

## PROYECTO TECNOLÓGICO EVALUACIÓN FINAL

### INSTRUCCIONES GENERALES:

- Firmar (registrar sus datos) el compromiso ético, caso contrario no será evaluado.
- La evaluación es **grupal** y el formato de respuesta se encuentra descrito en la *Guía para la Elaboración del Proyecto*.
- Antes de responder, revisa la Rúbrica de evaluación.

### COMPROMISO ÉTICO:

*Los firmantes, afirmamos ser los autores del contenido de esta evaluación, y aseguramos no haber tomado parcial o totalmente ningún texto académico de alumnos de esta u otra institución ni de otros documentos sin haber colocado la cita correspondiente.*

*Sabemos que esta actividad podrá ser analizada con software antiplagio y con software detector de inteligencia artificial generativa. Somos conscientes de que se aplicará el reglamento vigente de estudios y las sanciones que correspondan de encontrarse irregularidades en cuanto al contenido entregado.*

### INTEGRANTES DE GRUPO:

NOMBRES Y APELLIDOS	COD. ALUMNO (DNI)
1. Jasmin Xiomari Villaca Mamani	72921405
2. Gonzalo Tafur Bermúdez	77686373
3.	
4.	
5.	

### INSTRUCCIONES ESPECÍFICAS:

#### I. OBJETIVO DE LA EVALUACIÓN:

Presentación del Entregable #6.

#### II. CONTENIDO DEL ENTREGABLE:

- a) **Compromiso ético**, (*este documento*), se integra al informe -antes de la carátula-
- b) **Exposición**, (*en sesión de clase o en video pregrabado según lo indique el profesor, 25 min*)
- c) **El informe del proyecto**; (*comprende los ítems que se especifican en la Guía para la elaboración del proyecto, para este entregable*).

#### III. CRITERIOS DE EVALUACIÓN:

*[La rúbrica de calificación se anexa a este mismo documento]*



PROYECTO

**APLICACIÓN PARA DIETAS Y RUTINAS EN LINEA**

AUTORES: Tafur Bermúdez, Gonzalo

Villaca Mamani, Jasmin Xiomi

DOCENTE: Espinoza Rivera, Marco Aurelio

LINK: <https://youtu.be/i5M3SE7mib8>

LIMA – PERÚ

\_\_\_\_\_, de \_\_\_\_\_, de 2024

# ÍNDICE

<b>1. INTRODUCCIÓN</b>	5
1.2    Objetivo general	5
1.3.    Objetivos específicos	5
1.4.    Beneficios esperados	6
<b>2. PRODUCTO</b>	6
2.1.    Alcance	6
2.2.    Funcionalidades y características	7
2.3.    Usuarios	7
<b>3. METODOLOGÍA</b>	8
3.1 Roles y responsabilidades	9
3.2 Plataformas y herramientas	10
<b>4. PRODUCT BACKLOG</b>	10
4.1 Registro de stakeholder	10
4.2 Biblioteca de Historias de usuario	12
4.3 Definición de los Sprints:	23
<b>5. SPRINT 1</b>	25
5.1 Sprint Backlog:	25
5.2 Sprint Planning:	27
5.3 Daily Scrum:	32
5.4 Sprint Review:	36
5.5 Sprint Retrospective:	40
5.6 Product Backlog [actualizado]:	41
<b>6. SPRINT 2</b>	42
6.1 Sprint Backlog:	42
6.2 Sprint Planning:	47
6.3 Daily Scrum:	53
6.4 Sprint Review:	57
6.5 Sprint Retrospective:	63
6.6 Product Backlog [actualizado]:	65

<b>7. SPRINT 3</b>	66
<b>7.1 Sprint Backlog:</b>	66
<b>7.2 Sprint Planning:</b>	73
<b>7.3 Daily Scrum:</b>	78
<b>7.4 Sprint Review:</b>	87
<b>7.5 Sprint Retrospective:</b>	93
<b>7.6 Product Backlog [actualizado]:</b>	96
<b>8. SPRINT 4</b>	97
<b>8.1 Sprint Backlog:</b>	97
<b>8.2 Sprint Planning:</b>	106
<b>8.3 Daily Scrum:</b>	111
<b>8.4 Sprint Review:</b>	119
<b>8.5 Sprint Retrospective:</b>	126
<b>8.6 Product Backlog [actualizado]:</b>	129
<b>9. SPRINT 5: MVP</b>	129
<b>9.1 Sprint Backlog:</b>	129
<b>9.2 Sprint Planning:</b>	139
<b>9.3 Daily Scrum:</b>	141
<b>9.4 Sprint Review:</b>	144
<b>9.5 Sprint Retrospective:</b>	149
<b>10. DESPLIEGUE DEL PRODUCTO</b>	150
<b>11. LECCIONES APRENDIDAS</b>	152
<b>12. REFERENCIAS</b>	153

# 1. INTRODUCCIÓN

## 1.1 Realidad Problemática

A nivel global, existe una problemática de salud debido al sedentarismo y los malos hábitos de alimentación que tienen las personas, llevando consigo consecuencias como sobrepeso, enfermedades cardiovasculares, diabetes, anemia y hasta cáncer.

Según la Organización Mundial de la Salud (OMS), el 31% de los adultos y el 80% de los adolescentes a nivel global no cumplen con los niveles recomendados de actividad física. Esta falta de ejercicio físico es uno de los factores clave que contribuyen al aumento de enfermedades crónicas no transmisibles. A nivel nutricional, la desnutrición afecta a personas de todas las regiones del mundo, generando dos extremos preocupantes: 890 millones de adultos sufren de obesidad, mientras que 390 millones tienen insuficiencia ponderal.

En el Perú, solo el 26% de la población adulta realiza actividad física regularmente, según el Minsa. Esto contribuye a un creciente sedentarismo que aumenta el riesgo de diabetes, hipertensión y enfermedades cardiovasculares además de generar un costo económico significativo de 10,500 millones de dólares, equivalentes al 4.6% del Producto Bruto Interno (PBI) del país. El Minsa recomienda 150 minutos de actividad física semanal en adultos para prevenir estos problemas. Además, promueve una alimentación saludable basada en alimentos naturales y sugiere evitar productos ultra procesados, claves para mejorar la calidad de vida y extender la esperanza de vida, actualmente de 75 años en el país.

En este contexto, la aplicación web de dietas y rutinas se presenta como una herramienta clave para combatir el sedentarismo y mejorar los hábitos alimenticios. Al ofrecer recomendaciones personalizadas, busca ayudar a los usuarios a alcanzar sus objetivos de salud y prevenir enfermedades crónicas, mejorando así su calidad de vida.

## 1.2 Objetivo general

El objetivo es desarrollar una aplicación web para dietas y rutinas online utilizando Eclipso, y los lenguajes Java y SQL. Los nutriólogos podrán utilizar este aplicativo para publicar dietas y rutinas personalizadas para los pacientes, que tendrán acceso luego de registrarse. Los pacientes podrán registrarse y visualizar una lista de dietas y rutinas al acceder al menú, en los cuales pueden guardarlos, añadirlos a un calendario, y recibir notificaciones dentro de la página para mantenerse activos y al día con el calendario programado. Además, la aplicación permitirá programar reuniones virtuales entre nutriólogos y pacientes, brindando un espacio para realizar evaluaciones, resolver dudas y ofrecer un seguimiento personalizado.

## 1.3. Objetivos específicos

- **Desarrollar una interfaz intuitiva y amigable:** Crear un diseño que permita a los nutriólogos y pacientes interactuar fácilmente con la plataforma.
- **Implementar una base de datos robusta:** Almacenar y gestionar ejercicios, rutinas y dietas personalizadas, asegurando la integridad y accesibilidad de los datos.
- **Facilitar el registro de pacientes:** Incluir un sistema eficiente para el registro de pacientes, asegurando que tengan fácil acceso a sus rutinas y dietas personalizadas.
- **Incluir un sistema de notificaciones:** Desarrollar un sistema que mantenga a los pacientes informados sobre sus actividades programadas, recordatorios y sugerencias para mantener un estilo de vida activo.
- **Optimizar la experiencia de usuario (UX/UI):** Asegurar que la navegación sea fluida y atractiva, priorizando la satisfacción del usuario.
- **Implementar un algoritmo de recomendación:** Desarrollar un sistema que sugiera dietas y rutinas personalizadas según los objetivos del paciente (volumen o definición) al momento de su registro.

#### 1.4. Beneficios esperados

- Mayor accesibilidad para los nutriólogos al poder gestionar y recomendar rutinas y dietas personalizadas de manera remota y digital.
- Reducción de la inactividad y el sedentarismo al motivar a los pacientes a mantener un estilo de vida saludable mediante la aplicación.
- Aumento de la satisfacción del paciente al poder personalizar su plan de salud y recibir seguimiento en tiempo real.
- Fomento de la comunicación y el feedback entre los pacientes y los nutriólogos, mejorando la efectividad de las recomendaciones.
- Contribución a la lucha contra problemas de salud como el sobrepeso, la obesidad y las enfermedades relacionadas con malos hábitos alimenticios y falta de ejercicio.
- Personalización automática para los usuarios, quienes recibirán recomendaciones de dietas y rutinas adaptadas a sus objetivos desde su primer ingreso, sin necesidad de intervención manual del nutriólogo.
- Optimización del tiempo para los nutriólogos al permitir que el sistema genere sugerencias iniciales para los pacientes, enfocando su atención en ajustes más específicos o avanzados.
- Personalización automática para los usuarios, quienes recibirán recomendaciones de dietas y rutinas adaptadas a sus objetivos (volumen o déficit calórico) desde su primer ingreso.
- Mejora en los resultados de los usuarios al recibir planes alineados con sus metas específicas (volumen o déficit calórico)

## 2. PRODUCTO

### 2.1. Alcance

El alcance consiste en desarrollar una aplicación web para un nutriólogo, esta permitirá la gestión y distribución de rutinas y dietas a los usuarios registrados. Esta plataforma permitirá al nutriólogo cargar y categorizar rutinas y planes de alimentación basados en objetivos específicos, como volumen, recomposición corporal y déficit calórico.

Los usuarios podrán explorar y seleccionar rutinas y dietas, agregándolas a su repertorio personal. La página de inicio será el punto de acceso principal, donde los usuarios visualizarán las opciones organizadas para facilitar la selección. Además, cada usuario tendrá acceso a un calendario para programar sus rutinas.

La aplicación se desarrollará utilizando Eclipse, con una base de datos en SQL Server. Incluirá un sistema de notificaciones automáticas para recordar a los usuarios sobre su ingesta de alimentos y ejercicios. También se implementará un algoritmo de recomendación que ajustará automáticamente las dietas y rutinas según los datos proporcionados por el usuario.

## ATRIBUTOS TÉCNICOS:

- PLATAFORMA DE DESARROLLO: Eclipse
- LENGUAJE DE PROGRAMACIÓN BACKEND: Java + Spring Boot + Thymeleaf
- LENGUAJE DE PROGRAMACIÓN FRONTEND: html, CSS y Javascript
- BASE DE DATOS: SQL Server
- ENTORNO DE EJECUCIÓN: Aplicación web accesible a través de navegadores en dispositivos móviles y de escritorio.
- INTEGRACIÓN DE SOFTWARE: API para la gestión de usuarios y datos, y sistemas de notificación.

## 2.2. Funcionalidades y características

### INTERFAZ INTUITIVA:

- Diseño simple y fácil de usar.
- Facilita la navegación tanto para nutriólogos como para pacientes.

### CONFIGURACIÓN PERSONALIZADA:

- Los nutriólogos pueden personalizar rutinas y dietas según las necesidades específicas de cada paciente.
- Permite adaptar ejercicios y planes alimenticios basados en objetivos individuales.

### RECOMENDACIONES AUTOMÁTICAS:

- Sistema con algoritmo que sugiere dietas y rutinas en función de los objetivos del paciente, como volumen o déficit calórico.
- Personaliza las sugerencias desde el momento del registro.

### NOTIFICACIONES AUTOMÁTICAS:

- Envío de recordatorios y alertas para mantener a los pacientes al tanto de sus rutinas y dietas.
- Mantiene a los usuarios activos y comprometidos con su progreso.

### REAJUSTE DE RECOMENDACIONES:

- Si los datos del paciente, como peso o medidas, se alteran y no concuerdan con su objetivo (por ejemplo, perder peso cuando se busca volumen), el sistema ajusta las recomendaciones automáticamente.
- Los nuevos ajustes se muestran en la página de inicio o debajo del calendario.

## 2.3. Usuarios

### NUTRIÓLOGOS:

- Usuarios encargados de crear y personalizar rutinas y dietas, revisar el progreso de los pacientes y recibir feedback.

#### PACIENTES:

- Usuarios que reciben dietas y rutinas personalizadas, pueden añadirlas a su calendario, recibir recomendaciones y dar feedback.

## 3. METODOLOGÍA

### SCRUM

Para el desarrollo de la aplicación web de dietas y rutinas, se ha elegido la metodología ágil SCRUM. SCRUM es la metodología adecuada para el desarrollo de nuestro proyecto debido a:

- **FLEXIBILIDAD Y ADAPTABILIDAD:**  
SCRUM nos permite adaptarnos rápidamente a cambios y ajustes en los requerimientos. A medida que avanzamos, podemos incorporar feedback y hacer mejoras continuas.
- **TRABAJO COLABORATIVO**  
La metodología fomenta una comunicación constante entre el equipo. Las reuniones entre una a dos veces por semana nos mantienen alineados, asegurando que todos estén en la misma página y que se resuelvan problemas de manera eficiente.
- **ENTREGAS INCREMENTALES**  
A través de sprints, que son entre intervalos de 2 a 3 semanas, podemos entregar funcionalidades de manera regular.
- **PLANIFICACIÓN EFECTIVA**  
Al definir historias de usuario, podemos priorizar tareas y asegurarnos de que estamos trabajando en lo más importante en cada etapa del proyecto. Esto es crucial para cumplir con nuestro plazo de 16 semanas.

### CICLO DE VIDA

El proyecto tiene una duración de 16 semanas, y el intervalo entre los entregables de cada sprint es de 2 a 3 semanas.

Hito	Semana	Evaluación
<i>Kick Off</i>	2	
<i>Avance de Entregable 1</i>	3	<i>Avance EP1</i>
<b>Entregable 1</b>	<b>4</b>	<b>EP 1</b>
<i>Avance de Entregable 2</i>	5	<i>Avance EP2</i>
<b>Entregable 2</b>	<b>6</b>	<b>EP2</b>
<i>Avance de Entregable 3</i>	7	<i>Avance Parcial</i>
<b>Entregable 3</b>	<b>8</b>	<b>PARCIAL</b>
<i>Avance de Entregable 4</i>	9	<i>Avance EP4</i>
<i>Avance de Entregable 4</i>	10	<i>Avance EP4</i>
<b>Entregable 4</b>	<b>11</b>	<b>EP3</b>
<i>Avance de Entregable 5</i>	12	<i>Avance EP5</i>
<i>Avance de Entregable 5</i>	13	<i>Avance EP5</i>
<b>Entregable 5</b>	<b>14</b>	<b>EP 4</b>
<i>Avance de Entregable 6</i>	15	<i>Avance Final</i>

## 3.1 Roles y responsabilidades

### PRODUCT OWNER

Tafur Bermúdez, Gonzalo

- Diseño y programación de la aplicación para asegurarse de que sea atractiva y fácil de usar para los visitantes.
- Realizar revisiones periódicas del producto en desarrollo para garantizar que cumpla con las expectativas y los estándares de calidad establecidos.
- Trabajar en estrecha colaboración con el equipo de desarrollo, para asegurar una comunicación clara y efectiva en todas las etapas del proyecto
- Liderar las reuniones pactadas y definir alcances.

### SCRUM MASTER

Villacá Mamani, Jasmin Xiomi

- Identificar oportunidades para mejorar la aplicación de escritorio y proponer soluciones para mejorar su eficacia y alcance.
- Realizar cualquier otra tarea necesaria para garantizar el éxito del proyecto.
- Comunicarse con el equipo de proyecto y otras partes interesadas para mantenerlos informados sobre el progreso del proyecto.
- Asegurar que el equipo siga las prácticas ágiles y promueva la mejora continua.

### DEVELOPMENT TEAM

Villacá Mamani, Jasmin Xiomi

- Proveer información sobre dietas y rutinas.
- Investigar y recopilar datos sobre ejercicios.
- Asistir en la elaboración de contenido para la aplicación.
- Colaborar en la estructuración de las tablas de datos.
- Revisar y validar la información ingresada en el sistema.

Tafur Bermúdez, Gonzalo

- Desarrollar, escribir y optimizar el código para la aplicación web.
- Realizar pruebas continuas para asegurar la calidad del producto, identificando errores y fallos en el sistema.
- Implementar funciones y características según los requisitos establecidos en las historias de usuario.
- Implementar la base de datos y su estructura.
- Validar la funcionalidad del reajuste de recomendaciones y las notificaciones por alteración continua de datos.

### 3.2 Plataformas y herramientas

#### DESARROLLO

- Eclipse
- Java + Spring Boot + Thymeleaf (BACKEND)
- html, CSS y Javascript (FRONTEND)
- SQL Server

#### REUNIONES

- Google Meet

#### ORGANIZACIÓN

- JIRA Software

## 4. PRODUCT BACKLOG

### 4.1 Registro de stakeholder

#### RECOLECCIÓN DE REQUERIMIENTOS

Se realizó una reunión con los usuarios y el Product Owner para identificar necesidades y expectativas del sistema.

Stakeholder	Requerimientos
Nutriólogos	Crear dietas y rutinas personalizadas.
	Gestionar los perfiles de sus pacientes.
	Recibir notificaciones cuando los datos de los pacientes no coincidan con los objetivos establecidos.
	Función de reajuste automático de recomendaciones basadas en el progreso del paciente.

Stakeholder	Requerimientos
	Registrar su progreso (peso, medidas, etc.).

<b>Pacientes/Usuarios</b>	Acceder a dietas y rutinas recomendadas automáticamente por el sistema.
	Recibir notificaciones de recordatorio de dietas o rutinas.
	Ver recomendaciones ajustadas cuando sus datos no coinciden con sus objetivos de salud.

## REFINAMIENTO DE REQUERIMIENTOS

Los requerimientos recopilados fueron analizados, transformándose en requisitos claros que guiarán el desarrollo de historias de usuario.

Stakeholder	Requisitos
<b>Nutriólogos</b>	Configurar planes de alimentación y rutinas físicas personalizadas para cada paciente.
	Administrar los perfiles de los pacientes, incluyendo su historial y progreso.
	Recibir alertas cuando los datos ingresados por el paciente no concuerden con sus objetivos.
	Modificar automáticamente las recomendaciones de dieta y ejercicios según el avance del paciente.

Stakeholder	Requisitos
<b>Pacientes/Usuarios</b>	Ingresar y actualizar su progreso físico (peso, medidas corporales).
	Consultar dietas y rutinas generadas automáticamente por el sistema de acuerdo a sus metas.
	Recibir alertas para recordar rutinas y dietas pendientes.

	Acceder a ajustes automáticos en sus recomendaciones cuando su progreso no se alinee con sus objetivos.
--	---

## 4.2 Biblioteca de Historias de usuario

### HU01 – Registrar usuario

Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.

**Prioridad:** 1000

**Estimación:** 3 días

**Dependiente de:** N/A

**Criterios de aceptación:**

Escenario 1

La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.

El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.

### HU02 – Cuestionario para pacientes nuevos

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

**Prioridad:** 900

**Estimación:** 6 días

**Dependiente de:** HU01

**Criterios de aceptación:**

Escenario 1 – Cuestionario

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

## HU03 – Acceso para usuarios

Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario

**Prioridad:** 1000

**Estimación:** 3 días

**Dependiente de:** HU01

**Criterios de aceptación:**

Escenario 1

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

## HU04 – Menú de Inicio

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

**Prioridad:** 1000

**Estimación:** 5 días

**Dependiente de:** HU02, HU03

**Criterios de aceptación:**

Escenario 1 - Menú de inicio

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

## HU05 - Insertar rutinas personalizadas

Como nutriólogo, quiero crear y personalizar rutinas de ejercicios agregando descripciones específicas para recomendarlas a mis pacientes registrados.

**Prioridad:** 1000

**Estimación:** 2 días

**Dependiente de:** HU01, HU04

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear rutinas, se le redirige a una ventana donde puede buscar ejercicios en la base de datos, seleccionarlos y organizarlos en una rutina personalizada. Además, el nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar la rutina según los objetivos específicos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la rutina y recomendarla a sus pacientes registrados.

## HU06 – Insertar dietas personalizadas

Como nutriólogo, quiero crear y personalizar dietas agregando detalles específicos para recomendarlas a mis pacientes registrados.

**Prioridad:** 1000

**Estimación:** 5 días

**Dependiente de:** HU01, HU04

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear dietas, se le redirige a una ventana donde puede ingresar y organizar planes de alimentación personalizados. El nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar las dietas según objetivos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la dieta y recomendarla a sus pacientes registrados.

## HU07 - Editar rutinas y dietas

Como nutriólogo, quiero poder editar los detalles de las dietas y rutinas para asegurarme de que la información esté actualizada y sea precisa.

**Prioridad:** 700

**Estimación:** 4 días

**Dependiente de:** HU05, HU06

**Criterios de aceptación:**

Escenario 1 - Acción editar rutinas

Cuando el nutriólogo selecciona la opción para editar una rutina que él creó, podrá modificar sus datos, incluyendo el tipo de rutina, la categoría, su descripción y los ejercicios incluidos. Estas modificaciones permitirán mantener la base de datos actualizada.

Escenario 2 - Acción editar dietas

Cuando el nutriólogo selecciona la opción para editar una dieta que él creó, podrá modificar sus datos, incluyendo el tipo de dieta, la descripción, y los alimentos incluidos en esta dieta. Estas modificaciones permitirán mantener la base de datos actualizada.

## HU08 – Ver detalle de una rutina y dieta

Como paciente, quiero visualizar a detalle la dieta y rutina dentro de una lista para tenerla guardarla sea en calendario o por texto.

**Prioridad:** 700

**Estimación:** 4 días

**Dependiente de:** HU05, HU06

### Criterios de aceptación:

Escenario 1 – Ver detalle de una rutina

Cuando el usuario selecciona la opción para ver el detalle de una rutina, podrá visualizar sus datos, incluyendo el tipo de rutina, el nutriólogo quien lo creó, la categoría, su descripción y una lista de los ejercicios incluidos. También unos botones para descargarlo o guardarlo a tu calendario.

Escenario 2 - Ver detalle de una dieta

Cuando el usuario selecciona la opción para ver el detalle de una dieta, podrá visualizar sus datos, incluyendo el tipo de dieta, el nutriólogo quien lo creó, su descripción y una lista de alimentos que debería consumir. También unos botones para descargarlo o guardarlo a tu calendario.

## HU09 – Ver lista de alimentos

Como nutriólogo, quiero ver una lista de alimentos disponibles en la base de datos para poder consultarlos y utilizarlos en la creación de dietas personalizadas para mis pacientes.

**Prioridad:** 650

**Estimación:** 1 día

**Dependiente de:** HU05, HU06, HU08

### Criterios de aceptación:

## Escenario 1

Cuando el nutriólogo accede a la sección de alimentos, se le muestra una lista de alimentos que incluye el nombre, tipo (desayuno, almuerzo, cena, merienda), y sus nutrientes principales (proteínas, grasas, carbohidratos). El nutriólogo puede buscar alimentos específicos, filtrarlos por tipo o por nutrientes, y seleccionar aquellos que desea incluir en las dietas de sus pacientes.

