principal

En la página principal, los usuarios pueden encontrar fácilmente los enlaces a las secciones *Comenzar mi dieta*, *Mi dieta* y *Login*. Además, se ha agregado un botón de alternancia para cambiar entre el modo oscuro y el modo claro para la comodidad del usuario.

En la misma página, se muestra información detallada de la nutricionista y se incluye un formulario de contacto para que los usuarios puedan enviar un mensaje directamente a ella. En el formulario de contacto es obligatorio proporcionar el nombre, correo electrónico y el mensaje, mientras que el campo de teléfono es opcional. Con esta función, los usuarios pueden comunicarse con la nutricionista fácilmente y obtener asesoramiento personalizado en caso de tener preguntas o inquietudes sobre su dieta o la aplicación.

Comenzar mi plan

La página *Comenzar mi plan* es el primer paso para iniciar una dieta personalizada y adaptada a las necesidades y objetivos de cada cliente. Antes de la entrevista personal con la nutricionista, se debe responder a una serie de preguntas iniciales que ayudarán a recopilar información relevante sobre los hábitos personales y alimenticios de cada cliente.

Para acceder a estas preguntas, el cliente debe introducir el código que se le proporcionó al comenzar la dieta en el campo correspondiente. Una vez introducido el código, aparecerán las preguntas y un campo al lado para responderlas. Es importante destacar que no es necesario responder todas las preguntas de una vez, ya que se pueden modificar las respuestas en otra ocasión.

Al hacer clic en el botón *Enviar*, las respuestas se guardarán y podrán ser consultadas en cualquier momento introduciendo el código del cliente. Si el cliente desea realizar cambios en las respuestas, solo debe modificar los campos correspondientes y hacer clic en *Enviar* nuevamente para guardar las nuevas respuestas.

Estas preguntas permitirán a la nutricionista recopilar información relevante para poder elaborar un plan de dieta personalizado y adaptado a las necesidades de cada cliente. Además, el hecho de poder modificar las respuestas en cualquier momento garantiza que la información esté actualizada y que se ajuste a los cambios en los hábitos personales y alimenticios de cada cliente.

Mi dieta

La página *Mi dieta* es una de las secciones más importante de la aplicación ya que es donde el usuario podrá acceder a su dieta personalizada y su historial de peso. Para acceder a esta sección, el usuario deberá introducir el código que se le proporcionó al comenzar la dieta en el campo correspondiente.

Una vez introducido el código, el usuario tendrá acceso a una gran cantidad de información relevante para su dieta. En primer lugar, se le mostrará un gráfico con su historial de peso. Este gráfico incluirá varias funciones, como el peso teórico de pérdida que aparecerá como una línea ancha, el peso real perdido, una línea horizontal con el peso final y una segunda línea horizontal si el usuario se ha propuesto llegar a un nuevo peso final inferior. Además, el gráfico puede incluir una función con la nota de pérdida de peso de esa semana o los pasos que ha andado el usuario.

Además, el usuario también tendrá acceso a su dieta diaria dividida por tramos horarios, como desayuno, merienda y cena. El usuario podrá seleccionar el tramo horario que desee desde un selector y se le mostrará la dieta correspondiente. Si el usuario hace clic en uno de los platos de la dieta, se le mostrará un modal con información detallada sobre los nutrientes del alimento seleccionado.

Además de la dieta y el historial de peso, la página *Mi dieta* también incluirá recomendaciones personalizadas para el usuario. Estas recomendaciones estarán diseñadas específicamente para ayudar al usuario a alcanzar sus objetivos de pérdida de peso de manera efectiva y saludable.

Login

En la página principal del sitio web se encuentra el botón *Login* que permite a los usuarios registrados acceder a la sección de administración. Para ello, deberán introducir su dirección de correo electrónico y su contraseña en los campos correspondientes del formulario de inicio de sesión y hacer clic en el botón *Conectarse*.

Una vez autenticado, el usuario tendrá acceso a diversas herramientas de administración, como la capacidad de gestionar los datos de los clientes o ver los mensajes enviados, entre otras opciones. Es importante destacar que estas herramientas de administración solo están disponibles para los usuarios registrados y autenticados en el sistema.

La seguridad es una prioridad, por lo que hemos implementado medidas de protección de la información de los usuarios, como la encriptación de las contraseñas. Asimismo, recomendamos a los usuarios utilizar contraseñas seguras y cambiarlas regularmente para garantizar la protección de su información personal.

Cientes

Una vez logueado en el sistema, el usuario tendrá acceso a la página *Cientes* donde encontrará un menú desplegable con todos los clientes activos del sistema. Al

seleccionar uno de ellos, se mostrará la misma información que en la página *Mi dieta*, lo que permitirá a la nutricionista o al administrador ver la evolución en el peso del cliente, los platos de la dieta y las recomendaciones correspondientes.

Además, se incluye un apartado para introducir un nuevo peso y nota o pasos, o modificar los datos de peso y nota o pasos anteriores. Para hacerlo, el usuario deberá elegir la fecha correspondiente en el menú desplegable y luego ingresar los nuevos datos en el campo de entrada correspondiente. Una vez hecho esto, deberá hacer clic en el botón *Guardar* para que los cambios se guarden en el sistema.

Cabe destacar que la página *Cientes* está diseñada específicamente para que la nutricionista o el administrador puedan monitorear de cerca la evolución de los clientes y, de esta manera, ajustar la dieta y las recomendaciones según sea necesario para lograr los objetivos de pérdida de peso.