### HU10 – Perfil de nutriólogos

Como nutriólogo, quiero tener acceso a un perfil de usuario para insertar información visible y crear mis rutinas y dietas para que más pacientes puedan visitarme y ponerse en contacto conmigo.

**Prioridad:** 900

**Estimación:** 4 días

**Dependiente de:** HU01, HU04

#### Criterios de aceptación:

Escenario 1 – Perfil de nutriólogo

Cuando el nutriólogo entre a su perfil, podrá visualizar qué rutinas y dietas ha creado junto a un botón para crear otras, y una opción de editar perfil para añadir sus contactos, lugar en donde trabaja y descripción.

### HU11 – Perfil de pacientes

Como paciente, quiero tener acceso un perfil de usuario para visualizar mi actividad y gestionar mi información visible.

**Prioridad:** 900

**Estimación:** 4 días

**Dependiente de:** HU01, HU02, HU04

**Criterios de aceptación:**

Escenario 1 – Perfil de paciente

Cuando el paciente ingrese su perfil, podrá visualizar que rutina o dieta ha tiene guardado, junto a una opción de editar perfil para añadir su descripción y gestionar la información solicitada en el cuestionario.

## HU12 – Ver lista de ejercicios

Como nutriólogo, quiero ver una lista de ejercicios en la base de datos para poder consultarlos y utilizarlos en la creación de rutinas personalizadas para mis pacientes.

**Prioridad:** 650

**Estimación:** 1 día

**Dependiente de:** HU05, HU06, HU08

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de ejercicios, se le muestra una lista que incluye el nombre del ejercicio, el grupo muscular al que pertenece, el tipo de ejercicio y una breve descripción. El nutriólogo puede buscar ejercicios específicos, filtrarlos por grupo muscular o tipo de ejercicio, y seleccionar aquellos que desea incluir en las rutinas de sus pacientes.

## HU13 – Ajustes de Usuario

Como usuario, quiero hacerles ajustes de seguridad a mí cuenta para no descuidarla en caso de que se me olvide la contraseña.

**Prioridad:** 900

**Estimación:** 4 días

**Dependiente de:** HU01, HU04

**Criterios de aceptación:**

Escenario 1 – Ajustes de usuario

Cuando el usuario está en la opción de ajustes, tendrá una opción para actualizar contraseña, y también una para cerrar sesión.

## HU14 - Personalizar calendario de paciente

Como paciente, quiero gestionar un calendario personal para organizar mis rutinas de entrenamiento proporcionadas por el nutriólogo.

**Prioridad:** 900

**Estimación:** 7 días

**Dependiente de:** HU01, HU04, HU05, HU06

**Criterios de aceptación:**

Escenario 1

El paciente llega a una opción donde puede acceder a su calendario. El paciente tendrá acceso a un calendario donde podrá agregar sus rutinas de ejercicios para cada día de la semana. Podrá asignar los días de entrenamiento, seleccionar qué rutinas realizará cada día, y programar los días de descanso. Esto le permitirá visualizar y organizar su plan de entrenamiento semanal.

## HU15 – Feedback sobre rutinas y dietas

Como paciente, quiero poder calificar y dejar comentarios sobre las rutinas y dietas que he seguido para proporcionar feedback útil al nutriólogo y a otros pacientes.

**Prioridad:** 700

**Estimación:** 5 días

**Dependiente de:** HU01, HU08

**Criterios de aceptación:**

Escenario 1 - Feedback sobre rutinas y dietas

Después de completar una rutina o dieta, el paciente podrá calificarla usando un sistema de estrellas o puntos y dejar un comentario opcional describiendo su experiencia. Esta información será visible tanto para el nutriólogo como para otros pacientes registrados, ayudando a mejorar las recomendaciones y la personalización de futuras rutinas y dietas.

## HU16 – Notificaciones para usuario

Como paciente, quiero recibir notificaciones del sistema sobre recomendaciones y actividades programadas en mi calendario para estar al tanto y mantenerme al día con mi programa de salud.

**Prioridad:** 900

**Estimación:** 8 días

**Dependiente de:** HU04, HU13

**Criterios de aceptación:**

Escenario 1 – Notificaciones

Cuando el usuario ingresa a su e-mail con el que se creó su cuenta o a las notificaciones de la página, verá nuevas notificaciones sobre actividades que ha programado o sugerencias sobre dietas o rutinas que le pueden interesar.

### HU17 – Registro semanal de medidas y seguimiento de progreso

Como paciente, quiero ingresar mis medidas semanales (peso, perímetro de cintura y bíceps), para monitorear mi progreso en función de mis objetivos deportivos.

**Prioridad:** 750

**Estimación:** 6 días

**Dependiente de:** HU02, HU11

**Criterios de aceptación:**

Escenario 1

El paciente podrá ingresar sus medidas cada semana para monitorear su progreso. Estos datos se guardarán en su perfil para visualizar el cambio a lo largo del tiempo.

Escenario 2

En caso el paciente no ingrese sus datos dentro de la semana el sistema alertará al paciente enviando una notificación a su correo.

### HU18 – Recomendaciones automáticas basadas en datos alterados

Como paciente, quiero que el sistema me recomiende nuevas dietas o rutinas en caso de que mis datos (peso, medidas, etc.) no estén alineados con mis objetivos de salud, para mejorar mi progreso.

**Prioridad:** 800

**Estimación:** 8 días

**Dependiente de:** HU16

**Criterios de aceptación:**

Escenario 1 – Recomendaciones automáticas

Si los datos del paciente no coinciden con sus objetivos, el sistema generará nuevas recomendaciones de dietas o rutinas en su página de inicio.

## Escenario 2 – Notificación por datos persistentes

Si tras dos semanas los datos siguen desalineados, se enviará una notificación al paciente y al nutriólogo para programar una reunión y revisar el plan de salud.

### HU19 – Recomendación Inicial de Dietas y Rutinas Personalizadas

Como paciente recién registrado, quiero que el sistema me recomiende dietas y rutinas adecuadas a mis objetivos (déficit calórico o volumen) y experiencia física previa, para poder iniciar mi programa de forma segura y efectiva, considerando mis capacidades y posibles restricciones alimenticias.

**Prioridad:** 800

**Estimación:** 10 días

**Dependiente de:** HU04

**Criterios de aceptación:**

Escenario 1

El sistema recomendará dietas y rutinas basadas en el cuestionario realizado después del registro.

## 4.3 Definición de los Sprints:

### PRODUCT BACKLOG

#### PROCEDIMIENTO DE ESTIMACIÓN DE ESFUERZO Y COMPROMISO DEL EQUIPO

##### a) Revisión de las Historias de Usuario (HU):

- ✓ Identificamos todas las historias de usuario necesarias para cumplir con los objetivos del proyecto. Estos incluyen el registro de usuarios (HU01), el acceso a la plataforma (HU03), la inserción de rutinas y dietas (HU05, HU06), entre otras.

##### b) Estimación del esfuerzo:

- ✓ Utilizamos una técnica de estimación basada en días para determinar el tiempo que completará cada HU. Esta estimación se basa en la complejidad técnica, las dependencias entre historias, y la cantidad de contenido que necesita ser gestionado (dietas, rutinas, ejercicios, etc.).
- ✓ Para historias más simples como HU01 (Registrar usuario) y HU03 (Acceso), se asignan menos días (2-4 días), mientras que historias más complejas como HU05 (Insertar rutinas personalizadas) y HU18 (Recomendaciones automáticas basadas en datos alterados) requieren más tiempo (5-10 días).

**c) Procedimiento de Estimación de Esfuerzo y Compromiso del Equipo:**

- ✓ EQUIPO DE CONTENIDOS: Encargado de organizar y clasificar la información de rutinas, dietas y ejercicios que proporciona el nutriólogo, para que esté lista cuando el programador lo requiera. Esto incluye gestionar la base de datos y tener los datos ordenados en Excel.
- ✓ EQUIPO DE DESARROLLO: Dado que solo tenemos un programador en el equipo, él se encarga de implementar todas las funcionalidades técnicas, como la creación de formularios, la lógica de recomendación y la conexión con la base de datos. Esta carga técnica se ha distribuido cuidadosamente a lo largo de los sprints.

**d) Procedimiento de Estimación de Esfuerzo y Compromiso del Equipo:**

La capacidad del equipo se ha tenido en cuenta al organizar los sprints. Cada sprint incluye solo el número de historias que el equipo puede manejar dentro del tiempo disponible. Por ejemplo:

- ✓ SPRINT 1: Se completan HU01 (Registro de usuarios) y HU03 (Acceso de usuarios) porque son fundamentales para que el resto de las funcionalidades estén disponibles.
- ✓ SPRINT 2: Se trabaja en HU05 (Rutinas personalizadas) y HU06 (Diетas personalizadas), ya que el sistema necesita estar operativo con usuarios registrados.
- ✓ SPRINT 3 EN ADELANTE: Se incluyen funcionalidades más avanzadas como HU18 (Recomendaciones automáticas) y HU16 (Notificaciones), que requieren tener una base sólida de usuarios y contenido previamente cargado.

**e) Procedimiento de Estimación de Esfuerzo y Compromiso del Equipo:**

- ✓ ORGANIZADOR: Organizar y proporcionar la información de manera adecuada para que el programador pueda integrarla en la aplicación sin retrasos.
- ✓ PROGRAMADOR: Asumir la mayor parte del trabajo técnico, con una distribución que le permita trabajar de forma sostenible durante cada sprint.

ID	HISTORIA DE USUARIO	PRIORIDAD	ESTIMACION	DEPENDENCIA	SPRINT
<b>HU01</b>	Registrar usuario	1000	4 días	--	1
<b>HU03</b>	Acceso para usuarios	988	2 días	HU01	1
<b>HU04</b>	Menú de Inicio	980	5 días	HU01, HU03	1
<b>HU05</b>	Insertar rutinas personalizadas	975	5 días	HU01, HU03, HU06	2
<b>HU06</b>	Insertar dietas personalizadas	971	5 días	HU01, HU03	2
<b>HU02</b>	Cuestionario para pacientes nuevos	970	6 días	HU01	1
<b>HU08</b>	Ver detalles de rutinas y dietas	965	1 dia	HU08, HU09	2
<b>HU10</b>	Perfil de nutriólogos	963	4 días	HU02	3
<b>HU11</b>	Perfil de pacientes	960	1 dia	HU	3
<b>HU07</b>	Editar rutinas y dietas	955	5 días	HU01, HU03	2
<b>HU12</b>	Ver lista de ejercicios	945	4 días	HU01, HU02, HU06	2
<b>HU09</b>	Ver lista de alimentos	935	1 dia		2
<b>HU14</b>	Personalizar calendario de paciente	920	7 días	HU05	3
<b>HU15</b>	Feedback sobre rutinas y dietas	863	5 días	HU05, HU08, HU09	3
<b>HU16</b>	Notificaciones para usuario	860	8 días	HU12, HU13	4
<b>HU17</b>	Registro semanal de medidas y seguimiento de progreso	830	6 días	HU04	4
<b>HU19</b>	Recomendación Inicial de Dietas y Rutinas Personalizadas	820	10 días	HU04	4
<b>HU18</b>	Recomendaciones automáticas basadas en datos alterados	800	8 días	HU16	4
<b>HU13</b>	Ajustes de Usuario	780	1 día	HU02	4

## 5. SPRINT 1

## 5.1 Sprint Backlog:

### INCREMENTO DE FUNCIONALIDADES INICIALES PARA PACIENTES Y NUTRÍÓLOGOS

- **Fecha de Inicio del Sprint:** 24 de septiembre del 2024.
- **Fecha de Fin del Sprint:** 8 de octubre del 2024.
- **Duración del Sprint:** 2 semanas

#### HISTORIAS DE USUARIOS

Este Sprint corresponde del 24/09/24 al 08/10/24, en el que se desarrollaron las historias de usuarios del sprint respectivas HU01, HU02, HU03 y HU04

HU01 – Registrar usuario	
Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.	
<b>Criterios de aceptación:</b>	
Escenario 1  La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.  El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.	

HU03 – Acceso para usuarios	
Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario	
<b>Criterios de aceptación:</b>	
Escenario 1	

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

## HU02 – Cuestionario para pacientes nuevos

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

### Criterios de aceptación:

#### Escenario 1 – Cuestionario

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

## HU04 – Menú de Inicio

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

### Criterios de aceptación:

#### Escenario 1 - Menú de inicio

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

## 5.2 Sprint Planning:

### AGENDA TRATADA DURANTE LA CEREMONIA DE SPRINT PLANNING:

**a) Revisión de las Historias de Usuario del Sprint 1:**

- Se discutió la prioridad de las HU a trabajar.
- Se detallaron las tareas para cada HU, asegurando que cada tarea requiera más de 20 minutos.
- Se aclararon los criterios de aceptación para cada historia.

**b) Asignación de responsabilidades:**

- Gonzalo Tafur Bermúdez fue asignado como responsable de todas las tareas técnicas.
- Se hizo una distribución de tareas que respetara la capacidad de Gonzalo para completar el trabajo dentro del Sprint.

**c) Determinación de dependencias y predecesoras:**

- Las tareas de HU03 y HU02 dependen de la HU01 (registro de usuarios).
- La HU04 (Menú de Inicio) tiene como predecesora la HU03 (Acceso para usuarios).

**d) Aprobación del Sprint Backlog:**

- Se llegó a un acuerdo sobre la lista de tareas a realizar durante el sprint, así como la duración y el responsable de cada tarea.

**PROCEDIMIENTO SEGUIDO PARA LA CEREMONIA:**

- FECHA: La ceremonia de planificación del Sprint 1 se llevó a cabo el 01/10/2024.
- PARTICIPANTES: Incluyó a todo el equipo (Scrum Master, Product Owner y Gonzalo como programador).
- RESULTADOS DE LA CEREMONIA:

✓ **Sprint Backlog:** Se seleccionan las HU y tareas del Sprint Backlog para el Sprint 1.

✓ **Duración de las tareas:** Cada tarea fue estimada y calendarizada para respetar la capacidad de Gonzalo.

✓ **Predecesoras:** HU02 y HU03 no pueden comenzar hasta que la HU01 se haya completado.

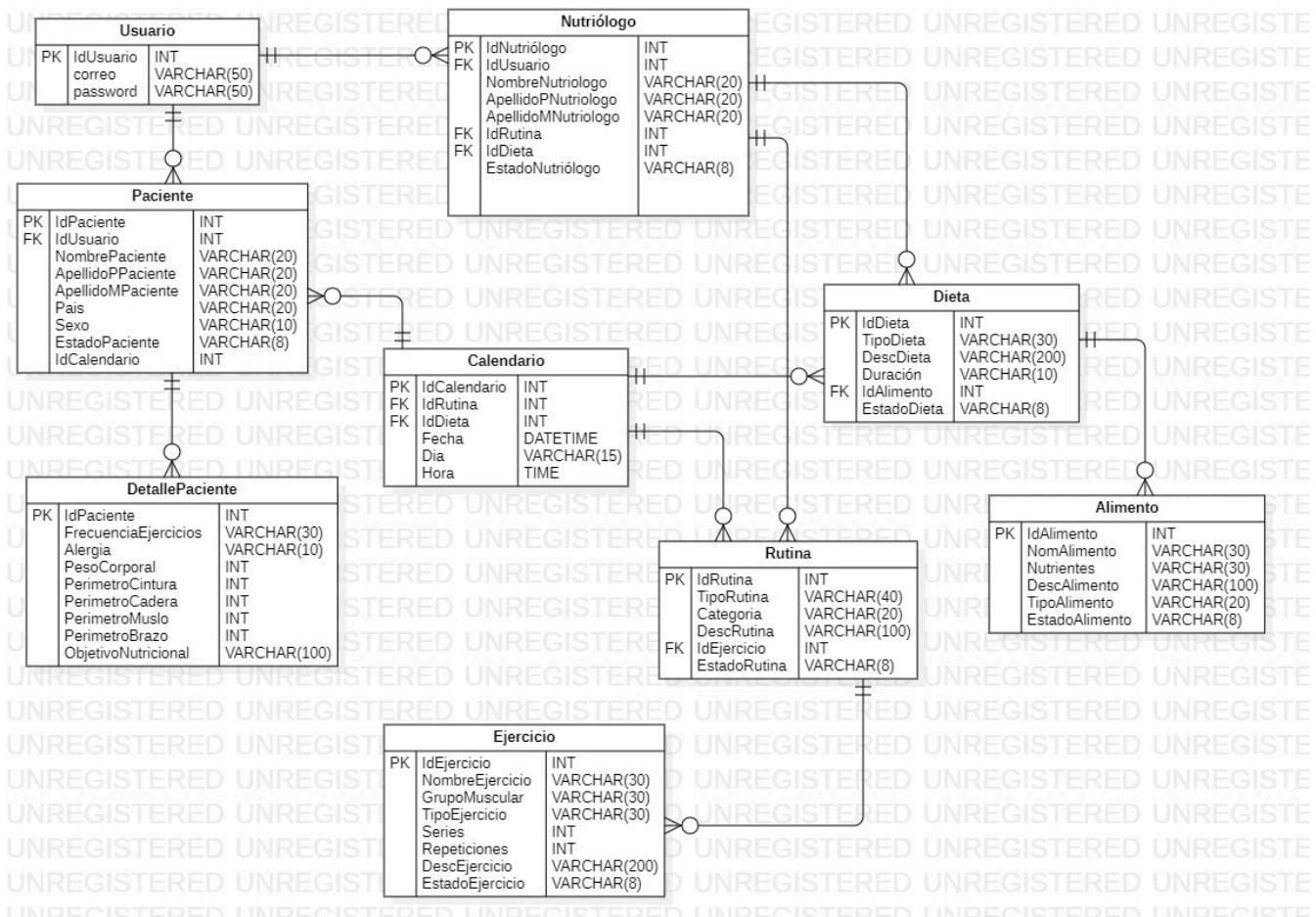
✓ **Evidencia:** Se presentaron prototipos de la interfaz de usuario y las bases de datos preliminares.

## PROTOTIPO INTERFAZ DE USUARIO

## Rutinas

Rutinas	Dietas	
Nombre de la Rutina	Descripción	Acciones
Rutina de Volumen	Intermedio	Rutina enfocada para ganar masa muscular

## PROTOTIPO DE LA BASE DE DATOS



Tareas/Tareas no técnicas	Estimación	Asignación de Trabajo
<b>HU01 – Registrar usuario</b>		
Realizar la Interfaz de Registro	2 días	Tafur Bermúdez, Gonzalo
Implementar el formulario con campos: nombre de usuario, contraseña, nombre, apellidos, correo, sexo y documento de identidad.	1 hora	Tafur Bermúdez, Gonzalo
Configurar la redirección según la selección de paciente o nutriólogo.	2 días	Tafur Bermúdez, Gonzalo
Conectar el formulario con el servidor para almacenar la información del paciente.	1 día	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad del formulario.	2 días	Tafur Bermúdez, Gonzalo
<b>HU02 – Cuestionario para pacientes nuevos</b>		
Diseñar el prototipo visual del cuestionario que se mostrará a los nuevos pacientes.	3 horas	Tafur Bermúdez, Gonzalo
Implementar las preguntas en el sistema: peso, última dieta, frecuencia de ejercicio, historial de hospitalizaciones, alergias alimentarias, etc.	2 días	Tafur Bermúdez, Gonzalo
Configurar la lógica para que el cuestionario se muestre automáticamente después del registro.	2 días	Tafur Bermúdez, Gonzalo
Implementar validaciones para asegurar que las respuestas se completen correctamente.	1 día	Tafur Bermúdez, Gonzalo
Conectar el cuestionario con el servidor para almacenar las respuestas de los pacientes.	4 días	Tafur Bermúdez, Gonzalo

Realizar pruebas de usabilidad y funcionalidad del cuestionario.	4 días	Tafur Bermúdez, Gonzalo
--	--------	-------------------------

<b>HU03 – Acceso para usuarios</b>		
Realizar la Interfaz de inicio de sesión.	3 horas	Tafur Bermúdez, Gonzalo
Implementar el formulario de validación de usuario con los campos correo electrónico y contraseña	3 horas	Tafur Bermúdez, Gonzalo
Implementar condiciones de vista según el tipo de usuario que inició sesión.	2 horas	Tafur Bermúdez, Gonzalo
Conectar el formulario con el servidor para validar la información del usuario.	3 horas	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad del formulario.	1 hora	Tafur Bermúdez, Gonzalo

<b>HU04 – Menú de Inicio</b>		
Diseñar el prototipo de la interfaz del menú de inicio, incluyendo la barra de navegación superior.	3 días	Tafur Bermúdez, Gonzalo
Implementar el ícono de usuario y el sistema de notificaciones en la barra de navegación.		Tafur Bermúdez, Gonzalo
Crear el banner de bienvenida que se mostrará al usuario en el menú de inicio.		Tafur Bermúdez, Gonzalo
Desarrollar las opciones de acceso a rutinas, dietas y calendario.	3 horas	Tafur Bermúdez, Gonzalo

Asegurar que todas las opciones sean accesibles y funcionales al hacer clic.	3 horas	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad para garantizar una navegación fluida en el menú de inicio.	4 horas	Tafur Bermúdez, Gonzalo

Este primer sprint se centra en el desarrollo de la interfaz de usuario para nutriólogos y pacientes en la aplicación web de rutinas y dietas. El objetivo principal es permitir que los nutriólogos gestionen rutinas y dietas personalizadas a través de una ventana intuitiva, donde puedan visualizar y agregar nuevas opciones. Paralelamente, se implementará un formulario de registro para pacientes que les permita acceder a la aplicación de forma sencilla y responder un cuestionario inicial que brinde información valiosa sobre su salud. También se desarrollará un menú de inicio que facilite la navegación entre rutinas, dietas y otras funcionalidades importantes.

Durante el Sprint Planning, el equipo completo, incluyendo al Scrum Master, Product Owner y stakeholders, se reunió para revisar el progreso del Sprint #1. Se discutieron los avances en la programación de la interfaz de usuario, enfocándose en la funcionalidad para nutriólogos y pacientes. Además, se presentaron los prototipos del formulario de registro y el menú de inicio, así como la estructura inicial de la base de datos que soportará la gestión de rutinas y dietas. Se abordaron las tareas completadas, los criterios de aceptación cumplidos

## KANBAN BOARD JIRA

The Kanban board displays the following tasks across three columns:

- POR HACER 14** (Top row):
  - Gestión de Usuarios (KAN-6)
  - Crear ejercicios (KAN-7)
  - Editar ejercicios (KAN-8)
  - Insertar rutinas personalizadas (KAN-9)
  - Insertar dietas personalizadas
- EN DESARROLLO** (Middle row):
  - Registrar usuario (KAN-3)
  - Cuestionario para pacientes nuevos (KAN-4)
  - Menú de Inicio (KAN-5)
- LISTO 3** (Bottom row):
  - Registrar usuario (KAN-3) - status: GT
  - Cuestionario para pacientes nuevos (KAN-4) - status: GT
  - Menú de Inicio (KAN-5) - status: GT

### 5.3 Daily Scrum:

Durante el Sprint 1, el equipo de desarrollo realizó reuniones semanales todos los domingos, con una duración de 2 a 3 horas, para sincronizar al equipo y abordar los obstáculos surgidos. En estas reuniones, se utilizaron preguntas específicas para evaluar el progreso y definir los siguientes pasos. A pesar de ser solo dos reuniones durante este sprint, se logró mantener una comunicación y se discutieron los avances en las tareas relacionadas con las historias de usuario, garantizando que el equipo estuviera alineado con los objetivos del proyecto.

## ¿QUÉ HICE AYER?

Ayer, el equipo finalizó el desarrollo del formulario de registro para pacientes. Se implementarán los campos necesarios, incluyendo nombre de usuario, contraseña y datos personales. También se avanzó en la configuración del cuestionario que los nuevos pacientes deberán completar tras el registro.

## ¿QUÉ HARÉ HOY?

Hoy, el equipo se enfocará en completar el cuestionario para pacientes nuevos, asegurando que las preguntas aborden aspectos como peso, historial de dietas y frecuencia de ejercicio. Además, comenzaremos a trabajar en el diseño del menú de inicio, que incluirá una barra de navegación y acceso a las funcionalidades clave.

## ¿QUÉ IMPEDIMENTOS TENGO?

Se ha presentado un impedimento relacionado con la integración del cuestionario a la base de datos. Debido a la falta de participantes, el desarrollador enfrenta dificultades al no tener a algo o alguien quien lo apoye, como sería conectar los datos del formulario y garantizar que se almacenen correctamente. Se están evaluando soluciones para superar este obstáculo y asegurar que los nuevos pacientes puedan completar su registro sin problemas.

## ¿CÓMO RESOLVI EL PROBLEMA?

El desarrollador se comunicó con un colega con experiencia en integración de bases de datos y Spring Boot. Juntos, identificaron la raíz del problema y encontraron una solución que permitió conectar el cuestionario a la base de datos correctamente. Gracias a esta colaboración, logramos asegurar que los datos de los nuevos pacientes se almacenen sin inconvenientes, sumado a esto, el desarrollador ha recibido unos comentarios o sugerencias que le servirán de lección para las próximas tareas a desarrollar.

## EVIDENCIA DE LA RUNION #1 (DOMINGO 29/09)

- DURACIÓN: 21:00 – 23:00 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Avances en el desarrollo del formulario de registro para pacientes (HU01):

- ✓ El equipo discutió el progreso en la implementación de los campos requeridos para el formulario, como nombre de usuario, contraseña y datos personales.
- ✓ Se comentará sobre los primeros retos en la conexión del formulario al servidor.

### Configuración del cuestionario de pacientes nuevos (HU02):

- ✓ Se avanzó en el diseño y la implementación de preguntas sobre peso, historial de dietas y frecuencia de ejercicio.

## AVANCE DEL DISEÑO DEL CUESTIONARIO

### Cuestionario

Este cuestionario nos ayudara a saber más de ti.  
Si no sabe como responder dale click a "Omitir por ahora".  
Puede responderlo en cualquier momento.

¿A que eres alergico?

Perímetro de cintura

Perímetro de muslo

Perímetro de brazo/bicep

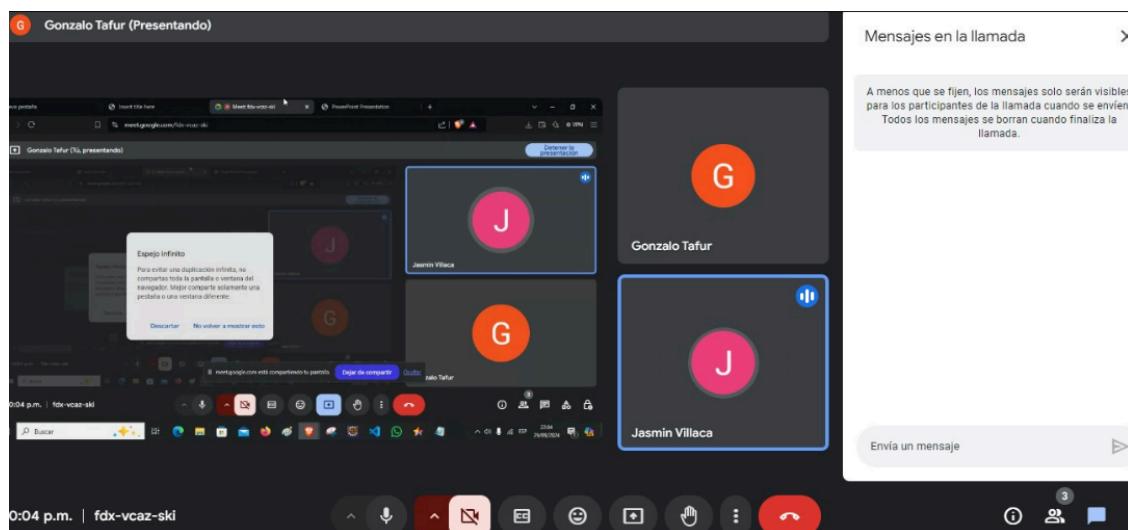
Su objetivo nutricional

Con que frecuencia hace ejercicio

Peso corporal

Perímetro de cadera

[Guardar e ir al menú](#) [Omitir por ahora](#)



## EVIDENCIA DE LA RUNION #2 (DOMINGO 06/10)

- DURACIÓN: 21:00 – 24:00 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Progreso en la integración del cuestionario de pacientes (HU02)

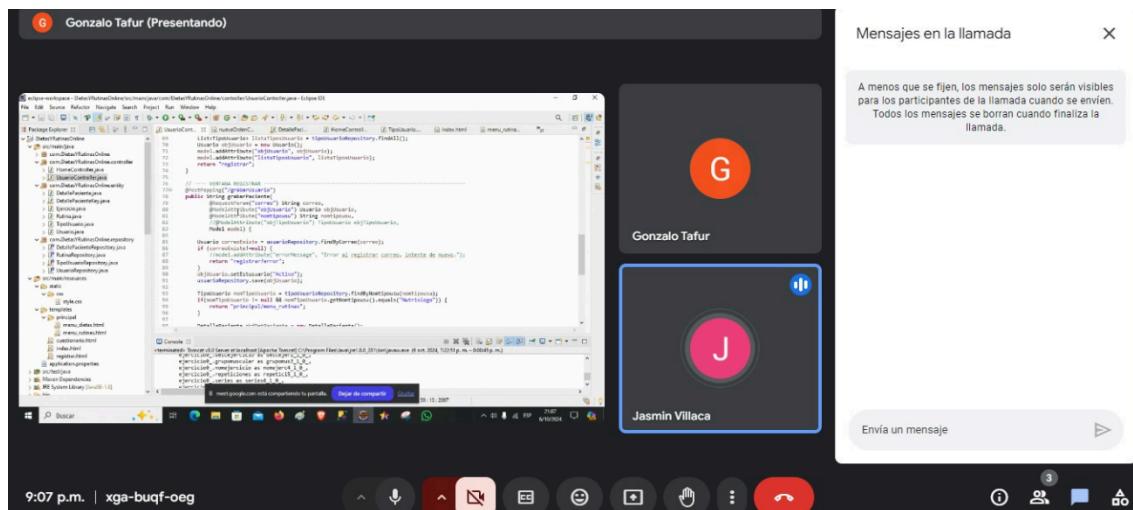
- ✓ Se discutió el estado del cuestionario de pacientes, donde se avanzó en las preguntas, pero aún faltaba la correcta integración a la base de datos

### Inicio del trabajo en el Menú de Inicio (HU04):

- ✓ Se comenzó a desarrollar la interfaz del menú de inicio, incluyendo la barra de navegación superior con el ícono de usuario y las notificaciones, además de un banner de bienvenida.

### Impedimentos:

- ✓ Integración del cuestionario con la base de datos: Se identificó un problema en la integración de los datos del cuestionario con la base de datos. Este se planteó como un obstáculo importante a resolver en la semana siguiente.
- ✓ El desarrollador se comprometió a buscar ayuda de un colega experto en bases de datos para resolver el impedimento.

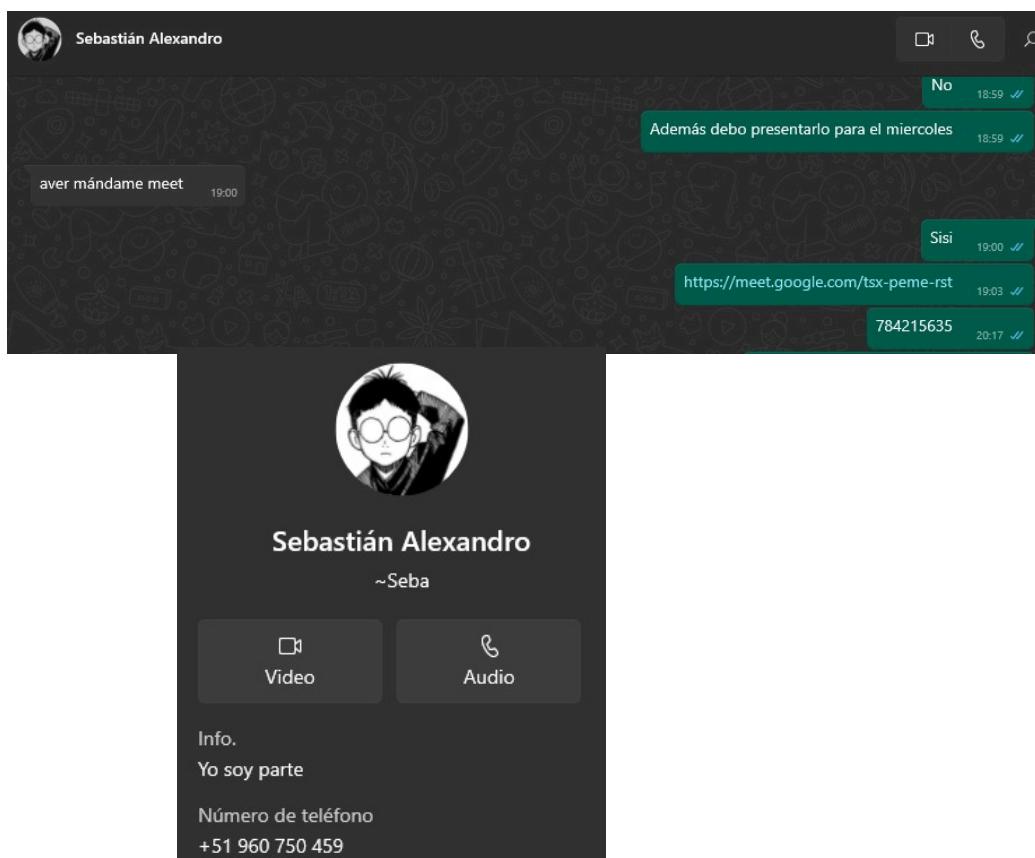


Durante esta reunión presentamos el problema relacionado con las redirecciones de los tipos de usuario y la integración del cuestionario de pacientes con la base de datos, lo que llevó a la colaboración entre el desarrollador y un colega para encontrar una solución adecuada.

## EVIDENCIA DE LA REUNIÓN DE RESOLUCIÓN DE IMPEDIMENTOS - INTEGRACIÓN DEL CUESTIONARIO DE PACIENTES (LUNES 07/10)

- PARTICIPANTES: Tafur Bermúdez Gonzalo, Sebastian Alejandro

Durante esta reunión, el desarrollador y un colega experimentado discutieron y solucionaron el problema de la integración del cuestionario de pacientes con la base de datos. Se identificó la raíz del problema y se implementaron ajustes en el código para garantizar que los datos de los nuevos pacientes se almacenen correctamente. El desarrollador también recibió sugerencias sobre buenas prácticas para futuras implementaciones.



### 5.4 Sprint Review:

Se presentará un resumen del incremento del producto completado durante este primer sprint.

### ALCANCE DEL SPRINT

- Desarrollo de la interfaz de registro para usuarios.
- Creación de un cuestionario para pacientes nuevos.
- Implementación de la interfaz de inicio de sesión.
- Diseño del menú de inicio de la aplicación.

## RESULTADOS DEL SPRINT

### o INTERFAZ DE REGISTRO PARA USUARIOS

En la siguiente imagen, se puede visualizar la interfaz de registro diseñada para los usuarios. Esta sección permite a los nuevos usuarios ingresar sus credenciales y datos personales, como el rol o tipo de usuario (paciente y nutriólogo) nombres, apellidos, correo electrónico, fecha de nacimiento, nacionalidad y sexo. Al completar este formulario, ya tendrán acceso a la aplicación para comenzar su recorrido hacia un estilo de vida más saludable.

The screenshot shows a web browser window with the URL [localhost:8080/DietasYRutinasOnline/usuario/registrarCuenta](http://localhost:8080/DietasYRutinasOnline/usuario/registrarCuenta). The page title is "Registrate". The form fields include:

- A dropdown menu labeled "¿Eres paciente o nutriólogo?" with options: Paciente (selected), Paciente, Nutriólogo, and Diego.
- Text input field for "Apellido Paterno" containing "Sanchez".
- Text input field for "Apellido Materno" containing "Alvarez".
- Text input field for "Fecha nacimiento" containing "19/04/2004".
- Text input field for "Nacionalidad".

### o CUESTIONARIO PARA PACIENTES NUEVOS

En esta imagen, se presenta el cuestionario que los nuevos pacientes deben completar tras registrarse. Este cuestionario recopila información clave sobre la salud del paciente, incluyendo peso, historial de dietas, frecuencia de ejercicio y alergias. Los datos recolectados son esenciales para personalizar las rutinas y dietas, garantizando una atención adecuada y efectiva. En caso que no sabe o no se acuerda de su información, puede saltar el cuestionario e ir al menú, pero puede llenarlo en otro momento actualizándolo en su perfil o por notificaciones.

The screenshot shows a web browser window with the URL [localhost:8080/DietasYRutinasOnline/usuario/grabarUsuario](http://localhost:8080/DietasYRutinasOnline/usuario/grabarUsuario). The page title is "Cuestionario". The form fields include:

Este cuestionario nos ayudara a saber más de ti.  
Si no sabe como responder dale click a "Omitir por ahora".  
Puede responderlo en cualquier momento.  
Con que frecuencia hace ejercicio

¿A que eres alérgico?

Peso corporal  
0

Perímetro de cintura  
0

Perímetro de cadera  
0

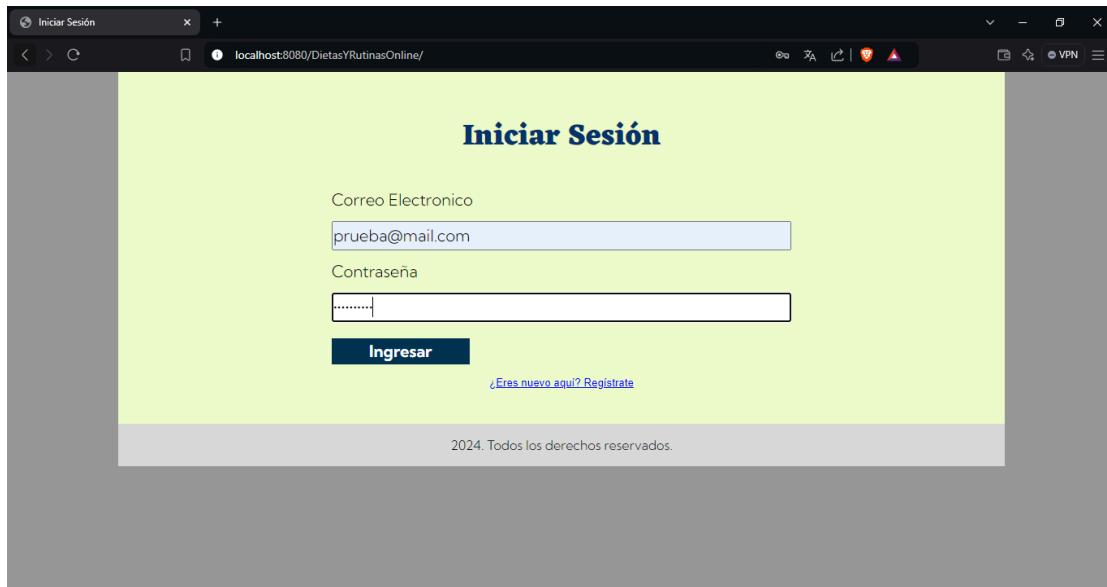
Perímetro de muslo  
0

Perímetro de brazo/bicep  
0

¿Tienes un objetivo nutricional?

## ○ INTERFAZ DE INICIO DE SESIÓN PARA USUARIOS

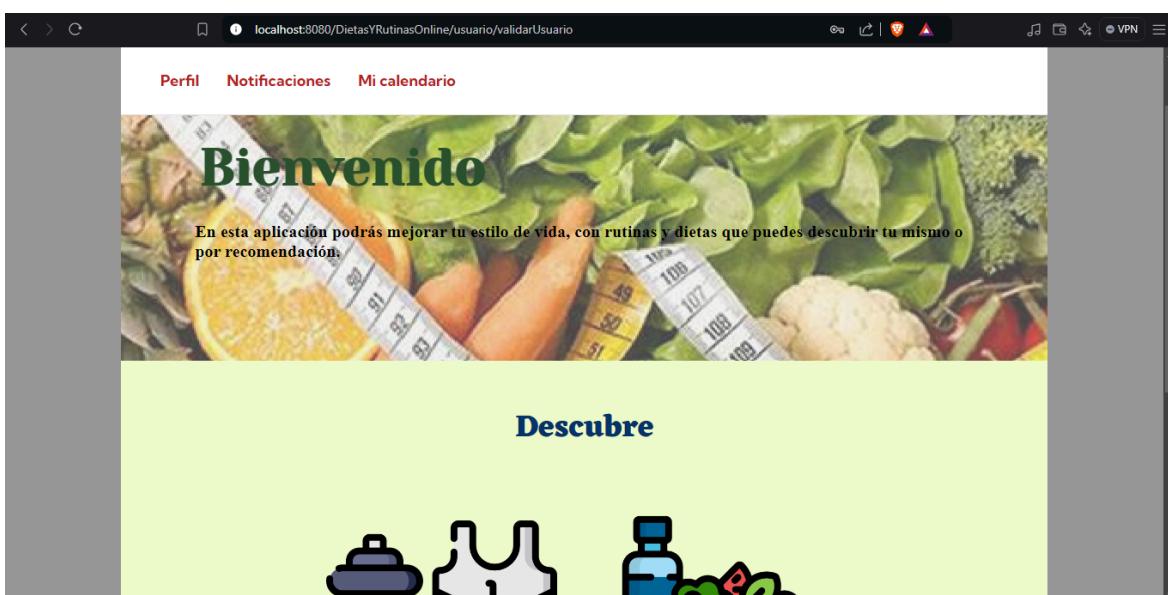
En la siguiente imagen, se puede visualizar la interfaz de inicio de sesión diseñada para los usuarios. Esta sección permite a los usuarios ingresar sus credenciales de correo electrónico y contraseña. Al completar esto, los usuarios pueden acceder a la aplicación y comenzar su recorrido hacia un estilo de vida más saludable.



The screenshot shows a web browser window with the title 'Iniciar Sesión'. The URL in the address bar is 'localhost:8080/DietasYRutinasOnline/'. The page has a light green header with the title 'Iniciar Sesión' in bold blue text. Below the header, there are two input fields: 'Correo Electronico' containing 'prueba@mail.com' and 'Contraseña' with a masked password. A dark blue 'Ingresar' button is positioned below the fields. At the bottom of the page, a grey footer bar contains the text '2024. Todos los derechos reservados.' and a link '¿Eres nuevo aquí? Regístrate'.

## ○ MENÚ DE INICIO DE LA APLICACIÓN

Aquí se muestra la interfaz del menú de inicio de la aplicación. Después de registrarse o iniciar sesión, los usuarios encontrarán una barra de navegación superior con acceso a las diferentes funcionalidades, incluyendo rutinas, dietas y calendario. Esta organización permite a los pacientes navegar de manera sencilla y eficiente, mejorando su experiencia en la aplicación.



## OBSERVACIONES DEL PRODUCT OWNER

Durante la presentación del producto del primer sprint, el Product Owner nos proporcionó las siguientes observaciones sobre el mismo:

*Los formularios de registro y de cuestionario pueden tener cambios en algunos campos, como el de nacionalidad, sexo, o cada cuanto haces ejercicio, es sugerible añadir un select en lugar de input para que el usuario tenga la facilidad de elegir y también para próximas tareas que se necesite recuperar o cambiar los atributos por nombre sin que se vean diferentes como “masculino” o “Masculino”, o los campos de peso corporal o perímetros para dar un límite correcto, y como además tenemos peso, faltaría consultar por su estatura, y en la base de datos los dos deberían ser decimales.*

*Volviendo a la parte del cuestionario, se hallaron múltiples dificultades, el más notable es el hecho que el método para grabar el cuestionario no esté recibiendo el id del paciente registrado, aún si accede al menú sin problemas, en la base de datos se puede observar no está registrado. Hay que observar más esa parte ya con la corrección en la tabla Cuestionario (o DetallePaciente si se le cambia de nombre).*

*El menú principal está listo en la parte de diseño y un poco las condiciones de vista, pero también falta el acceso y diseño a las demás ventanas además de la lista de dietas y rutinas que aún no tiene datos en la base de datos que no sean de prueba.*

## SUGERENCIAS PARA MEJORAR

- **Revisión de campos de formulario:**  
Implementar menús desplegables para opciones predefinidas, asegurando una entrada de datos más uniforme.
- **Integración de ID del Paciente:**  
Investigar y corregir el método de grabación del cuestionario para garantizar que el ID del paciente registrado se almacene correctamente.

## ACCESO A LA PLATAFORMA DE DESARROLLO

- **Acceso:**

Actualmente, se está investigando el tema del hosting para la aplicación. Una vez definido el servicio de hosting, se proporcionarán instrucciones detalladas sobre cómo acceder y utilizar la plataforma.

Para demostrar la funcionalidad de la aplicación hasta la fecha, se tienen evidencias en video que muestran el uso.

- **Evidencias en Video:**

<https://youtu.be/C-Rk0IQcz7k>

## 5.5 Sprint Retrospective:

**Fecha y Hora:** 9 oct 2024 11:0 am - 11:40am

**Participantes:**

- ✓ Product owner
- ✓ Equipo de desarrollo

Sección tratada	Observación	Lo vamos a tomar SI/NO	Sustentación
Cambiar algunos campos de los formularios	Los formularios de registro y de cuestionario pueden tener cambios en algunos campos, como usar un select en lugar de input para que el usuario tenga la facilidad de elegir y que tambien para proximas tareas que se necesite recuperar o cambiar los atributos por nombre sin que se vean diferentes como "masculino" o "Masculino", y los campos de peso corporal o perímetros para dar un límite correcto.	SI	Vamos a revisar el código para configurar los campos y ver los datos que debemos añadir por nombre, o sino por id en caso que necesitemos crear más tablas en la base de datos.
Cuestionario que no recibe el id del usuario.	El cuestionario está recibiendo todos los datos menos el id del paciente registrado que debería ser indispensable, aún si accede al menú sin problemas, en la base de datos se puede observar que no está registrado.	SI	Debido a que esta HU dependerá para las que se realizarán en el Sprint 3, habrá tiempo para solucionar ese problema. Veremos si podemos cambiarlo para una información de perfil.