Dietas

La página *Dietas* ofrece una serie de herramientas para que los nutricionistas y administradores puedan gestionar la información de sus clientes de forma eficiente. Dividida en dos zonas principales, *Nuevo cliente* y *Modificar cliente*, ofrece una amplia gama de opciones para dar de alta y gestionar la información de los clientes.

En la zona de *Nuevo cliente*, podrán dar de alta a un nuevo cliente introduciendo sus datos personales, incluyendo el ID del cliente que se calculará externamente, el nombre y apellidos, la fecha de inicio, el peso inicial y el peso final, que son obligatorios. También hay campos opcionales, como el teléfono, el correo electrónico, la dirección y el peso final 2. Además, se puede agregar preguntas adicionales para que el cliente las responda en la página *Comenzar mi plan*. Una vez introducidos los datos, se debe hacer clic en el botón *Guardar cliente nuevo* para registrarlos en el sistema.

Por otro lado, la zona de *Modificar cliente* permite acceder a todos los datos del cliente y modificarlos según sea necesario. Hay cinco secciones diferentes para editar: datos personales, pérdida de peso semanal, recomendaciones para el cliente, platos y preguntas y respuestas. Al hacer clic en un cliente específico en el menú desplegable, se puede acceder a su información y modificarla según sea necesario. Es importante destacar que una vez que se han realizado las modificaciones, es necesario guardarlas en la base de datos haciendo clic en el botón *Guardar cliente*.

Por último, en la parte superior de la zona de *Modificar cliente* se encuentra el botón *Borrar cliente*, que elimina todos los datos del cliente del sistema. Sin embargo, es importante tener en cuenta que esta acción es irreversible y que hay que tener precaución al utilizarla.

Mensajes

La página *Mensajes* es una herramienta fundamental para la comunicación entre la nutricionista y los clientes. Es importante destacar que esta página se divide en dos secciones: la primera sección muestra una lista de los mensajes recibidos, donde los

mensajes enviados por clientes estarán identificados con un icono de una casa, mientras que los mensajes de usuarios externos no tendrán ningún icono identificativo. En esta sección los mensajes disponen una previsualización del remitente y del mensaje, así como un icono de verificación que indica si el mensaje ha sido leído previamente.

La segunda sección de la página muestra el mensaje completo con todos los detalles. En esta sección, la nutricionista puede revisar el mensaje completo y eliminarlo mediante el icono de la papelera. Es importante mencionar que los mensajes eliminados no se podrán recuperar, por lo que se debe tener precaución al realizar esta acción.

Es esencial que la nutricionista revise regularmente la página *Mensajes* para asegurarse de que todas las comunicaciones de los clientes y usuarios externos sean atendidas de manera oportuna y eficiente. Además, es recomendable responder los mensajes lo antes posible para brindar un mejor servicio al cliente.

8 Referencias bibliográficas

W3Schools. (s. f.). HTML Tutorial. Recuperado el 1 de mayo de 2023, de <https://www.w3schools.com/>

Mozilla Developer Network. (s. f.). Web technologies. Recuperado el 1 de mayo de 2023, de <https://developer.mozilla.org/en-US/>

Laravel. (s. f.). The PHP Framework for Web Artisans. Recuperado el 31 de abril de 2023, de <https://laravel.com/>

Stack Overflow. (s. f.). Stack Overflow - Where Developers Learn, Share, & Build Careers. Recuperado el 12 de abril de 2023, de <https://stackoverflow.com/>

Módulo Desarrollo Web Entorno Servidor. (s. f.). IES San Andres. Recuperado el 1 de mayo de 2023, de <https://aulavirtual.educa.jcyl.es/iessanandres/>

Tailwind CSS. (s. f.). A Utility-First CSS Framework for Rapid UI Development. Recuperado el 1 de mayo de 2023, de <https://tailwindcss.com/>

Flowbite. (s. f.). Flowbite - Open Source Tailwind CSS UI Kit and Admin Dashboard Template. Recuperado el 25 de abril de 2023, de <https://flowbite.com/>

Tailwind Elements. (s. f.). Tailwind Elements - High-Quality UI Components Built with Tailwind CSS. Recuperado el 25 de abril de 2023, de <https://tailwind-elements.com/>

Tailwind Shades. (s. f.). Tailwind Shades - Beautifully Designed Tailwind CSS Templates. Recuperado el 12 de abril de 2023, de <https://www.tailwindshades.com/>

Pexels. (s. f.). Free stock photos · Pexels. Recuperado el 16 de abril de 2023, de <https://www.pexels.com/>

Canva. (s. f.). Amazingly Simple Graphic Design Software – Canva. Recuperado el 3 de abril de 2023, de <https://www.canva.com/>

Chart.js. (s. f.). Simple yet flexible JavaScript charting for designers & developers. Recuperado el 25 de abril de 2023, de <https://www.chartjs.org/>

TypeScript. (s. f.). TypeScript: Typed JavaScript at Any Scale. Recuperado el 9 de abril de 2023, de <https://www.typescriptlang.org/>

Bootstrap. (s. f.). Build responsive, mobile-first projects on the web with the world's most popular front-end component library. Recuperado el 13 de abril de 2023, de <https://getbootstrap.com/>

midudev. (s. f.). midudev - YouTube. Recuperado el 8 de abril de 2023, de <https://www.youtube.com/c/midudev>