Acceso para las ventanas del menú.	En el menú hace accesos para a las demás ventanas además de la lista de dietas y rutinas que aún no tiene datos en la base de datos que no sean de prueba.	SI	Para el desarrollo de otras HU empezaremos a diseñar las interfaces y aplicar sus funcionalidades.
------------------------------------	--	----	--

## Acuerdos Finales

- Se decidió implementar los cambios sugeridos en los formularios para mejorar la experiencia del usuario.
- Se acordó dar prioridad a la solución del problema con el ID del usuario en el cuestionario en el próximo sprint.
- Se establecerán tiempos específicos para la creación de las interfaces de las ventanas restantes en función de las historias de usuario (HU) planificadas.

## 5.6 Product Backlog [actualizado]:

ID	HISTORIA DE USUARIO	PRIORIDAD	ESTIMACION	DEPENDENCIA	SPRINT
<b>HU01</b>	Registrar usuario	1000	4 días	--	1
<b>HU03</b>	Acceso para usuarios	988	2 días	HU01	1
<b>HU04</b>	Menú de Inicio	980	5 días	HU01, HU03	1
<b>HU05</b>	Insertar rutinas personalizadas	975	5 días	HU01, HU03, HU06	2
<b>HU06</b>	Insertar dietas personalizadas	971	5 días	HU01, HU03	2
<b>HU02</b>	Cuestionario para pacientes nuevos	970	6 días	HU01	1
<b>HU08</b>	Ver detalles de rutinas y dietas	965	1 dia	HU08, HU09	2
<b>HU10</b>	Perfil de nutriólogos	963	4 días	HU02	3
<b>HU11</b>	Perfil de pacientes	960	1 dia	HU	3
<b>HU07</b>	Editar rutinas y dietas	955	5 días	HU01, HU03	2
<b>HU12</b>	Ver lista de ejercicios	945	4 días	HU01, HU02, HU06	2
<b>HU09</b>	Ver lista de alimentos	935	1 dia		2

<b>HU14</b>	Personalizar calendario de paciente	920	7 días	HU05	3
<b>HU15</b>	Feedback sobre rutinas y dietas	863	5 días	HU05, HU08, HU09	3
<b>HU16</b>	Notificaciones para usuario	860	8 días	HU12, HU13	4
<b>HU17</b>	Registro semanal de medidas y seguimiento de progreso	830	6 días	HU04	4
<b>HU19</b>	Recomendación Inicial de Dietas y Rutinas Personalizadas	820	10 días	HU04	4
<b>HU18</b>	Recomendaciones automáticas basadas en datos alterados	800	8 días	HU16	4
<b>HU13</b>	Ajustes de Usuario	780	1 día	HU02	4

## 6. SPRINT 2

### 6.1 Sprint Backlog:

#### PERSONALIZACIÓN Y ACCESO A RUTINAS Y DIETAS

- Fecha de Inicio del Sprint:** 9 de octubre del 2024.
- Fecha de Fin del Sprint:** 23 de octubre del 2024.
- Duración del Sprint:** 2 semanas

#### HISTORIAS DE USUARIOS

Este Sprint corresponde del 9/10/24 al 23/10/24, en el que se desarrollaron las historias de usuarios del sprint 2: HU05, HU06, HU07, HU08, HU09 y HU12

#### INCREMENTO PREVIO (SPRINT 1)

HU01 – Registrar usuario
Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.
<b>Criterios de aceptación:</b>
Escenario 1

La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.

El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.

## HU03 – Acceso para usuarios

Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario

### Criterios de aceptación:

Escenario 1

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

## HU02 – Cuestionario para pacientes nuevos

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

### Criterios de aceptación:

Escenario 1 – Cuestionario

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

## HU04 – Menú de Inicio

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

**Criterios de aceptación:**

Escenario 1 - Menú de inicio

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

**INCREMENTO ACTUAL (SPRINT 2)****HU05 - Insertar rutinas personalizadas**

Como nutriólogo, quiero crear y personalizar rutinas de ejercicios agregando descripciones específicas para recomendarlas a mis pacientes registrados.

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear rutinas, se le redirige a una ventana donde puede buscar ejercicios en la base de datos, seleccionarlos y organizarlos en una rutina personalizada. Además, el nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar la rutina según los objetivos específicos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la rutina y recomendarla a sus pacientes registrados.

**HU06 – Insertar dietas personalizadas**

Como nutriólogo, quiero crear y personalizar dietas agregando detalles específicos para recomendarlas a mis pacientes registrados.

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear dietas, se le redirige a una ventana donde puede ingresar y organizar planes de alimentación personalizados. El nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar las dietas según objetivos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la dieta y recomendarla a sus pacientes registrados.

## HU07 - Editar rutinas y dietas

Como nutriólogo, quiero poder editar los detalles de las dietas y rutinas para asegurarme de que la información esté actualizada y sea precisa.

### Criterios de aceptación:

#### Escenario 1 - Acción editar rutinas

Cuando el nutriólogo selecciona la opción para editar una rutina que él creó, podrá modificar sus datos, incluyendo el tipo de rutina, la categoría, su descripción y los ejercicios incluidos. Estas modificaciones permitirán mantener la base de datos actualizada.

#### Escenario 2 - Acción editar dietas

Cuando el nutriólogo selecciona la opción para editar una dieta que él creó, podrá modificar sus datos, incluyendo el tipo de dieta, la descripción, y los alimentos incluidos en esta dieta. Estas modificaciones permitirán mantener la base de datos actualizada.

## HU08 – Ver detalle de una rutina y dieta

Como paciente, quiero visualizar a detalle la dieta y rutina dentro de una lista para tenerla guardarla sea en calendario o por texto.

### Criterios de aceptación:

#### Escenario 1 – Ver detalle de una rutina

Cuando el usuario selecciona la opción para ver el detalle de una rutina, podrá visualizar sus datos, incluyendo el tipo de rutina, el nutriólogo quien lo creó, la categoría, su descripción y una lista de los ejercicios incluidos. También unos botones para descargarlo o guardarlo a tu calendario.

## Escenario 2 - Ver detalle de una dieta

Cuando el usuario selecciona la opción para ver el detalle de una dieta, podrá visualizar sus datos, incluyendo el tipo de dieta, el nutriólogo quien lo creó, su descripción y una lista de alimentos que debería consumir. También unos botones para descargarlo o guardarlo a tu calendario.

### HU09 – Ver lista de alimentos

Como nutriólogo, quiero ver una lista de alimentos disponibles en la base de datos para poder consultarlos y utilizarlos en la creación de dietas personalizadas para mis pacientes.

#### Criterios de aceptación:

##### Escenario 1

Cuando el nutriólogo accede a la sección de alimentos, se le muestra una lista de alimentos que incluye el nombre, tipo (desayuno, almuerzo, cena, merienda), y sus nutrientes principales (proteínas, grasas, carbohidratos). El nutriólogo puede buscar alimentos específicos, filtrarlos por tipo o por nutrientes, y seleccionar aquellos que desea incluir en las dietas de sus pacientes.

### HU12 – Ver lista de ejercicios

Como nutriólogo, quiero ver una lista de ejercicios en la base de datos para poder consultarlos y utilizarlos en la creación de rutinas personalizadas para mis pacientes.

#### Criterios de aceptación:

##### Escenario 1

Cuando el nutriólogo accede a la sección de ejercicios, se le muestra una lista que incluye el nombre del ejercicio, el grupo muscular al que pertenece, el tipo de ejercicio y una breve descripción. El nutriólogo puede buscar ejercicios específicos, filtrarlos por grupo muscular o tipo de ejercicio, y seleccionar aquellos que desea incluir en las rutinas de sus pacientes.

## 6.2 Sprint Planning:

## AGENDA TRATADA DURANTE LA CEREMONIA DE SPRINT PLANNING:

### e) Revisión de las Historias de Usuario del Sprint 2:

- Se discutió la prioridad de las HU a trabajar, subrayando la importancia de cada funcionalidad para el desarrollo de la aplicación.
- Se detallaron las tareas específicas para cada HU, asegurando que cada tarea requerirá más de 20 minutos.
- Se aclararon los criterios de aceptación para cada historia y se identificaron áreas donde se requerirá la colaboración de Jasmin Villaca para proporcionar la organización de los datos sobre ejercicios y alimentos brindados por el nutriólogo.

### f) Asignación de responsabilidades:

- Gonzalo Tafur Bermúdez fue asignado como responsable de las tareas técnicas, mientras que Jasmin Villaca colaborará proporcionando la organización de la información necesaria sobre ejercicios y alimentos en las HU correspondientes.
- Se distribuyeron las tareas, respetando la capacidad de Gonzalo para completar el trabajo dentro del Sprint y asegurando que Jasmin tenga tiempo para colaborar efectivamente.

### g) Determinación de dependencias y predecesoras:

- La HU07 (Editar rutinas y dietas) depende de la HU05 (Insertar rutinas personalizadas) y HU06 (Ver lista de alimentos y ejercicios).
- La HU08 (Ver detalle de una rutina y dieta) y HU09 (Ver lista de alimentos) dependen de la HU05 y HU06.
- La HU12 (Ver lista de ejercicios) también tiene como predecesoras la HU05 y HU06.

### h) Aprobación del Sprint Backlog:

- Se llegó a un acuerdo sobre la lista de tareas a realizar durante el sprint, así como la duración y el responsable de cada tarea.

## PROCEDIMIENTO SEGUIDO PARA LA CEREMONIA:

- FECHA: La ceremonia de planificación del Sprint 2 se llevó a cabo el 12/10/2024.
- PARTICIPANTES: Incluyó a todo el equipo
- RESULTADOS DE LA CEREMONIA:

✓ **Sprint Backlog:** Se seleccionan las HU y tareas del Sprint Backlog para el Sprint 2.

✓ **Duración de las tareas:** Cada tarea fue estimada y calendarizada para respetar la capacidad de Gonzalo, teniendo en cuenta la disponibilidad de Jasmin para proporcionar la organización de los datos.

- ✓ **Predecesoras:** HU08 y HU09 no pueden comenzar hasta que la HU05 y HU06 se hayan completado.
- ✓ **Evidencia:** Se presentaron prototipos de la interfaz para las nuevas funcionalidades, junto con la recopilación de datos de ejercicios y alimentos que Jasmin proveerá para completar las HU.

## PROTOTIPO INTERFAZ DE DIETAS Y ALIMENTOS

### Alimentos

Dieta	Alimentos		
Alimento	Nutrientes	Tipo	Descripción
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.
Pan con Palta y Huevo	20g de proteína, 50g de carbohidratos, 18g de grasas	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteínas.
Arroz con Pollo	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias y aceite vegetal.
Lentejas con Carne	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteínas para volumen.
Papa Rellena con Carne	22g de proteína, 50g de carbohidratos, 12g de grasas	Cena	Papas rellenas con carne molida de res (100g), con cebolla, ajo, y especias. Aporta calorías para recuperación nocturna.

## LISTA ORGANIZADA DE ALIMENTOS Y DIETAS

EJERCICIO						
IDEjercicio	Nombre Ejercicio	GrupMuscular	TipoEjercicio	Series	Repeticiones	DescEjercicio
1	Press de banca	Pecho	Tren Superior	3	6 a 8	Acuéstate en un banco plano con los pies en el suelo. Baja la barra hasta el pecho y empujala hacia arriba hasta estirar los brazos.
2	Press inclinado	Pecho	Tren Superior	3	6 a 8	En un banco inclinado, baja la barra o mancuernas hacia el pecho superior. Empuja hacia arriba contrayendo el pecho. Controla la bajada.
3	Remo con barra	Espalda	Tren Superior	3	6 a 8	Inclina el torso hacia adelante con la barra en las manos. Sube la barra hacia el abdomen y baja controladamente. Mantén la espalda recta.
4	Curl con barra	Biceps	Tren Superior	3	6 a 8	Sostén la barra con las palmas hacia arriba. Flexiona los codos elevando la barra hasta que los antebrazos toquen los bíceps. Baja lentamente.
5	Press francés	Tríceps	Tren Superior	3	6 a 8	Acuéstate en un banco, sujetá una barra Z. Flexiona los codos bajando la barra hacia la frente y luego extiende los brazos completamente.
6	Extensión de tricep Tríceps		Tren Superior	3	6 a 8	Sujeta la cuerda o barra en una polea alta. Empuja hacia abajo hasta estirar completamente los codos y vuelve a la posición inicial.
7	Elevaciones laterales Hombros		Tren Superior	3	6 a 8	De pie, con una mancuerna en cada mano, eleva los brazos hacia los lados hasta que estén paralelos al suelo. Baja lentamente.
8	Sentadilla con barra	Glúteos	Tren Inferior	3	6 a 8	Con la barra sobre los hombros, baja flexionando las rodillas hasta que los muslos estén paralelos al suelo. Sube extendiendo la pierna.
9	Peso muerto rumo Femorales		Tren Inferior	3	6 a 8	Baja la barra con las piernas ligeramente flexionadas y la espalda recta, hasta sentir el estiramiento en los isquiotibiales. Sube la barra.
10	Hip thrust	Glúteos	Tren Inferior	3	6 a 8	Apoya la parte superior de la espalda en un banco y coloca una barra sobre las caderas. Eleva las caderas contrayendo los glúteos.
11	Press de banca con Pecho		Tren Superior	4	12 a 15	Similar al press de banca regular, pero enfocado en más repeticiones (12-15), usando un peso moderado para aumentar la quemadura.
12	Remo con mancuera Espalda		Tren Superior	4	12 a 15	Con una mancuerna en una mano y apoyando la otra en un banco, lleva la mancuerna hacia el abdomen. Controla la bajada y la subida.
13	Curl con mancuera	Biceps	Tren Superior	4	12 a 15	Con las palmas hacia arriba, flexiona los codos levantando las mancuernas hacia los hombros. Usa un peso ligero y repeticiones altas.
14	Fondos en paralelo	Tríceps	Tren Superior	4	12 a 15	Con las manos en las barras paralelas, baja el cuerpo flexionando los codos y sube estirando los brazos. Mantén las piernas extendidas.
15	Elevaciones laterales Hombros		Tren Superior	4	12 a 15	Eleva los brazos hacia los lados con mancuernas ligeras hasta que estén paralelos al suelo. Mantén el control sin balancear el cuerpo.
16	Zancadas caminar	Glúteos	Tren Inferior	4	12 a 15	Da un paso largo hacia adelante y baja flexionando ambas rodillas. Alterna las piernas mientras avanzas, manteniendo la espalda recta.
17	Curl de piernas rectos		Tren Inferior	4	12 a 15	Acuéstate en la máquina de curl de piernas y flexiona las rodillas llevando los talones hacia los glúteos. Realiza series de repeticiones.
18	Hip thrust repetitivo	Glúteos	Tren Inferior	4	12 a 15	Eleva las caderas contrayendo los glúteos, pero con un peso moderado y más repeticiones para aumentar la quema calórica.
19	Elevaciones de tall Pantorrillas		Tren Inferior	4	12 a 15	Sube y bajas los talones manteniendo el peso sobre la parte delantera de los pies. Hazlo con repeticiones altas para enfocar el tren anterior.

ALIMENTO				
IDAlimento	NomAlimento	Nutrientes	TipoAlimento	DescAlimento
1	Batido de Plátano y 15g de proteína, Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.		
2	Pan con Palta y Huevos 20g de proteína, Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteínas.		
3	Arroz con Pollo 30g de proteína, Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias y aceite vegetal.		
4	Lentejas con Carné 25g de proteína, Cena	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteínas para volumen.		
5	Papa Rellena con Cebolla 22g de proteína, Cena	Papas rellenas con carne molida de res (100g), con cebolla, ajo, y especias. Aporta calorías para recuperación nocturna.		
6	Tallarines Verdes con 28g de proteína, Cena	Pasta (200g) con pesto de albahaca, espinacas y pollo a la plancha (150g). Alto en carbohidratos y proteínas.		
7	Yogurt con Quinua 12g de proteína, Merienda	Yogurt natural (200 ml) con quinua cocida (50g). Ideal para complementar la ingesta calórica del día.		
8	Batido de Proteínas 25g de proteína, Merienda	Polvo de proteína (30g) mezclado con leche (200 ml), plátano y fresas. Ayuda a alcanzar el requerimiento proteico.		
9	Batido de Frutas y Cereales 10g de proteína, Desayuno	Batido de plátano, fresas y chía (1 cda), mezclado con agua o leche vegetal. Ligero y nutritivo.		
10	Tostadas Integrales 7g de proteína, Desayuno	Tostadas de pan integral (2 rebanadas), con palta (30g) y tomate. Bajo en calorías, alto en fibra y grasas saludables.		
11	Ensalada de Pollo y 25g de proteína, Almuerzo	Quinua (100g) con pollo a la plancha (150g), espinaca, tomate, y palta. Rico en proteínas y bajo en grasas.		
12	Pescado a la Plancha: 30g de proteína, Almuerzo	Filete de pescado (150g) a la plancha, acompañado de camote sancochado (100g). Bajo en calorías, perfecto para déficit.		
13	Sopa de Verduras con 15g de proteína, Cena	Sopa a base de verduras (zapallo, zanahoria, espinaca) y pechuga de pollo desmenuzada. Ligero y bajo en calorías.		
14	Tortilla de Claras con 12g de proteína, Cena	Tortilla hecha con claras de huevo (4) y espinacas. Ideal para una cena ligera y alta en proteinas.		
15	Yogurt Descremado 8g de proteína, Merienda	Yogurt descremado (200g) con almendras (15g). Aporta saciedad sin aumentar mucho las calorías.		
16	Gelatina de Fresa con 5g de proteína, Merienda	Gelatina baja en calorías (1 taza) con una cucharada de chía. Refrescante y ligera para un déficit calórico.		

Tareas/Tareas no técnicas	Estimación	Asignación de Trabajo
---------------------------	------------	-----------------------

HU05 – Insertar rutinas personalizadas		
Realizar la Interfaz de Creación de Rutinas	2H	Tafur Bermúdez, Gonzalo
Implementar la funcionalidad para buscar y seleccionar ejercicios en la base de datos.	2H	Tafur Bermúdez, Gonzalo
Organizar datos de las rutinas proporcionadas por el nutriólogo	40min	Villaca Mamani, Jasmin
Permitir la inclusión de descripciones, objetivos, duración y nivel de dificultad.	1H	Tafur Bermúdez, Gonzalo
Configurar la categorización de rutinas según objetivos (volumen, déficit, recomposición corporal).	20min	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la creación de rutinas.	1h	Tafur Bermúdez, Gonzalo

<b>HU06 – Insertar dietas personalizadas</b>		
Realizar la Interfaz de Creación de Dietas	2h	Tafur Bermúdez, Gonzalo
Implementar la funcionalidad para buscar y seleccionar alimentos en la base de datos.	2h	Tafur Bermúdez, Gonzalo
Organizar datos de las dietas proporcionadas por el nutriólogo	40min	Villaca Mamani, Jasmin
Permitir la inclusión de descripciones, objetivos y tipo de comida.	1h	Tafur Bermúdez, Gonzalo
Configurar la categorización de dietas según tipo (desayuno, almuerzo, cena, merienda).	2h	Tafur Bermúdez, Gonzalo
Guarde y recomiende la dieta a los pacientes registrados.		Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la creación de dietas.	1h	Tafur Bermúdez, Gonzalo

<b>HU07 – Editar rutinas y dietas</b>		
Realizar la Interfaz de Edición de Rutinas	30min	Tafur Bermúdez, Gonzalo
Implementar la opción de editar detalles de rutinas (tipo, categoría, descripción).	2h	Tafur Bermúdez, Gonzalo
Implementar la opción de editar detalles de dietas (tipo, descripción, alimentos).	1h	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la edición.	30min	Tafur Bermúdez, Gonzalo

<b>HU08 – Ver detalle de una rutina y dieta</b>		
Realizar la Interfaz para Visualizar Detalles de Rutinas y Dietas	2h	Tafur Bermúdez, Gonzalo
Implementar la visualización de datos de la rutina (tipo, nutriólogo, categoría, ejercicios).	3h	Tafur Bermúdez, Gonzalo
Implementar la visualización de datos de la dieta (tipo, nutriólogo, alimentos).	3h	Tafur Bermúdez, Gonzalo
Permitir la inclusión de descripciones, objetivos, duración y nivel de dificultad.	1h	Tafur Bermúdez, Gonzalo
Agregar botones para guardar la rutina y dieta en el calendario.	20min	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la visualización.	3h	Tafur Bermúdez, Gonzalo

<b>HU09 – Ver lista de alimentos</b>		
Realizar la interfaz para ver la lista de alimentos	30min	Tafur Bermúdez, Gonzalo
Implementar la visualización de la lista de alimentos (nombre, tipo, nutrientes).	1h	Tafur Bermúdez, Gonzalo
Organizar datos de alimentos proporcionados por el nutriólogo	40min	Villaca Mamani, Jasmin
Implementar funcionalidades de búsqueda y filtrado por tipo o nutrientes.	1h 30min	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la lista de alimentos.	50min	Tafur Bermúdez, Gonzalo

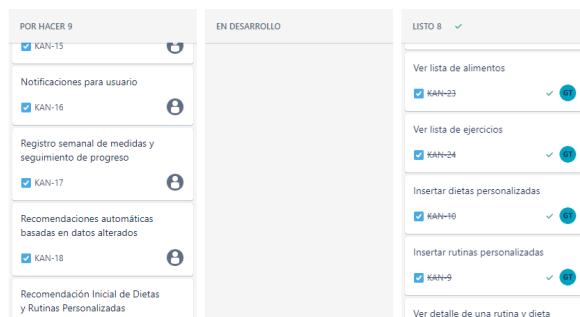
HU12 – Ver lista de ejercicios		
Realizar la interfaz para ver la lista de ejercicios	30min	Tafur Bermúdez, Gonzalo
Implementar la visualización de la lista de ejercicios (nombre, grupo muscular, tipo, descripción).	1h	Tafur Bermúdez, Gonzalo
Organizar datos de ejercicios proporcionados por el nutriólogo	40min	Villacá Mamani, Jasmin
Implementar funcionalidades de búsqueda y filtrado por grupo muscular o tipo.	1h	Tafur Bermúdez, Gonzalo
Realizar pruebas de usabilidad y funcionalidad de la lista de ejercicios.	50min	Tafur Bermúdez, Gonzalo

Este segundo sprint se centra en el desarrollo de funcionalidades clave para la gestión de rutinas y dietas personalizadas en la aplicación web. El objetivo principal es facilitar a los nutriólogos la creación, edición y visualización de rutinas y dietas específicas para sus pacientes, asegurando que cada plan se ajuste a las necesidades individuales de salud y objetivos. También se busca implementar una interfaz que permita a los nutriólogos acceder a listas de alimentos y ejercicios, facilitando así la personalización de sus recomendaciones.

Durante la ceremonia de Sprint Planning, el equipo, que incluye al Scrum Master, Product Owner y el programador, se reunió para revisar el progreso del Sprint 1 y planificar las actividades del Sprint 2. Se discutieron las historias de usuario que se abordarán, enfocándose en la creación de rutinas y dietas, así como en la posibilidad de editar y visualizar los detalles de estos planos. Además, se revisarán las listas de ejercicios y alimentos que se organizarán y se compartirán con el nutriólogo para asegurar que la información proporcionada sea precisa y útil para el desarrollo de las rutinas personalizadas. Se aprobaron los criterios de aceptación necesarios para garantizar que las funcionalidades cumplan con las expectativas de calidad.

## KANBAN BOARD JIRA (Contraseña y correo para el acceso)

[proyecdietasyrutinas@gmail.com](mailto:proyecdietasyrutinas@gmail.com)  
DyROtafvill



## 6.3 Daily Scrum:

Durante el Sprint 2, el equipo de desarrollo realizó reuniones semanales todos los domingos, con una duración de 1 a 2 horas, para sincronizar al equipo y abordar los obstáculos surgidos. En estas reuniones, se utilizaron preguntas específicas para evaluar el progreso y definir los siguientes pasos. Se lograron avances importantes en las tareas relacionadas con las historias de usuario HU05, HU06, HU07, HU08, HU09 y HU12, garantizando que el equipo estuviera alineado con los objetivos del proyecto.

### ¿QUÉ HICE AYER?

Ayer, el equipo avanzó en la organización de las listas de ejercicios y alimentos, asignando los datos correspondientes a cada dieta personalizada (HU06). Se elaborará la información proporcionada por el nutriólogo y se revisarán los registros anteriores para garantizar que estén correctamente categorizados en la base de datos. Además, se comenzaron las primeras pruebas para insertar dietas en la plataforma.

### ¿QUÉ HARÉ HOY?

Hoy, se finalizará la integración de la funcionalidad para ingresar dietas personalizadas (HU06) en la interfaz del sistema, asegurando que el nutriólogo pueda crear y agregar nuevas dietas a los usuarios. Por último, se iniciará el trabajo en la conexión entre los datos de ejercicios y las rutinas generadas (HU05).

### ¿QUÉ IMPEDIMENTOS TENGO?

Un problema surgió con la visualización de las rutinas (HU05), ya que algunos datos relacionados con la lista de ejercicios no se mostraban correctamente. Sin embargo, el programador identificó que el problema no radicaba en la base de datos, sino en las entidades del código que estaban configuradas de manera incorrecta. Este obstáculo fue superado con ajustes en el código.

### ¿CÓMO RESOLVI EL PROBLEMA?

El programador, Gonzalo, solucionó el problema al día siguiente de realizada la primera reunión de Daily Scrum, tras darse cuenta de que había un error en las entidades de las tablas, específicamente con dos atributos que estaban mal configurados como int. Esto fue corregido y la visualización de las rutinas en la plataforma fue restaurada. No se presentarán mayores complicaciones después de este ajuste.

## EVIDENCIA DE LA REUNIÓN #1 (DOMINGO 13/10)

- DURACIÓN: 22:00 – 23:00 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Avances en HU09 y HU12 – Ver lista de alimentos y ejercicios:

- ✓ Se logró completar la visualización de las listas de alimentos y ejercicios, donde los nutriólogos pueden consultar los detalles y hacer búsquedas por categoría o tipo.

### Avances en HU05 – Crear rutinas personalizadas:

- ✓ Se discutió el progreso en la creación de rutinas, permitiendo a los nutriólogos seleccionar ejercicios de la base de datos, agregar descripciones y categorizarlas.
- ✓ El equipo destacó los primeros desafíos en la organización de los ejercicios y las descripciones.

## Avances en HU06 – Crear dietas personalizadas:

- ✓ Se avanzó en la creación de dietas personalizadas, permitiendo a los nutriólogos añadir detalles específicos y clasificar las dietas según el objetivo de los pacientes.
- ✓ Se planteó la necesidad de un mejor filtrado en la lista de alimentos.

## Impedimentos:

- ✓ Se presentó un problema con la integración de los datos en la base de datos debido a un error en las entidades. Sin embargo, el desarrollador ajustó los tipos de datos incorrectos y resolvió el problema.

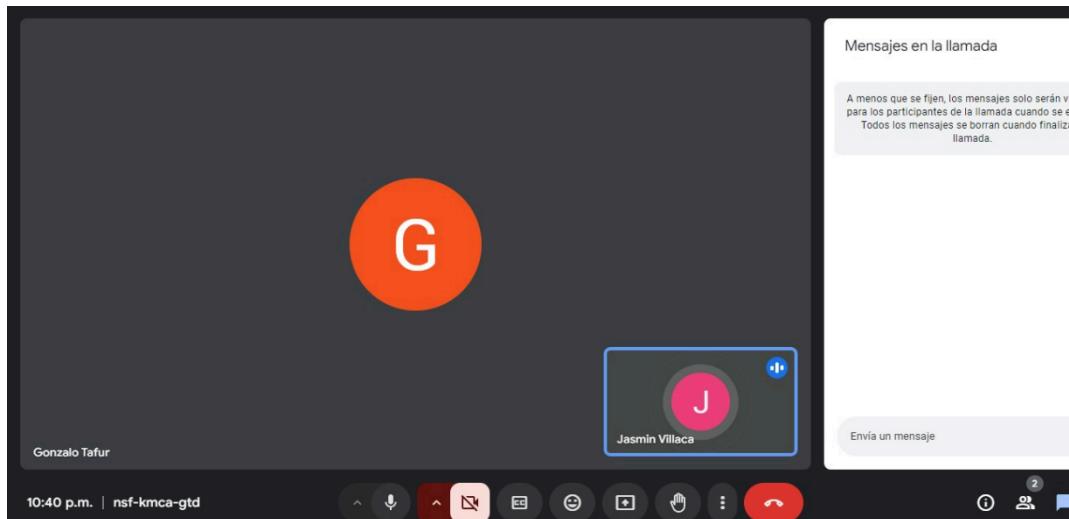
## AVANCE VER LISTA DE ALIMENTOS

Alimentos				
Dieta	Alimentos			
Alimento	Nutrientes	Tipo	Descripción	
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.	
Pan con Palta y Huevo	20g de proteína, 50g de carbohidratos, 18g de grasas	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteinas.	
Arroz con Pollo	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias y aceite vegetal.	
Lentejas con Carne	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteinas para volumen.	
Papa Rellena con Carne	22g de proteína, 50g de carbohidratos, 12g de grasas	Cena	Papas rellenas con carne molida de res (100g), con cebolla, ajo, y especias. Aporta calorías para recuperación nocturna.	

## EVIDENCIA DE IMPEDIMENTOS

```

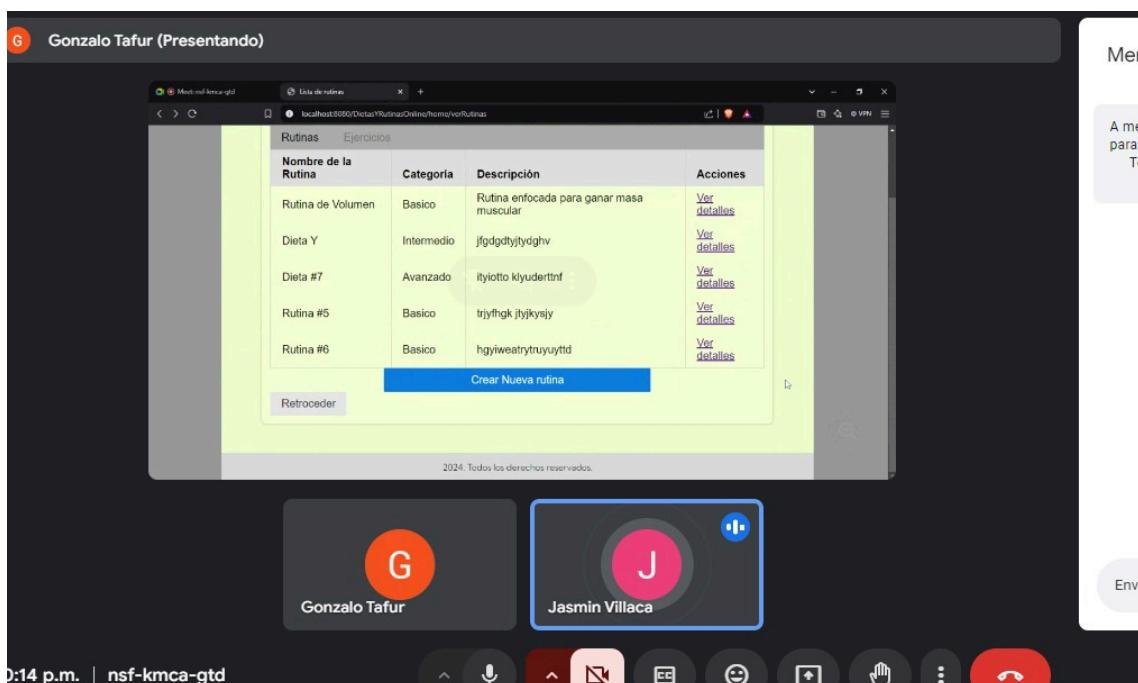
... org.springframework.dao.support.PersistenceExceptionTranslationInterceptor.invoke(PersistenceExceptionTranslationInterceptor.java:110)
... 60 common frames omitted
Caused by: com.microsoft.sqlserver.jdbc.SQLServerException: Error al convertir el valor varchar al tipo de datos JDBC INTEGER.
    at com.microsoft.sqlserver.jdbc.DBC.convertStreamToObject(DBC.java:811) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.ServerDTImpl.getValue(DT.java:3777) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.DTV.getValue(DT.java:247) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.Column.getValue(Column.java:100) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.SQLServerResultSet.getValue(SQLServerResultSet.java:2054) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.SQLServerResultSet.getvalue(SQLServerResultSet.java:2060) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.microsoft.sqlserver.jdbc.SQLServerResultSet.getint(SQLServerResultSet.java:2335) ~[msql-jdbc-9.2.1.jre8.jar:na]
    at com.zaxxer.hikari.pool.HikariProxyResultSet.getint(HikariProxyResultSet.java) ~[HikariCP-4.0.3.jar:na]
    at org.hibernate.type.descriptor.sql.IntegerTypeDescriptor$2.doExtract(IntegerTypeDescriptor.java:62) ~[hibernate-core-5.6.15.Final.jar]
    at org.hibernate.type.descriptor.sql.BasicExtractor.extract(BasicExtractor.java:47) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.type.AbstractStandardBasicType.nullSafeGet(AbstractStandardBasicType.java:257) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.type.AbstractStandardBasicType.nullSafeGet(AbstractStandardBasicType.java:253) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.type.AbstractStandardBasicType.nullSafeGet(AbstractStandardBasicType.java:243) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.type.AbstractStandardBasicType.hydrate(AbstractStandardBasicType.java:329) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.persister.entity.AbstractEntityPersister.hydrate(AbstractEntityPersister.java:3214) ~[hibernate-core-5.6.15.Final.jar]
    at org.hibernate.loader.Loader.loadFromResultSet(Loader.java:1887) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.hydrateEntityState(Loader.java:1811) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.instanceNotYetLoaded(Loader.java:1784) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.getRow(Loader.java:1624) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.getRowFromResultSet(Loader.java:748) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.getRowsFromResultSet(Loader.java:1047) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.loadFromResultSet(Loader.java:998) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.executeQuery.ExecuteReader_(Loader.java:967) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.executeQueryAndInitializeNonLazyCollections(Loader.java:357) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    at org.hibernate.loader.Loader.list(Loader.java:2868) ~[hibernate-core-5.6.15.Final.jar:5.6.15.Final]
    ...
100 common frames omitted
Caused by: java.lang.NumberFormatException: For input string: "Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y fi
at java.lang.NumberFormat.createNumberForInputString(Unknown Source) ~[na:1.8.0_281]
```



## EVIDENCIA DE LA RESOLUCIÓN DE IMPEDIMENTOS - PROBLEMA CON LA VISUALIZACIÓN DE RUTINAS (LUNES 14/10)

- PARTICIPANTES: Tafur Bermúdez Gonzalo, Jasmin Xiomi Villaca Mamani
- DURACIÓN: 22:00 – 22:20 horas
- TEMAS TRATADOS:

Durante esta reunión, Gonzalo, el programador, explicó el problema relacionado con la visualización incorrecta de algunos datos de las rutinas personalizadas. Tras revisar el código, se descubrió que el error estaba en la configuración de dos atributos como int en lugar de string. Una vez corregido, las rutinas comenzaron a mostrarse adecuadamente en la plataforma.



## EVIDENCIA DE LA REUNIÓN #2 (DOMINGO 22/10)

- DURACIÓN: 22:00 – 23:00 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Progreso en HU07 – Editar rutinas y dietas:)

- ✓ Se discutió la implementación de la funcionalidad para que los nutriólogos puedan editar tanto rutinas como dietas. Se trabajó en la opción de modificar descripciones y categorías.

### Progreso en HU08 – Visualizar detalle de rutinas y dietas:

- ✓ El equipo avanzó en la opción de que los pacientes puedan visualizar a detalle una rutina o dieta específica, con botones para descargarlas o guardarlas en su calendario.

## PROGRESO VER DETALLES DE RUTINAS Y DIETAS

Dieta #1			
Descripción: Esta dieta es ideal para los			
Creado por: Jose			
Alimentos incluidos			
Alimento	Nutrientes	Tipo	Descripción
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.
Pan con Palta y Huevo	20g de proteína, 50g de carbohidratos, 18g de grasas	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteínas.
Tallarines Verdes	28g de proteína, 90g de carbohidratos, 18g de	Cena	Pasta (200g) con pesto de albahaca, espinacas y pollo a la plancha (150g). Alto

## PROGRESO EDICION RUTINAS Y DIETAS

Gonzalo Tafur (Presentando)

localhost:8080/DietasYRutinasOnline/usuario/grabarUsuario

Perímetro de muslo  
10

Perímetro de brazo/bicep  
10

¿Tienes un objetivo nutricional?  
*bajar de peso*

Grabar e ir al n° 1  
Omitir por ahora

2024. Todos los derechos reservados.

Gonzalo Tafur (Presentando)

Gonzalo Tafur

Jasmin Villaca

## 6.4 Sprint Review:

Se presentará un resumen del incremento del producto completado durante este segundo sprint.

### ALCANCE DEL SPRINT

- Desarrollo de la funcionalidad para insertar rutinas personalizadas.
- Desarrollo de la funcionalidad para insertar dietas personalizadas.
- Funcionalidad para ver el detalle de una rutina y una dieta.
- Implementación de la lista de ejercicios.
- Implementación de la lista de alimentos.
- Edición de rutinas y dietas por parte del nutriólogo.

### RESULTADOS DEL SPRINT

#### ○ INSERCIÓN DE RUTINAS PERSONALIZADAS

En la siguiente imagen, se visualiza la interfaz donde el nutriólogo puede crear rutinas personalizadas. Los nutriólogos pueden seleccionar ejercicios de la base de datos, categorizarlos según el objetivo (volumen o déficit calórico), añadir descripciones específicas, definir el nivel de dificultad y duración. Además, de poder añadir recomendaciones en la descripción para los pacientes sobre los tiempos de descanso, el peso sugerido y otras pautas importantes para la correcta realización de las rutinas.

The screenshot shows a user interface for creating a personalized routine. At the top, there is a dropdown menu labeled "Tipo de rutina" with the following options: "Rutina de Deficit" (selected), "Rutina de Deficit", "Rutina de Volumen", and "Intermedio". Below this is a text area labeled "Descripción" containing the text "fjgdynbndfayjfb". At the bottom, there is a section labeled "Ejercicios" with a button labeled "Press de banca □".

## ○ INSERCIÓN DE DIETAS PERSONALIZADAS

En esta imagen, se presenta el cuestionario que los nuevos pacientes deben completar tras registrarse. Este cuestionario recopila información clave sobre la salud del paciente, incluyendo peso, historial de dietas, frecuencia de ejercicio y alergias. Los datos recolectados son esenciales para personalizar las rutinas y dietas, garantizando una atención adecuada y efectiva. En caso que no sabe o no se acuerda de su información, puede saltar el cuestionario e ir al menú, pero puede llenarlo en otro momento actualizándolo en su perfil o por

**Insertar nueva dieta**

Nombre o tipo de dieta  
Dieta para veganos

Descripción

¿Qué alimentos incluye?

Batido de Plátano y Avena  
 Pan con Palta y Huevo  
 Arroz con Pollo  
 Lentejas con Carne  
 Papa Rellena con Carne

notificaciones.

## ○ VER DETALLE DE UNA RUTINA Y DIETA

Esta sección permite a los pacientes visualizar a detalle las rutinas y dietas asignadas o recomendadas. En el caso de las rutinas, los usuarios pueden ver si están enfocadas en volumen o déficit, el grupo muscular que trabajan (tren superior, tren inferior o cuerpo completo), y además, se incluye un apartado de recomendaciones específicas para la ejecución: tiempos de descanso entre series, cargas sugeridas, y el número de repeticiones. Para las dietas, se muestra la lista de alimentos, las porciones y un botón para añadir la dieta a su calendario.

Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción
Press inclinado	Pecho	Tren Superior	3	6 a 8	En un banco inclinado, baja la barra o mancuernas hacia el pecho superior. Empuja hacia arriba contrayendo el pecho. Controla el movimiento para evitar lesiones.
Remo con barra	Espalda	Tren Superior	3	6 a 8	Inclina el torso hacia adelante con la barra en las manos. Sube la barra hacia el abdomen y baja controladamente. Mantén la espalda recta durante el ejercicio.

## o VER LISTA DE EJERCICIOS

La lista de ejercicios disponible para los nutriólogos contiene el nombre del ejercicio, el grupo muscular, y una breve descripción de cómo realizarlo. Esta sección está optimizada para que los nutriólogos puedan buscar fácilmente los ejercicios que necesitan al momento de crear nuevas rutinas para sus pacientes. Ademas de que se añadió un filtrado para los ejercicios

Rutinas	Ejercicios				
Tren Inferior					
Ejercicio	Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción
Sentadilla con barra	Cuádriceps	Tren Inferior	3	6 a 8	Con la barra sobre los hombros, baja flexionando las rodillas hasta que los muslos estén paralelos al suelo. Sube extendiendo las rodillas y caderas. Baja la barra con las piernas ligeramente flexionadas y la

## o VER LISTA DE ALIMENTOS

Se presenta una lista de alimentos clasificados por tipo de comida (desayuno, almuerzo, cena, merienda) y con información detallada sobre los nutrientes principales (proteínas, grasas y carbohidratos). Esto ayuda a los nutriólogos a seleccionar los alimentos adecuados para cada paciente y garantizar que las dietas sean balanceadas y adaptadas a los objetivos de cada usuario. Se añadió una opción de filtrado.

Alimentos			
Dieta	Alimentos		
Desayuno			
Alimento	Nutrientes	Tipo	Descripción
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.
Pan con Palta y Huevo	20g de proteína, 50g de carbohidratos, 18g de grasas	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteínas.
Batido de Frutas y Chía	10g de proteína, 45g de carbohidratos, 5g de grasas	Desayuno	Batido de plátano, fresas y chía (1 cda), mezclado con agua o leche vegetal. Ligero y nutritivo.
Tostadas Integrales con Palta y Tomate	7g de proteína, 30g de carbohidratos, 10g de grasas	Desayuno	Tostadas de pan integral (2 rebanadas), con palta (30g) y tomate. Bajo en calorías, alto en fibra y grasas saludables.

## o EDICIÓN DE RUTINAS Y DIETAS

Se implementó la funcionalidad para que los nutriólogos editen tanto rutinas como dietas. Pueden actualizar descripciones, cambiar los ejercicios o alimentos incluidos y ajustar las recomendaciones de acuerdo con los progresos del paciente. Esta opción asegura que la información esté siempre actualizada y alineada con los objetivos actuales del paciente.

**Editar mi dieta**

Nombre de dieta  
Dieta #1

Descripción  
Esta dieta es ideal para los

¿Qué alimentos incluye?  
Batido de Plátano y Avena   
Pan con Palta y Huevo   
Arroz con Pollo   
Lentejas con Carne   
Papa Rellena con Carne   
Tallarines Verdes con Pollo   
Yogurt con Quinua   
Batido de Proteínas con Frutas

## OBSERVACIONES DEL PRODUCT OWNER (PROFESOR)

PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo, Marco Aurelio Espinoza Rivera (Profesor)

Es necesario que las descripciones de las rutinas y dietas sean más detalladas. Los pacientes no asumen que una rutina es válida solo para un día o cuántos días a la semana se debe seguir, o cuánto tiempo descansar entre series.

EVIDENCIA DE LA REUNION REALIZADA EL DIA 22/10/24 a las 20:40p

The screenshot shows a web application interface titled "Perfil". At the top, there is a header bar with the URL "localhost:8080/Dietas/RutinasOnline/home/verPerfil". Below the header, there are two tables: "Rutinas creadas" and "Dietas creadas".

**Rutinas creadas:**

Nombre de la Rutina	Categoría	Descripción	Acciones
Rutina de Volumen	Intermedio	Rutina enfocada para ganar masa muscular	<a href="#">Ver detalles</a>
Rutina de Deficit	Basico	Esta rutina es para deficit	<a href="#">Ver detalles</a>
Rutina de Volumen	Intermedio	Esta rutina es de volumen	<a href="#">Ver detalles</a>

**Dietas creadas:**

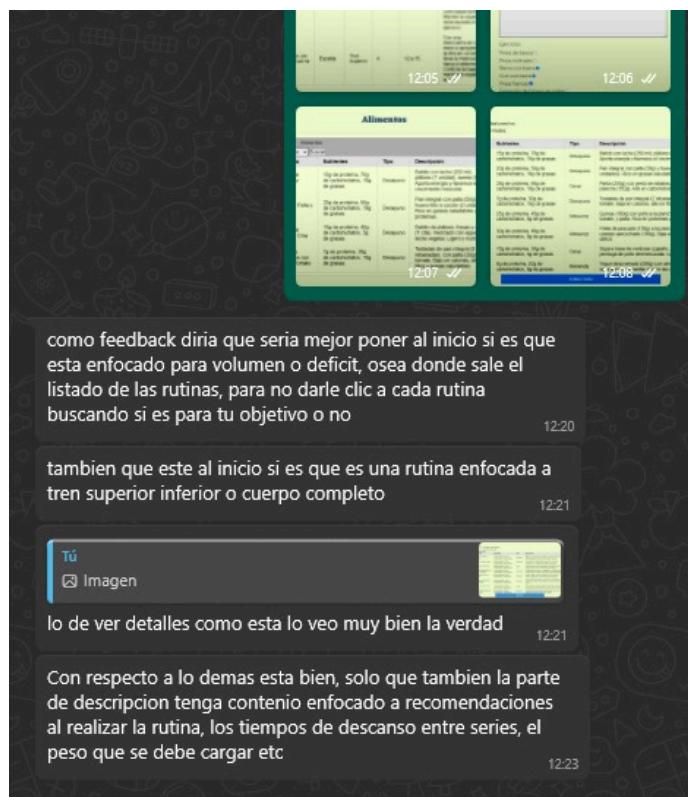
Tipo de dieta	Descripción	Acciones
Dieta #1	Esta dieta es ideal para los	<a href="#">Ver detalles</a>

## SUGERENCIAS PARA MEJORAR

- **Paciente:**

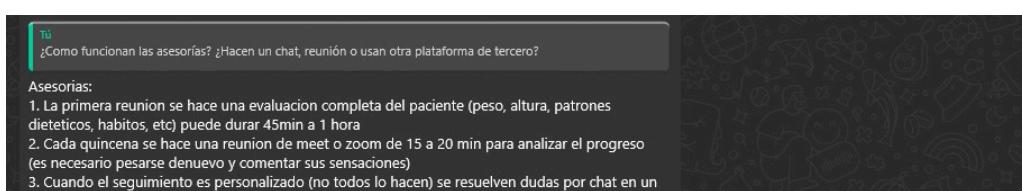
- ✓ **FILTROS PARA RUTINAS:** Los pacientes sugieren que sería más conveniente poder ver al inicio si la rutina es para volumen o déficit, y si está enfocada en tren superior, inferior o cuerpo completo, sin necesidad de entrar en cada una para averiguarlo.
- ✓ **DETALLES ADICIONALES EN LAS RUTINAS:** Se sugiere que las descripciones de las rutinas incluyan recomendaciones sobre el tiempo de descanso entre series, el peso sugerido y otros detalles importantes para realizar los ejercicios correctamente.

## EVIDENCIA DE COMUNICACIÓN CON EL USUARIO PACIENTE INTERESADO



- **Nutriólogo:**

- ✓ **ASESORÍAS QUINCENALES Y EVALUACIÓN INICIAL:** El nutriólogo sugiere añadir una funcionalidad para programar reuniones de asesoría. La primera reunión debería durar entre 45 minutos a 1 hora, y sería una evaluación completa del paciente (peso, altura, hábitos alimentarios, etc.). Luego, cada quincena, se realizaría una reunión de seguimiento de 15 a 20 minutos por Meet o Zoom, donde se evalúan el progreso, el peso actualizado, y se comentan las sensaciones del paciente.



## ACCESO A LA PLATAFORMA DE DESARROLLO

- **Acceso:**

Actualmente, se está investigando el tema del hosting para la aplicación. Una vez definido el servicio de hosting, se proporcionarán instrucciones detalladas sobre cómo acceder y utilizar la plataforma.

Para demostrar la funcionalidad de la aplicación hasta la fecha, se demostró en la reunión realizada con el producto owner (profesor)

## 6.5 Sprint Retrospective:

**Fecha y Hora:** 23 oct 2024 11:00 am - 11:40am

**Participantes:**

- ✓ Product owner
- ✓ Equipo de desarrollo

Sección tratada	Observación	Lo vamos a tomar SI/NO	Sustentación

Cambios en la presentación de rutinas	Sugerencia para mostrar al inicio si una rutina está enfocada en volumen o déficit, y también si es para tren superior, inferior o cuerpo completo.	SI	Vamos a priorizar esta mejora en la interfaz del listado de rutinas para facilitar la búsqueda de acuerdo a los objetivos de los pacientes.
Recomendaciones en las rutinas	Se solicita que las rutinas incluyan detalles sobre recomendaciones, tiempos de descanso entre series y pesos recomendados. Es importante incluir descripciones detalladas para evitar malentendidos sobre el uso de las rutinas.	SI	Implementaremos una sección detallada en cada rutina para incluir estas recomendaciones, asegurando que los pacientes entiendan cómo deben ejecutar cada ejercicio y qué esperar de la rutina.
Estructura de reuniones con el nutriólogo	Proponer reuniones iniciales para evaluar al paciente y sesiones quincenales para seguimiento del progreso.	SI	Se creará una nueva HU para programar estas reuniones. La primera reunión se enfocará en una evaluación completa del paciente, y las sesiones quincenales se utilizarán para revisar el progreso y ajustar las recomendaciones.
Información en el calendario	Es necesario incluir detalles en el calendario sobre que una rutina es solo para un día, que no puede haber dos rutinas en un día, y especificaciones sobre los días de descanso a la semana.	SI	Estos detalles se agregarán a la sección del calendario para que los pacientes comprendan la estructura y uso adecuado de las rutinas.

## Acuerdos Finales

- Se decidió implementar cambios en la presentación de rutinas para mostrar claramente si están enfocadas en volumen o déficit, así como el tipo de rutina (tren superior, inferior o cuerpo completo).

- Se acordó incluir recomendaciones detalladas en cada rutina, como tiempos de descanso, pesos sugeridos y consejos para la ejecución correcta de los ejercicios.
- Se programarán reuniones con el nutriólogo como parte de la nueva historia de usuario, con la primera reunión de evaluación del paciente y seguimiento quincenal para analizar el progreso.
- Se establecerán detalles en el calendario para indicar que una rutina es solo para un día, que no se pueden tener dos rutinas en un día y las recomendaciones sobre los días de descanso.

HU20 – Programar reuniones con el nutriólogo
Como paciente, quiero programar reuniones con mi nutriólogo para recibir evaluaciones y seguimiento de mi progreso.
<b>Criterios de aceptación:</b>
<p>Escenario 1</p> <p>Cuando el paciente acceda a la sección de programación de reuniones, podrá elegir entre una evaluación inicial completa o una reunión de seguimiento quincenal. Al llegar la fecha de la reunión, recibirá una notificación recordándole la cita y los detalles de acceso, como el enlace de Zoom o Meet. Además, tendrá la opción de reprogramar una reunión seleccionando la cita previamente agendada y eligiendo una nueva fecha y hora.</p>

## 6.6 Product Backlog [actualizado]:

ID	HISTORIA DE USUARIO	PRIORIDAD	ESTIMACION	DEPENDENCIA	SPRINT
<b>HU01</b>	Registrar usuario	1000	4 días	--	1
<b>HU03</b>	Acceso para usuarios	988	2 días	HU01	1
<b>HU04</b>	Menú de Inicio	980	5 días	HU01, HU03	1
<b>HU05</b>	Insertar rutinas personalizadas	975	5 días	HU01, HU03, HU06	2
<b>HU06</b>	Insertar dietas personalizadas	971	5 días	HU01, HU03	2
<b>HU02</b>	Cuestionario para pacientes nuevos	970	6 días	HU01	1
<b>HU08</b>	Ver detalles de rutinas y dietas	965	1 dia	HU08, HU09	2
<b>HU10</b>	Perfil de nutriólogos	963	4 días	HU02	3

<b>HU11</b>	Perfil de pacientes	960	1 dia	HU02	3
<b>HU07</b>	Editar rutinas y dietas	955	5 días	HU01, HU03	2
<b>HU12</b>	Ver lista de ejercicios	945	4 días	HU01, HU02, HU06	2
<b>HU09</b>	Ver lista de alimentos	935	1 dia		2
<b>HU14</b>	Personalizar calendario de paciente	920	7 días	HU05	3
<b>HU15</b>	Feedback sobre rutinas y dietas	863	5 días	HU05, HU08, HU09	3
<b>HU16</b>	Notificaciones para usuario	860	8 días	HU12, HU13	4
<b>HU17</b>	Registro semanal de medidas y seguimiento de progreso	830	6 días	HU04	4
<b>HU19</b>	Recomendación Inicial de Dietas y Rutinas Personalizadas	820	10 días	HU04	4
<b>HU18</b>	Recomendaciones automáticas basadas en datos alterados	800	8 días	HU16	4
<b>HU13</b>	Ajustes de Usuario	780	1 día	HU02	3
<b>HU20</b>	Programar reuniones con el nutriólogo	760	1 dia	HU10	4

## 7. SPRINT 3

### 7.1 Sprint Backlog:

#### PERFILES Y PERSONALIZACIÓN DEL CALENDARIO

- **Fecha de Inicio del Sprint:** 24 de octubre del 2024.
- **Fecha de Fin del Sprint:** 13 de noviembre del 2024.
- **Duración del Sprint:** 3 semanas

#### HISTORIAS DE USUARIOS

Este Sprint corresponde del 24/10/24 al 13/10/24, en el que se desarrollaron las historias de usuarios del sprint 3: HU05, HU06, HU07, HU08, HU09 y HU12

#### INCREMENTO PREVIO (SPRINT 1)

## HU01 – Registrar usuario

Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.

### Criterios de aceptación:

#### Escenario 1

La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.

El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.

## HU03 – Acceso para usuarios

Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario

### Criterios de aceptación:

#### Escenario 1

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

## HU02 – Cuestionario para pacientes nuevos

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

### Criterios de aceptación:

#### Escenario 1 – Cuestionario

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

## HU04 – Menú de Inicio

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

### Criterios de aceptación:

Escenario 1 - Menú de inicio

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

## INCREMENTO PREVIO (SPRINT 2)

## HU05 - Insertar rutinas personalizadas

Como nutriólogo, quiero crear y personalizar rutinas de ejercicios agregando descripciones específicas para recomendarlas a mis pacientes registrados.

### Criterios de aceptación:

Escenario 1

Cuando el nutriólogo accede a la sección de crear rutinas, se le redirige a una ventana donde puede buscar ejercicios en la base de datos, seleccionarlos y organizarlos en una rutina personalizada. Además, el nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar la rutina según los objetivos específicos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la rutina y recomendarla a sus pacientes registrados.

## HU06 – Insertar dietas personalizadas

Como nutriólogo, quiero crear y personalizar dietas agregando detalles específicos para recomendarlas a mis pacientes registrados.

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear dietas, se le redirige a una ventana donde puede ingresar y organizar planes de alimentación personalizados. El nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar las dietas según objetivos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la dieta y recomendarla a sus pacientes registrados.

## HU07 - Editar rutinas y dietas

Como nutriólogo, quiero poder editar los detalles de las dietas y rutinas para asegurarme de que la información esté actualizada y sea precisa.

**Criterios de aceptación:**

Escenario 1 - Acción editar rutinas

Cuando el nutriólogo selecciona la opción para editar una rutina que él creó, podrá modificar sus datos, incluyendo el tipo de rutina, la categoría, su descripción y los ejercicios incluidos. Estas modificaciones permitirán mantener la base de datos actualizada.

Escenario 2 - Acción editar dietas

Cuando el nutriólogo selecciona la opción para editar una dieta que él creó, podrá modificar sus datos, incluyendo el tipo de dieta, la descripción, y los alimentos incluidos en esta dieta. Estas modificaciones permitirán mantener la base de datos actualizada.

## HU08 – Ver detalle de una rutina y dieta

Como paciente, quiero visualizar a detalle la dieta y rutina dentro de una lista para tenerla guardarla sea en calendario o por texto.

**Criterios de aceptación:****Escenario 1 – Ver detalle de una rutina**

Cuando el usuario selecciona la opción para ver el detalle de una rutina, podrá visualizar sus datos, incluyendo el tipo de rutina, el nutriólogo quien lo creó, la categoría, su descripción y una lista de los ejercicios incluidos. También unos botones para descargarlo o guardarlo a tu calendario.

**Escenario 2 - Ver detalle de una dieta**

Cuando el usuario selecciona la opción para ver el detalle de una dieta, podrá visualizar sus datos, incluyendo el tipo de dieta, el nutriólogo quien lo creó, su descripción y una lista de alimentos que debería consumir. También unos botones para descargarlo o guardarlo a tu calendario.

**HU09 – Ver lista de alimentos**

Como nutriólogo, quiero ver una lista de alimentos disponibles en la base de datos para poder consultarlos y utilizarlos en la creación de dietas personalizadas para mis pacientes.

**Criterios de aceptación:****Escenario 1**

Cuando el nutriólogo accede a la sección de alimentos, se le muestra una lista de alimentos que incluye el nombre, tipo (desayuno, almuerzo, cena, merienda), y sus nutrientes principales (proteínas, grasas, carbohidratos). El nutriólogo puede buscar alimentos específicos, filtrarlos por tipo o por nutrientes, y seleccionar aquellos que desea incluir en las dietas de sus pacientes.

**HU12 – Ver lista de ejercicios**

Como nutriólogo, quiero ver una lista de ejercicios en la base de datos para poder consultarlos y utilizarlos en la creación de rutinas personalizadas para mis pacientes.

**Criterios de aceptación:****Escenario 1**

Cuando el nutriólogo accede a la sección de ejercicios, se le muestra una lista que incluye el nombre del ejercicio, el grupo muscular al que pertenece, el tipo de ejercicio y una breve

descripción. El nutriólogo puede buscar ejercicios específicos, filtrarlos por grupo muscular o tipo de ejercicio, y seleccionar aquellos que desea incluir en las rutinas de sus pacientes.

## INCREMENTO ACTUAL (SPRINT 3)

### PROCEDIMIENTO PARA LA ORGANIZACIÓN DEL BACKLOG Y PRIORIZACIÓN DE HISTORIAS DE USUARIO

Durante la planificación del Sprint 3, el equipo de desarrollo, en conjunto con el nutriólogo responsable, realizó una revisión exhaustiva del Backlog con el objetivo de seleccionar las Historias de Usuario (HU) que aportarían mayor valor al producto en esta fase. A continuación, se describe el procedimiento seguido y los acuerdos que se tomaron:

#### a) Revisión Inicial del Backlog

Se llevó a cabo una sesión de análisis del Backlog, donde el equipo evaluó todas las HU pendientes en función de su prioridad, estimación de tiempo, y dependencia con otras funcionalidades previamente desarrolladas. La finalidad era identificar aquellas HU que eran fundamentales para la experiencia del usuario, tanto para nutriólogos como para pacientes.

#### b) Acuerdos para la Priorización

El equipo consideró los siguientes criterios al priorizar las HU:

- **Impacto en la experiencia del usuario:** Se priorizaron funcionalidades que no solo mejorarán la interacción de nutriólogos y pacientes, sino que también sentarán las bases para las futuras funcionalidades del Sprint 4, enfocadas en mejorar la comunicación e interacción directa entre ambos.
- **Dependencias técnicas:** Se seleccionaron HU que tenían dependencias con funcionalidades ya implementadas en sprints anteriores, asegurando un flujo de trabajo más eficiente.
- **Facilidad de implementación:** Se valoraron aquellas HU con estimaciones de tiempo moderadas, para optimizar el uso del tiempo disponible en este sprint ya que se desea ir avanzando en lo posible las funcionalidades del próximo sprint 4.

#### c) Selección de Historias de Usuario para el Sprint 3

##### HU10 – Perfil de nutriólogos

Como nutriólogo, quiero tener acceso a un perfil de usuario para insertar información visible y crear mis rutinas y dietas para que más pacientes puedan visitarme y ponerse en contacto conmigo.

**Criterios de aceptación:**

## Escenario 1 – Perfil de nutriólogo

Cuando el nutriólogo entre a su perfil, podrá visualizar qué rutinas y dietas ha creado junto a un botón para crear otras, y una opción de editar perfil para añadir sus contactos, lugar en donde trabaja y descripción.

## HU11 – Perfil de pacientes

Como paciente, quiero tener acceso un perfil de usuario para visualizar mi actividad y gestionar mi información visible.

### Criterios de aceptación:

#### Escenario 1 – Perfil de paciente

Cuando el paciente ingrese su perfil, podrá visualizar que rutina o dieta ha tiene guardado, junto a una opción de editar perfil para añadir su descripción y gestionar la información solicitada en el cuestionario.

## HU13 – Ajustes de Usuario

Como usuario, quiero hacerles ajustes de seguridad a mí cuenta para no descuidarla en caso de que se me olvide la contraseña.

### Criterios de aceptación:

#### Escenario 1 – Ajustes de usuario

Cuando el usuario está en la opción de ajustes, tendrá una opción para actualizar contraseña, y también una para cerrar sesión.

## HU14 - Personalizar calendario de paciente

Como paciente, quiero gestionar un calendario personal para organizar mis rutinas de entrenamiento proporcionadas por el nutriólogo.

## Criterios de aceptación:

### Escenario 1

El paciente llega a una opción donde puede acceder a su calendario. El paciente tendrá acceso a un calendario donde podrá agregar sus rutinas de ejercicios para cada día de la semana. Podrá asignar los días de entrenamiento, seleccionar qué rutinas realizará cada día, y programar los días de descanso. Esto le permitirá visualizar y organizar su plan de entrenamiento semanal.

### d) Decisión de Eliminar HU15 - Feedback sobre Rutinas y Dietas

Inicialmente, se consideró incluir la HU15 – Feedback sobre Rutinas y Dietas, que permitiría a los pacientes calificar y comentar las rutinas y dietas seguidas. Sin embargo, tras una discusión entre el nutriólogo y el desarrollador, se decidió que esta HU no era crítica para el funcionamiento básico de la aplicación ni tampoco era tan necesaria con respecto a las demás HU.

#### HU15 – Feedback sobre rutinas y dietas (ELIMINADA)

Como paciente, quiero poder calificar y dejar comentarios sobre las rutinas y dietas que he seguido para proporcionar feedback útil al nutriólogo y a otros pacientes.

## Criterios de aceptación:

### Escenario 1 - Feedback sobre rutinas y dietas

Después de completar una rutina o dieta, el paciente podrá calificarla usando un sistema de estrellas o puntos y dejar un comentario opcional describiendo su experiencia. Esta información será visible tanto para el nutriólogo como para otros pacientes registrados, ayudando a mejorar las recomendaciones y la personalización de futuras rutinas y dietas.

## 7.2 Sprint Planning:

### AGENDA TRATADA DURANTE LA CEREMONIA DE SPRINT PLANNING:

#### a) Revisión de las Historias de Usuario del Sprint 3:

- Se discutió la prioridad de las HU a trabajar, destacando su relevancia para mejorar la personalización de perfiles y la gestión de calendarios.

- Se detallaron las tareas específicas para cada HU, asegurando que cada tarea requerirá más de 20 minutos.
- Se aclararon los criterios de aceptación para cada historia y se identificaron áreas donde se requerirá la colaboración de Jasmin para proporcionar la organización de los datos sobre rutinas y dietas en Excel, proporcionados por el nutriólogo.

**b) Asignación de responsabilidades:**

- Gonzalo Tafur Bermúdez fue asignado como responsable de las tareas técnicas y de programación de las nuevas funcionalidades.
- Jasmin colaborará proporcionando la organización de la información necesaria sobre rutinas y dietas, completando los archivos de Excel con los datos del nutriólogo para las HU correspondientes.
- Se distribuyeron las tareas respetando la capacidad de Gonzalo para programar la plataforma, mientras que Jasmin completará el Excel de rutinas y dietas en paralelo.

**c) Determinación de dependencias y predecesoras:**

- HU10 (Perfil de nutriólogos), HU11 (Perfil de pacientes) y HU11 (Perfil de pacientes) dependen de la HU02
- HU14 (Personalizar calendario de paciente) depende de HU05

**d) Aprobación del Sprint Backlog:**

- Se llegó a un acuerdo sobre la lista de tareas a realizar durante el Sprint 3, así como la duración y el responsable de cada tarea.
- Se priorizaron funcionalidades que sentarán las bases para mejorar la comunicación e interacción entre nutriólogos y pacientes en el próximo sprint.

**PROCEDIMIENTO SEGUIDO PARA LA CEREMONIA:**

- FECHA: La ceremonia de planificación del Sprint 3 se llevó a cabo el 12/10/2024.
- PARTICIPANTES: Incluyó a todo el equipo
- RESULTADOS DE LA CEREMONIA:
  - ✓ **Sprint Backlog:** Se seleccionan las HU y tareas del Sprint Backlog para el Sprint 2.
  - ✓ **Duración de las tareas:** Cada tarea fue estimada para respetar la capacidad de Gonzalo en la programación, teniendo en cuenta la disponibilidad de Jasmin para completar el Excel de rutinas y dietas.
  - ✓ **Predecesoras:** Las HU10, HU11 y HU13 dependen del registro y acceso de usuarios (cuestionario); HU14 necesita la carga previa de ejercicios y rutinas.
  - ✓ **Evidencia:** Se presentaron prototipos de los perfiles de usuario (pacientes y nutriólogos), junto con los datos organizados en Excel por Jasmin para implementar en la plataforma.

## PROTOTIPO PERFILES DE USUARIOS (Nutriólogos y Pacientes)

The screenshot shows a user profile interface. On the left, there is a sidebar with an 'Avatar' placeholder, the name 'Jane Doe', a 'Editar perfil' link, and a date '2006-05-17 Chile'. The main area is titled 'Información' and contains the following data:

- Frecuencia de ejercicios: 2 horas o más
- Alergia o intolerancia: Otro
- Peso corporal: 10
- Perímetro de cintura: 10
- Perímetro de cadera: 10
- Perímetro de muslo: 10
- Perímetro de brazo/bicep: 10

## PROTOTIPO CALENDARIO (mi horario)

The screenshot shows a calendar interface. At the top, it says 'Programa tus tiempos para hacer rutinas a tu gusto.' Below, there are two entries:

- Rutina de Volumen** (Tuesday)
  - Martes De 10:00 a las 11:00
  - [Eliminar de tu horario](#)
- Rutina de Volumen** (Friday)
  - Viernes De 10:00 a las 10:40
  - [Eliminar de tu horario](#)

At the bottom, there is a search bar: 'Rutina: Rutina de Volumen ▾ Día de la semana: Lunes ▾ Hora de Inicio: --:-- ⏱ Hora de Fin: --:-- ⏱ Guardar en mi horario'.

En un principio se tenía planeado colocar un horario de inicio y fin de la rutina, pero esto fue retirado y cambiado mejor a trabajar por períodos (mañana, medio día, tarde y noche)

## PROGRESO EXCEL DE RUTINAS Y DIETAS

VOLUMEN	NOTAS	Los dos primeros de la lista son el desayuno, luego sigue la 1ra merienda, le sigue el almuerzo, la 2da merienda y finalmente la Por error use 5 celdas para cada rutina, ya que son 5 comidas al dia, pero me olvide que en el desayuno son dos alimentos (lo		
DEFICIT	Nombre dieta	Objetivo	Alimentos	Descripción
	Energía Completa	VOLUMEN	Batido de Plátano y Avena / Pan con Paita y Huevo Batido de Proteínas con Frutas Arroz con Pollo Yogurt con Quinua Pollo a la Plancha con Camote	Esta dieta está diseñada para quienes buscan aumentar su masa muscular
			Batido de Plátano y Avena / Tortilla de Huevo con Queso	Esta dieta está pensada para quienes buscan ganar masa muscular de forma

	VOLUMEN SUPERIOR		DEFICIT SUPERIOR		Descripción
	VOLUMEN INFERIOR		DEFICIT INFERIOR		
Nombre rutina	Ejercicios	Tipo	Parte del cuerpo	Nivel	
Fuerza Superior 1	Press de banca	Volumen	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en fase de volumen.
	Remo con barra				
	Curl con barra				
Fuerza Superior 2	Press francés				
	Press inclinado	Volumen	Tren Superior	Principiante	Esta rutina se enfoca en construir masa muscular en el tren superior.
	Elevaciones laterales				
Fuerza Superior 2	Fondos en paralelas				
	Curl de bíceps				
	Elevaciones laterales				
Fuerza Inferior 1	Press de pecho en máquina	Volumen	Tren Superior	Principiante	En esta rutina, se utilizan máquinas y ejercicios con poco peso.
	Remo en máquina				
	Curl de bíceps				
Fuerza Inferior 2	Elevaciones laterales				
	Presa de piernas	Volumen	Tren Inferior	Principiante	Esta rutina está diseñada para fomentar el crecimiento muscular.
	Elevación de talones en prensa				
Fuerza Inferior 3	Zancadas caminando				
	Peso muerto con mancuernas ligeras				
	Sentadilla con barra	Volumen	Tren Inferior	Principiante	Esta rutina está pensada para principiantes que buscan aumentar la fuerza.
Fuerza Inferior 2	Hip thrust				
	Peso muerto rumano				
	Elevación de talones				
Fuerza Inferior 3	Peso muerto sumo	Volumen	Tren Inferior	Principiante	En esta rutina de volumen, los principiantes deben descansar durante las series.
	Curl de piernas en máquina				
	Sentadilla búlgara				
	Elevación de talones sentados				
	Press de banca con repeticiones altas	Deficit	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en déficit calórico.

Tareas/Tareas no técnicas	Estimación	Asignación de Trabajo
<b>HU10 – Perfil de nutriólogos</b>		
Crear interfaz diferenciada para perfil de nutriólogos	3h	Tafur Bermúdez, Gonzalo
Implementar opción para editar su perfil, permitiendo principalmente editar su nombre de usuario y biografía.	3h	Tafur Bermúdez, Gonzalo
Implementar sección para mostrar las dietas y rutinas creadas por el nutriólogo	1h	Tafur Bermúdez, Gonzalo
Implementar vista de información profesional y especialidades	2h	Tafur Bermúdez, Gonzalo

Registrar actividad dentro del perfil del nutriólogo	1h	Tafur Bermúdez, Gonzalo
Pruebas de usabilidad para la visualización del perfil del nutriólogo	3h	Tafur Bermúdez, Gonzalo

<b>HU11 – Perfil de pacientes</b>		
Crear interfaz diferenciada para perfil de pacientes	3h	Tafur Bermúdez, Gonzalo
Implementar opción para editar su perfil, permitiendo principalmente editar su nombre de usuario y biografía.	3h	Tafur Bermúdez, Gonzalo
Implementar sección para ver dietas seguidas, medidas y datos adicionales	5h	Tafur Bermúdez, Gonzalo
Implementar un botón para dejar de seguir una dieta	6h	Tafur Bermúdez, Gonzalo
Configurar visualización de las listas según objetivos nutricionales (volumen o déficit)	5h	Tafur Bermúdez, Gonzalo
Registrar actividad de datos actualizados por el usuario (perfil, información de paciente y dietas seguidas)	1h	Tafur Bermúdez, Gonzalo
Pruebas de experiencia de usuario para el perfil del paciente	3h	Tafur Bermúdez, Gonzalo

<b>HU13 – Ajustes de Usuario</b>		
Crear página de ajustes con opciones para cambiar las credenciales	2h	Tafur Bermúdez, Gonzalo
Registrar actividad de los cambios que hizo el usuario.	1h	Tafur Bermúdez, Gonzalo

Validar datos y realizar pruebas de seguridad	30min	Tafur Bermúdez, Gonzalo
Revisión de contenido y configuración de ajustes	30min	Tafur Bermúdez, Gonzalo

<b>HU14 – Personalizar calendario de paciente</b>		
Desarrollar la interfaz del calendario para agregar rutinas	3h	Tafur Bermúdez, Gonzalo
Implementar condicional para evitar agregar más de una rutina por día.	30min	Tafur Bermúdez, Gonzalo
Registro de actividad de la creación y eliminación de los horarios.	1h	Tafur Bermúdez, Gonzalo
Integrar filtro de rutinas según objetivo nutricional (volumen/deficit) debajo del calendario como ayuda	40min	Tafur Bermúdez, Gonzalo
Organizar datos de rutinas proporcionadas para su integración en el calendario	1h 30min	Villaca Mamani, Jasmin
Pruebas de funcionalidad y experiencia de usuario para el calendario	2h	Tafur Bermúdez, Gonzalo
Crear y organizar ejercicios adicionales en Excel para nuevas rutinas	1h	Villaca Mamani, Jasmin

Durante la ceremonia de Sprint Planning, el equipo aprobó las tareas necesarias para implementar estas funcionalidades esenciales. Se priorizó la creación de vistas personalizadas para nutriólogos y pacientes, con énfasis en que el nutriólogo pueda visualizar y gestionar la información del paciente. Además, se elabora la base de datos de ejercicios y rutinas en Excel para asegurar que la información sea accesible y para las futuras interacciones entre nutriólogo y paciente.

### KANBAN BOARD JIRA (Contraseña y correo para el acceso)

<https://jasmin.atlassian.net/jira/software/projects/KAN/boards/1>  
[proyecdietasyrutinas@gmail.com](mailto:proyecdietasyrutinas@gmail.com)

DyROtafvill

## 7.3 Daily Scrum:

Durante el Sprint 3, el equipo de desarrollo se centró en implementar funcionalidades clave para la visualización y gestión de los perfiles de los usuarios, tanto para nutriólogos como para pacientes, así como en la personalización de los calendarios. Las historias de usuario que se trabajaron fueron HU10 (Perfil de nutriólogos), HU11 (Perfil de pacientes), HU13 (Ajustes de usuario) y HU14 (Personalización del calendario de paciente). En las reuniones que tienen una duración de 40min a 1h, se discutieron avances, obstáculos y acciones a tomar para garantizar el cumplimiento de los criterios de aceptación establecidos.

### ¿QUÉ HICE AYER?

Ayer, el equipo avanzó en la implementación de las vistas personalizadas para los perfiles de nutriólogos (HU10) y pacientes (HU11). En el perfil del nutriólogo, se trabajó en la visualización de las rutinas y dietas creadas, junto con las opciones de editar perfil y agregar nuevas rutinas y dietas. Para los pacientes, se implementó la visualización de rutinas y dietas guardadas, así como las opciones de editar perfil y gestionar la información personal. Además, se comenzó a trabajar en la integración de los ajustes de seguridad (HU13), permitiendo la actualización de contraseñas y la opción de cerrar sesión.

### ¿QUÉ HARÉ HOY?

Hoy, se continuará con la implementación de la funcionalidad para personalizar el calendario de los pacientes (HU14). El objetivo es permitir que los pacientes gestionen y organicen sus rutinas de entrenamiento semanales, asignando días de actividad y descanso. Además, se terminará de pulir la parte de ajustes de usuario (HU13) y se realizará la conexión entre las rutinas y los calendarios para asegurar que las rutinas se muestren correctamente en los días correspondientes.

### ¿QUÉ IMPEDIMENTOS TENGO?

Un problema surgió con el manejo de condicionales en el código debido a la lógica compleja para diferenciar entre las vistas del perfil de nutriólogos y pacientes. El código tiene muchas condiciones (if/else) para mostrar solo la información relevante según el rol del usuario (nutriólogo o paciente), lo que está llevando a una saturación en la estructura del código. El programador sugirió que el uso de una solución como Spring podría ayudar a simplificar la lógica, ya que permitiría gestionar mejor las dependencias y condicionales, mejorando la escalabilidad y legibilidad del código.

### ¿CÓMO RESOLVI EL PROBLEMA?

El equipo discutió la posible implementación de Spring para simplificar la lógica condicional y mejorar la gestión de vistas y funcionalidades para nutriólogos y pacientes. Esta solución permitiría asignar dinámicamente las vistas según el rol del usuario, lo que reduciría la cantidad de condicionales y optimizaría el flujo del código. Sin embargo, por ahora, solo se está evaluando esta opción, y su implementación quedará pendiente en caso de que el equipo tenga tiempo disponible. Dado que actualmente podemos avanzar con la estructura existente, esta situación no afectará el progreso del sprint ni los entregables planificados.

## EVIDENCIA DE LA REUNIÓN #1 (VIERNES 25/10)

- DURACIÓN: 22:00 – 22:40 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

## Definición de funcionalidades para HU10 y HU11 – Perfiles de Nutriólogos y Pacientes:

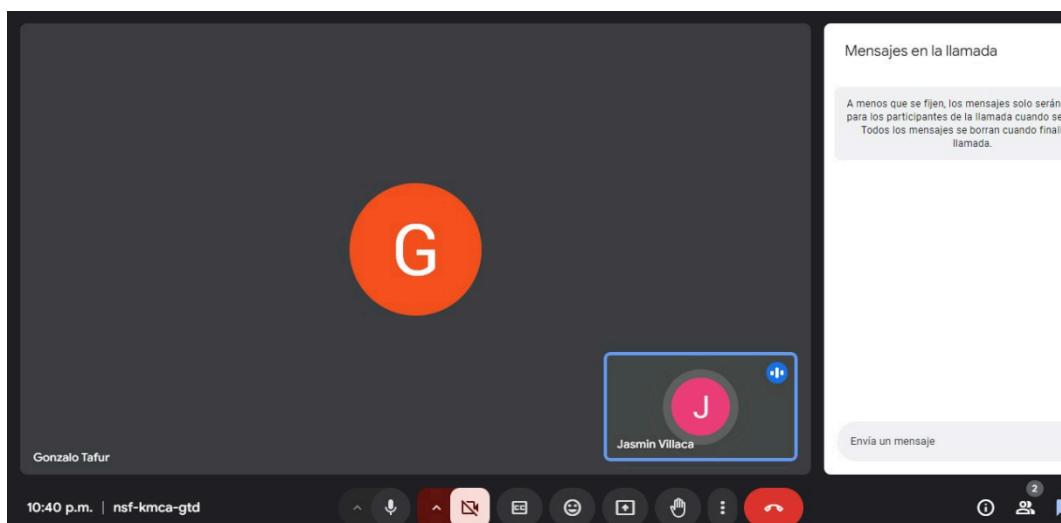
- ✓ Se discutió cómo se verán las vistas correspondientes al perfil de nutriólogos y pacientes, definiendo los elementos que estarán presentes en cada sección, como las opciones para editar perfil, visualizar rutinas/dietas creadas (para nutriólogos), y las dietas/rutinas seguidas por el paciente.
- ✓ Se estableció dónde se ubicarán estos componentes dentro de la interfaz para facilitar la navegación y mejorar la experiencia de usuario.

## Estructuración del Calendario – HU14:

- ✓ El equipo revisó la estructura del calendario para los pacientes, definiendo cómo se organizarán las rutinas diarias, con énfasis en asegurar que no se puedan duplicar rutinas en un mismo día.
- ✓ Se discutieron ideas sobre cómo implementar un filtrado basado en los objetivos del paciente (volumen o déficit), mostrando las rutinas recomendadas debajo del calendario.
- ✓ Se acordó que el calendario empezaría a desarrollarse de manera técnica en la segunda semana del sprint

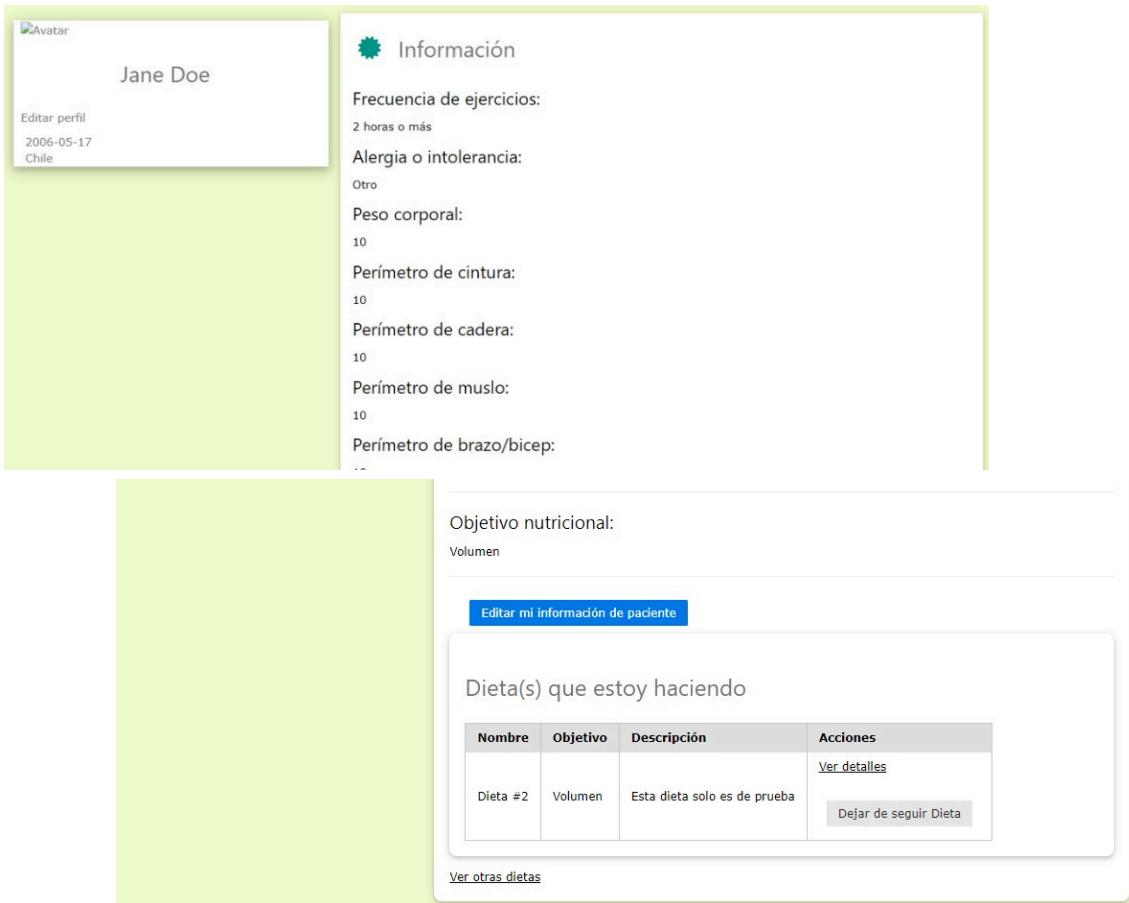
## Avances en la gestión de alimentos y ejercicios:

- ✓ Se presentaron los nuevos datos de alimentos y ejercicios añadidos a la base de datos, asegurando que los nutriólogos tengan acceso a una mayor variedad para personalizar rutinas y dietas.
- ✓ Se plantearon mejoras en la búsqueda y categorización para agilizar el acceso a esta información dentro de la plataforma.



## RESULTADOS DE LA REUNION

## 1) PROTOTIPO INICIAL DEL PERFIL DE USUARIOS



The screenshot shows a user profile page for "Jane Doe". On the left, there's a sidebar with an "Avatar" placeholder, the name "Jane Doe", and edit buttons for "Editar perfil", "2006-05-17", and "Chile". The main content area has a green header bar. Below it, there are two sections: "Información" and "Objetivo nutricional".

**Información**

- Frecuencia de ejercicios: 2 horas o más
- Alergia o intolerancia: Otro
- Peso corporal: 10
- Perímetro de cintura: 10
- Perímetro de cadera: 10
- Perímetro de muslo: 10
- Perímetro de brazo/bicep: 10

**Objetivo nutricional:**  
Volumen

[Editar mi información de paciente](#)

**Dieta(s) que estoy haciendo**

Nombre	Objetivo	Descripción	Acciones
Dieta #2	Volumen	Esta dieta solo es de prueba	<a href="#">Ver detalles</a> <a href="#">Dejar de seguir Dieta</a>

[Ver otras dietas](#)

## 2) EXCEL DE NUEVOS ALIMENTOS Y EJERCICIOS AGREGADOS

CATEGORÍA		ENFOQUE A VOLUMEN (no necesariamente) Enfocadas a DEFICIT (no necesariamente)		
	NUEVO			
	IDAlimento	ALIMENTO		DescAlimento
		Nutrientes	TipoAlimento	
1	Batido de Plátano y Avena	15g de proteína, 70g de cá	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía.
2	Pan con Palta y Huevo	20g de proteína, 50g de c.	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en carbohidratos.
3	Arroz con Pollo	30g de proteína, 80g de c.	Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias.
4	Lentejas con Carne	25g de proteína, 60g de c.	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. F
5	Papa Rellena con Carne	22g de proteína, 50g de c.	Cena	Papas rellenas con carne molida de res (100g), con cebolla, ajo, y especias.
6	Tallarines Verdes con Pollo	28g de proteína, 90g de c.	Cena	Pasta (200g) con pesto de albahaca, espinacas y pollo a la plancha (150g).
7	Yogurt con Quinua	12g de proteína, 40g de cá	Merienda	Yogurt natural (200 ml) con quinua cocida (50g). Ideal para complementar la dieta.
8	Batido de Proteínas con Frutas	25g de proteína, 50g de cá	Merienda	Pollo de proteína (30g) mezclado con leche (200 ml), plátano y fresas.
9	Batido de Frutas y Chía	10g de proteína, 45g de cá	Desayuno	Batido de plátano, fresas y chía (1 cdca), mezclado con agua o leche vegetal.
10	Tostadas Integrales con Palta y Tomate	7g de proteína, 30g de cál	Desayuno	Tostadas de pan integral (2 rebanadas), con palta (30g) y tomate. Bajo en calorías.
11	Ensalada de Pollo y Quinua	25g de proteína, 45g de cá	Almuerzo	Quinua (100g) con pollo a la plancha (150g), espinaca, tomate, y palta. F
12	Pescado a la Plancha con Camote	30g de proteína, 40g de cá	Almuerzo	Filete de pescado (150g) a la plancha, acompañado de camote sancochado.
13	Sopa de Verduras con Pollo	15g de proteína, 30g de cá	Cena	Sopa a base de verduras (zapallo, zanahoria, espinaca) y pechuga de pollo.
14	Tortilla de Claras con Espinacas	12g de proteína, 10g de cá	Cena	Tortilla hecha con claras de huevo (4) y espinacas. Ideal para una cena ligera.
15	Yogurt Descremado con Almendra	8g de proteína, 20g de cál	Merienda	Yogurt descremado (200g) con almendras (15g). Aporta saciedad sin azúcar.
16	Gelatina de Fresa con Chía	5g de proteína, 10g de car	Merienda	Gelatina baja en calorías (1 taza) con una cucharada de chía. Refrescante.
17	Panqueques de Avena y Plátano	15 g de proteína, 30 g de cá	Desayuno	Panqueques elaborados con 80 g de avena, 1 plátano mediano (120 g).
18	Batido de yogur y frutas	12 g de proteína, 25 g de cá	Desayuno	Batido de 200 g de yogur natural con 100 g de fresas, 1 plátano pequeño.
19	Tostadas Integrales con Palta y Huevo	18 g de proteína, 22 g de cá	Desayuno	Dos rebanadas de pan integral (80 g) tostadas, cubiertas con una pura de quinua cocida.
20	Quinua con Leche y Frutos Secas	10 g de proteína, 35 g de cá	Desayuno	100 g de quinua cocida en 200 ml de leche, enriquecida con 30 g de almendras.
21	Licuado de Quinua y Plátano	12 g de proteína, 35 g de cá	Desayuno	Licuado de 100 g de quinua cocida, 1 plátano y 200 ml de leche de soja.
22	Batido de maíz y plátano	12 g de proteína, 35 g de cá	Desayuno	Batido preparado con 1 plátano mediano, 1 cucharada de maíz en polvo.
23	Tortilla de Huevo con Queso Fresco	15 g de proteína, 8 g de cá	Desayuno	Tortilla hecha con 3 huevos, 50 g de queso fresco y 100 g de espinacas.
24	Panqueques de Avena con Mantequilla	20 g de proteína, 40 g de cá	Desayuno	Panqueques hechos con 100 g de avena, 1 huevo y 30 g de mantequilla.
25	Tostadas Francesas con Frutas y Mantequilla	18 g de proteína, 50 g de cá	Desayuno	2 rebanadas de pan integral empanadas en mezcla de huevo y canela, s
26	Tostadas Integrales con Mantequilla	14 g de proteína, 40 g de cá	Desayuno	Dos rebanadas de pan integral con 30 g de mantequilla de maní y 1 plátano.
27	Lomo Saltado con Arroz y Papas	30 g de proteína, 45 g de cál	Almuerzo	50 g de carne res salteada con 100 g de cebolla y 100 g de tomate, servido con arroz y papas.
28	Estofado de Pollo con Quinua	25 g de proteína, 40 g de cál	Almuerzo	150 g de pollo cocido lentamente con 150 g de verduras variadas y 100 g de quinua.
29	Tallarines verdes con bistec	28 g de proteína, 50 g de cál	Almuerzo	150 g de pasta con salsa de albahaca y espinaca, acompañado de 150 g de bistec.
30	Pescado a la Plancha con Tacu-Tacu	28 g de proteína, 40 g de cál	Almuerzo	200 g de filete de pescado a la plancha con 150 g de tacu-tacu de lentejas.
31	Tortilla de espinacas, queso y quinua	22 g de proteína, 20 g de cá	Cena	Tortilla preparada con 150 g de espinacas frescas, 100 g de queso bajo en grasa.
32	Pollo a la Plancha con Camote	30 g de proteína, 40 g de cá	Cena	200 g de pollo a la plancha acompañado de 200 g de camote asado y 100 g de quinua.
33	Ensalada de Menestras con Atún	22 g de proteína, 20 g de cá	Cena	Mezcla de 100 g de lentejas y 100 g de garbanzos con 150 g de atún en aceite de oliva.

ALIMENTOS

DIETAS

EJERCICIOS

RUTINAS



ANTIGUO		VOLUMEN (no necesariamente) DEFICIT (no necesariamente)			
	NUEVOS				
IDEjercicio	NomEjercicio	EJERCICIO		DescEjercicio	
		IDejercicio	GrupMuscular		
1	Press de banca	Pecho	Tren Superior	3 a 8	Acuéstate en un banco plano con los pies en el suelo. Baja la barra hasta el pecho.
2	Press inclinado	Pecho	Tren Superior	3 a 8	En un banco inclinado, baje la barra o mancuernas hacia el pecho superior.
3	Remo con barra	Espalda	Tren Superior	3 a 8	Inclina el torso hacia adelante con la barra en las manos. Sube la barra hacia arriba.
4	Curl con barra	Biceps	Tren Superior	3 a 8	Sostén la barra con las palmas hacia arriba. Flexiona los codos elevando la barra.
5	Press francés	Tríceps	Tren Superior	3 a 8	Acuéstate en un banco, sujetá una barra 2. Flexiona los codos bajando la barra.
6	Extensión de tríceps en po	Tríceps	Tren Superior	3 a 8	Sujeta la cuerda o barra en una polea alta. Empuja hacia abajo hasta estirar los tríceps.
7	Elevaciones laterales	Hombros	Tren Superior	3 a 8	De pie, con una mancuerna en cada mano, eleva los brazos hacia los lados.
8	Sentadilla con barra	Cuádriceps	Tren Inferior	3 a 8	Con la barra sobre los hombros, baje flexionando las rodillas hasta que los cuádriceps toquen el suelo.
9	Peso muerto rumano	Femorales	Tren Inferior	3 a 8	Baja la barra con las piernas ligeramente flexionadas y la espalda recta, hasta que las rodillas toquen el suelo.
10	Hip thrust	Glúteos	Tren Inferior	3 a 8	Apoya la parte superior de la espalda en un banco y coloca una barra sobre las nalgas.
11	Press de banca con repetición	Pecho	Tren Superior	4 a 15	Similar al press de banca regular, pero enfocado en más repeticiones (12-15).
12	Remo con mancuerna	Espalda	Tren Superior	4 a 15	Con una mancuerna en una mano y apoyando la otra en un banco, lleva la barra hacia arriba.
13	Curl con mancuernas ligeras	Biceps	Tren Superior	4 a 15	Con las palmas hacia arriba, flexiona los codos levantando las mancuernas.
14	Fondos en paralelas	Tríceps	Tren Superior	4 a 15	Con las manos en las barras paralelas, baje el cuerpo flexionando los codos.
15	Elevaciones laterales con Hombros	Hombros	Tren Superior	4 a 15	Eleva los brazos hacia los lados con mancuernas ligeras hasta que estén paralelos.
16	Zancadas caminando	Cuádriceps	Tren Inferior	4 a 15	Da un paso largo hacia adelante y baje flexionando ambas rodillas. Alterna las piernas.
17	Curl de piernas repetición	Femorales	Tren Inferior	4 a 15	Acuéstate en la máquina de curl de piernas y flexiona las rodillas llevando las piernas hacia el pecho.
18	Hip thrust repeticiones altas	Glúteos	Tren Inferior	4 a 15	Eleva las caderas contrayendo los glúteos, pero con un peso moderado y controlado.
19	Elevaciones de talones	Pantorrillas	Tren Inferior	4 a 15	Sube y baje los talones manteniendo el peso sobre la parte delantera de los talones.
20	Prensa de Banca con Manos	Pecho	Tren Superior	3 a 10	Recuéstate en un banco, sostén una mancuerna en cada mano, baje hasta el pecho.
21	Prensa de Banca Declinada Pecho	Pecho	Tren Superior	3 a 10	Recuéstate en un banco inclinado, sujetá la barra con un agarre ancho, baje hasta el pecho.
22	Remo en máquina con pesas	Espalda	Tren Superior	3 a 10	Siéntate en la máquina de remo, sujetá las manijas y jala hacia la cintura.
23	Remo con Mancuernas en Espalda	Espalda	Tren Superior	3 a 10	Acuéstate boca abajo en un banco inclinado con mancuernas en las manos.
24	Prensa de Piernas	Piernas	Tren Inferior	3 a 10	Siéntate en la máquina de remo, sujetá las manijas y jala hacia la cintura.
25	Peso Muerto Sumo	Piernas	Tren Inferior	3 a 10	Con las piernas separadas y la barra entre ellas, baje y sube con la espalda recta.
26	Curl Concentrado con Manos	Biceps	Tren Superior	3 a 10	Sentado, apoya el codo en el muslo, eleva la mancuerna hacia el hombro.
27	Curl de bíceps en Predicador	Biceps	Tren Superior	3 a 10	Sentado en banco predicador, sujetá la barra con agarre supino, sube hasta que las mancuernas toquen el pecho.
28	Press de Tríceps con Manos	Tríceps	Tren Superior	3 a 10	Recuéstate en un banco, con las mancuernas hacia arriba, baje hacia la cara.
29	Dips entre bancos	Tríceps	Tren Superior	3 a 10	Coloca las manos en un banco detrás de ti, baje hasta que los codos estén estirados.
30	Prensa Militar con Mancuernas	Hombros	Tren Superior	3 a 10	Siéntate, sujetá una mancuerna en cada mano, presiona hacia arriba hasta que las mancuernas toquen el pecho.
31	Patada de glúteos en máquina	Glúteos	Tren Inferior	3 a 10	Coloque una pierna en la máquina, empujela hacia atrás y hacia arriba, regresa a la posición inicial.
32	Sentadilla Búlgara	Glúteos	Tren Inferior	3 a 10	Apoja un pie en un banco detrás de ti, baje en sentadilla con la pierna delantera.
33	Elevación de Talones Sentado	Pantorrillas	Tren Inferior	4 a 12	Siéntate en la máquina de pantorrillas, coloca el peso sobre las rodillas, eleva las piernas y baje.
34	Elevación de Talones en F	Pantorrillas	Tren Inferior	4 a 12	En la prensa, coloque los pies en la parte baja de la plataforma y empuje hacia arriba.
35	Elevación de Caderas	Caderas	Tren Inferior	4 a 12	En la prensa, coloque los pies en la parte alta de la plataforma y empuje hacia arriba.

ALIMENTOS

DIETAS

EJERCICIOS

RUTINAS



## EVIDENCIA DE LA REUNIÓN #2 (DOMINGO 03/11)

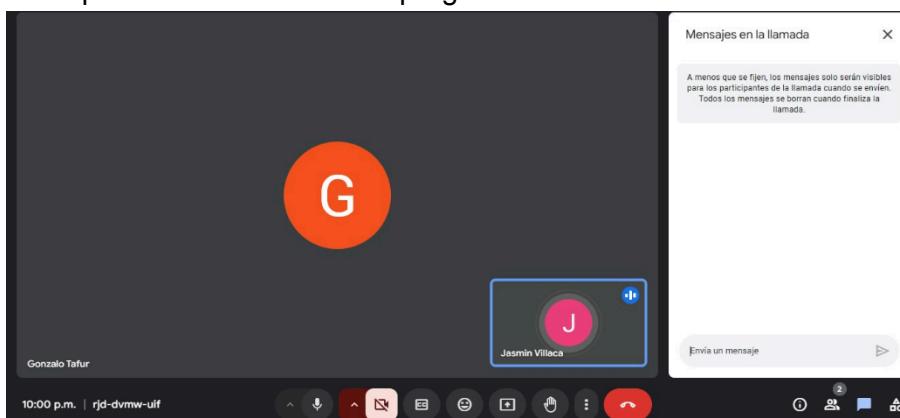
- DURACIÓN: 21:30 – 10:10 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Progreso en HU13 – Ajustes de Usuario:

- ✓ Se presentó el avance en la funcionalidad de ajustes para usuarios, permitiendo cambiar la contraseña y cerrar sesión. Esta funcionalidad fue validada por el equipo y aprobada para su implementación final.

### Revisión del prototipo de calendario – HU14:

- ✓ Se mostró el prototipo inicial del calendario para pacientes. Durante la revisión, se sugirieron ajustes importantes, como la visualización de rutinas recomendadas debajo del calendario para facilitar la planificación de entrenamientos.
- ✓ Se decidió modificar el esquema de organización de las rutinas, cambiando el enfoque de horarios específicos a períodos (mañana, tarde, mediodía, noche) para mayor flexibilidad.
- ✓ También se implementará una restricción que impide añadir más de una rutina en un mismo día para evitar conflictos de programación.



## RESULTADOS DE LA REUNION

### 1) AJUSTES DE USUARIO

The screenshot shows a web form titled "Cambiar contraseña". It has two input fields: "Nueva contraseña" and "Confirmar contraseña", both with placeholder text. Below the fields is a "Guardar" button. At the bottom, there is a question "¿Quieres cerrar sesión?" with a "Cerrar Sesión" link. The page is from the URL "localhost:8080/DietasYRutinasOnline/usuario/ajustesCuenta".

## 2) PRIMER PROTOTIPO DEL CALENDARIO

Programa tus tiempos para hacer rutinas a tu gusto.

Rutina de Volumen

Martes De 10:00 a las 11:00  
[Eliminar de tu horario](#)

Viernes De 10:00 a las 10:40  
[Eliminar de tu horario](#)

Rutina:  Día de la semana:  Hora de Inicio:  Hora de Fin:   Guardar en mi horario

## 3) SEGUNDO PROTOTIPO DEL CALENDARIO

Sabado Periodo: Tarde  
[Eliminar de tu horario](#)

Fuerza Superior  
[Ver detalles](#)

Viernes Periodo: Tarde  
[Eliminar de tu horario](#)

Inserte un nuevo horario

Rutina:  Día de la semana:  Periodo de día:   Guardar en mi horario

## EVIDENCIA DE LA REUNIÓN #3 (DOMINGO 10/11)

- DURACIÓN: 21:30 – 10:30 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Presentación final del calendario personalizado:

- ✓ Se presentó el calendario final con todas las funcionalidades acordadas en reuniones anteriores, como la gestión de rutinas por períodos del día (mañana, tarde, noche) y la opción de visualizar rutinas preseleccionadas debajo del calendario.

- ✓ Se implementó un filtrado avanzado según el objetivo del paciente (déficit calórico, volumen) y su nivel (principiante, intermedio, avanzado), permitiendo a los pacientes ver recomendaciones de rutinas específicas para sus necesidades al planificar su semana.

### Filtrado avanzado basado en datos del cuestionario:

- ✓ Se mostró el progreso en el filtrado de rutinas y dietas según las respuestas del cuestionario inicial del paciente. Ahora, si un paciente indicó un objetivo específico, como "volumen" o "déficit", o un nivel de entrenamiento, el sistema le mostrará automáticamente las opciones de acuerdo al cuestionario respondido.
- ✓ Las rutinas también aparecerán en un listado debajo del calendario, con detalles adicionales para que el paciente pueda elegir qué añadir a su calendario semanal.

### Avance en la implementación de transaccionalidades:

- ✓ Se discutió la estructuración de las próximas historias de usuario (HUs) para garantizar que las funcionalidades clave utilicen transacciones, mejorando la robustez del sistema.
- ✓ El equipo acordó priorizar la optimización del filtrado según las respuestas del cuestionario y finalizar la gestión transaccional antes de avanzar con las últimas historias de usuario planificadas.

### Problema identificado y solución propuesta:

- ✓ Se identificó un problema relacionado con la saturación de código debido a múltiples condicionales para manejar las vistas y roles de usuarios (nutriólogo y paciente). El equipo discutió la posible implementación de una dependencia de Spring para simplificar esta lógica en caso se disponga de tiempo adicional. Esto incluiría el uso de Spring Security para gestionar sesiones, roles y posiblemente integrarlo con la lógica transaccional.
- ✓ El programador mencionó haber investigado una dependencia de seguridad en Spring, la cual podría optimizar el manejo de roles y autenticación, así como mejorar el control de sesiones y la seguridad general del sistema. Sin embargo, se está evaluando si es factible integrarlo dentro del plazo actual, ya que requiere tiempo adicional para su configuración.

## RESULTADOS DE LA REUNION

### 1) CALENDARIO FINAL

Jueves

Periodo: Medio día

Eliminar de tu horario

Definición Superior 1

ver detalles

Lunes

Periodo: Mañana

Eliminar de tu horario

Inserte un nuevo horario

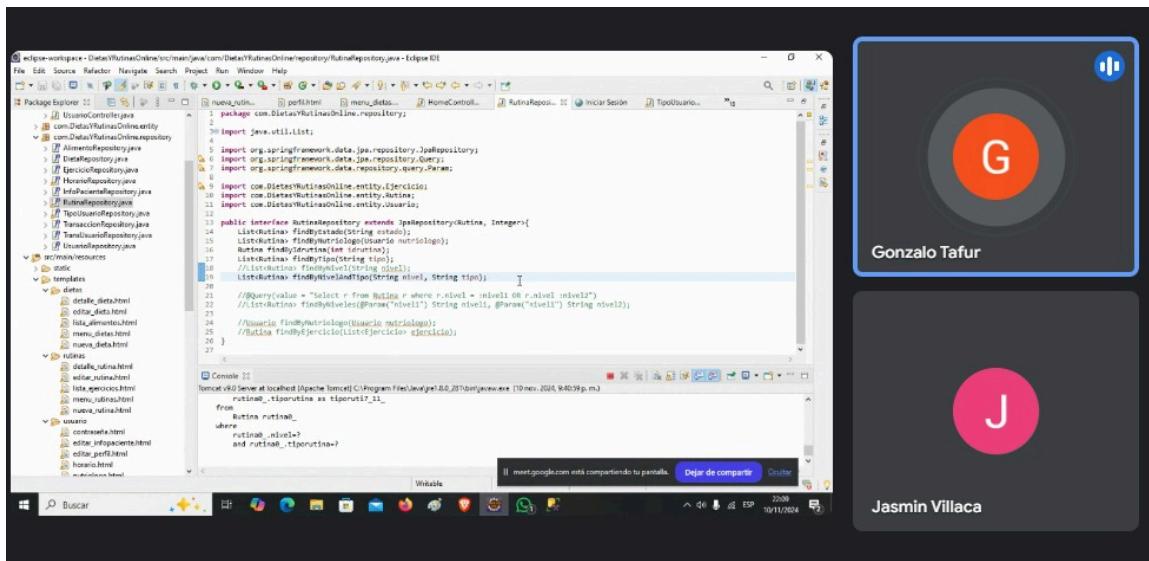
Rutina: Definición Superior 1 | Día de la semana: Lunes | Período de día: Mañana | Guardar nuevo horario

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Definición Superior 1	Deficit	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en déficit calórico. Comienza con el press de banca y realiza repeticiones altas (8-15). Se recomienda un descanso de 2 minutos entre series. La carga de peso debe ser liviana y enfocarse en un mayor número de repeticiones para promover la fuerza muscular. Se recomienda realizar esta rutina en un espacio amplio y seguro. Mantener la respiración controlada y mantener la postura correcta durante todo el ejercicio.	Ver detalles

## 2) TRANSACCIONALIDAD PRUEBAS

Results		Messages								
		idtransaccion	fechatrans	tipotrans	idusuario	idrutina	iddieta	idhorario	idinfopaciente	
1	1	1	2024-11-08 22:48:24.740	CREACIÓN	1	NULL	NULL	1	NULL	
2	2	2	2024-11-08 22:49:54.413	CREACIÓN	1	NULL	NULL	2	NULL	
3	3	3	2024-11-08 22:50:07.730	Logout	1	NULL	NULL	NULL	NULL	
4	4	4	2024-11-08 22:55:32.607	CREACIÓN	5	NULL	NULL	3	NULL	
5	5	5	2024-11-08 23:34:26.987	CREACIÓN	1	NULL	NULL	4	NULL	
6	6	6	2024-11-08 23:45:07.777	Logout	1	NULL	NULL	NULL	NULL	
		idtransaccion	idusuario	idtipousu	fechalogin	fechalogout	cambioPassword	cambioCorreo	fecharegistro	
8	8	8	1	1	2024-11-08 23:16:03.037	NULL	NULL	NULL	NULL	
9	9	9	1	1	2024-11-08 23:34:03.187	NULL	NULL	NULL	NULL	
10	10	10	1	1	2024-11-08 23:36:08.860	NULL	NULL	NULL	NULL	
11	11	11	1	1	2024-11-08 23:40:52.680	NULL	NULL	NULL	NULL	
12	12	12	1	1	NULL	2024-11-0...	NULL	NULL	NULL	
13	13	13	1	1	2024-11-09 23:21:31.127	NULL	NULL	NULL	NULL	
14	14	14	1	1	2024-11-10 00:12:39.373	NULL	NULL	NULL	NULL	
15	15	15	1	1	2024-11-10 00:15:42.687	NULL	NULL	NULL	NULL	
16	16	16	1	1	NULL	2024-11-1...	NULL	NULL	NULL	
17	17	17	5	1	2024-11-10 00:16:17.357	NULL	NULL	NULL	NULL	

## 3) LIGERO INCONVENIENTE CON EL CODIGO

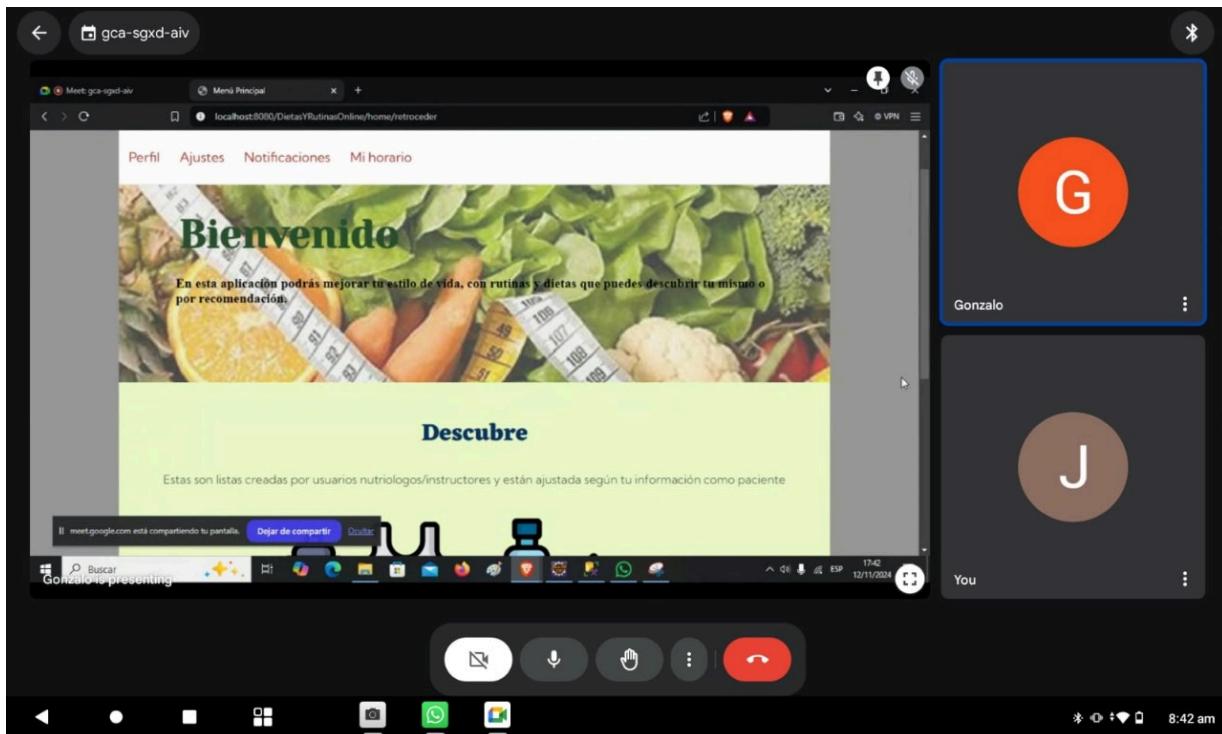


## EVIDENCIA DE LA REUNIÓN #4 (MARTES 12/11)

- DURACIÓN: 17:00 – 17:43 horas
- PARTICIPANTES: Tafur Bermúdez Gonzalo, Nutriólogo
- TEMAS TRATADOS:

✓ Se mostró al nutriólogo el avance completo de las funcionalidades hasta la fecha.

- ✓ Feedback para mejorar las condiciones alimenticias en pacientes y optimizar la creación de dietas, sugiriendo más filtros o detalles escritos.
- ✓ Se discutió la inclusión de un nuevo objetivo enfocado en "mejorar hábitos" además de los existentes (déficit y volumen).
- ✓ Revisión sobre cómo simplificar el registro de medidas para pacientes, incluyendo guías para el uso del IMC.



(El horario mostrado en la captura es debido a que el nutriólogo (J) se encuentra fuera del país)

## 7.4 Sprint Review:

Se presentará un resumen del incremento del producto completado durante este tercer sprint.

### ALCANCE DEL SPRINT

- Desarrollo del perfil de nutriólogo.
- Desarrollo del perfil de paciente.
- Implementación de ajustes de usuario (cambio de contraseña y cierre de sesión).
- Implementación de calendario personalizado para pacientes.
- Visualización de rutinas y dietas personalizadas según los objetivos y datos llenados en el cuestionario de los pacientes.

### RESULTADOS DEL SPRINT

## ○ PERFIL DE NUTRÍLOGOS

El nutriólogo ahora tiene acceso a su perfil donde puede ver todas las rutinas y dietas que ha creado. Además, se ha implementado un botón para crear nuevas rutinas y dietas. También se incluye una opción para editar su perfil, donde puede añadir detalles como la descripción personal, su lugar de trabajo, y sus datos de contacto. Esta funcionalidad permite al nutriólogo gestionar su presencia en la plataforma y hacerla más accesible a los pacientes.

The screenshot shows the Nutrólogo's profile page with the following details:

- Perfil:** Tester Nutriologo
- Biografía:** Sin biografía
- Opciones:** Editar Perfil, Fecha de nacimiento: 1998-04-18, Nacionalidad: Peru, Sexo: Masculino, Me uní en: [redacted]
- Rutinas creadas:**

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Fuerza Inferior 2	Deficit	Tren Inferior	Intermedio	Esta rutina está pensada para principiantes que buscan aumentar su masa muscular en el tren inferior. Comienza con la sentadilla con barra para activar los cuádriceps. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice un peso ligero y concentrado en la técnica, especialmente en la sentadilla y el peso muerto. Estos ejercicios son fundamentales para el desarrollo de glúteos y piernas fuertes, así que asegúrese de realizar cada movimiento con control.	<a href="#">Ver detalles</a>
Fuerza Inferior 3	Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Fuerza Inferior 3	Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Rutina de prueba 1	Volumen	Tren Superior	Intermedio	Esta rutina es de prueba.	<a href="#">Ver detalles</a>
- Categoría:** Crear Nueva rutina
- Rutinas adicionales:**

Fuerza Inferior 3	Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Rutina de prueba 1	Volumen	Tren Superior	Intermedio	Esta rutina es de prueba.	<a href="#">Ver detalles</a>
- Dietas creadas:**

Nombre	Objetivo	Descripción	Acciones
Dieta de déficit	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
Dieta #1	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
Dieta #2	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
Dieta #2	Volumen	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
Dieta #3	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
- Categoría:** Crear Nueva Dieta

## ○ PERFIL DE PACIENTES

Los pacientes ahora tienen su propio perfil donde pueden ver las rutinas y dietas que han seguido. Se les permite editar su perfil para incluir una descripción personal y actualizar la información relacionada con el cuestionario de salud. Esta funcionalidad brinda al paciente un mayor control sobre su información y le permite gestionar sus datos dentro de la plataforma.

The screenshot shows the Patient's profile page with the following details:

- Perfil:** Gonzalo Tafur Bermúdez
- Biografía:** Hola, esta es mi biografía
- Opciones:** Editar Perfil, Fecha de nacimiento: 2003-10-28, Nacionalidad: Otro, Sexo: Masculino, Me uní en: [redacted]
- Información:**
  - Dieta(s) que estoy haciendo:**

Nombre	Objetivo	Descripción	Acciones
Dieta #2	Volumen	Esta dieta solo es de prueba	<a href="#">Ver detalles</a>
  - Opciones:** Dejar de seguir Dieta, Seguir otras dietas
- Frecuencia de ejercicios:** 1 hora a diario
- Condición alimenticia:** Nada

## ○ AJUSTES DE USUARIO

Se ha implementado la opción de ajustes de usuario, donde los usuarios pueden cambiar su contraseña si la olvidan y también tienen la opción de cerrar sesión. Esto proporciona un nivel adicional de seguridad y control sobre las cuentas de usuario.

## ○ PERSONALIZAR CALENDARIO DE PACIENTE

Los pacientes ahora pueden gestionar un calendario personal donde pueden agregar las rutinas de ejercicios proporcionadas por el nutriólogo. El calendario permite asignar días específicos de entrenamiento, seleccionar las rutinas para esos días y programar los días de descanso. Esta funcionalidad facilita que los pacientes sigan sus programas de entrenamiento y se organicen de manera eficiente.

## OBSERVACIONES DEL PRODUCT OWNER (PROFESOR)

PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo, Marco Aurelio Espinoza Rivera (Profesor)

El profesor mencionó que el avance en cuanto a transacciones es algo que aún no hemos logrado implementar adecuadamente. Aunque entendemos las dificultades que conlleva el desarrollo de la aplicación con solo dos miembros en el equipo, reconocemos que debemos mejorar en este aspecto. Nos instó a apresurarnos con los avances para cumplir con los plazos establecidos y garantizar que la aplicación esté lo más funcional posible en el próximo sprint.

Además, el profesor indicó que no es recomendable permitir que los nutriólogos o pacientes cambien su nombre de usuario, ya que este es un dato fundamental y cambiarlo podría ocasionar problemas a largo plazo. Se nos recordó que debemos adherirnos a buenas prácticas en la gestión de cuentas de usuario, y evitar la modificación de estos datos.

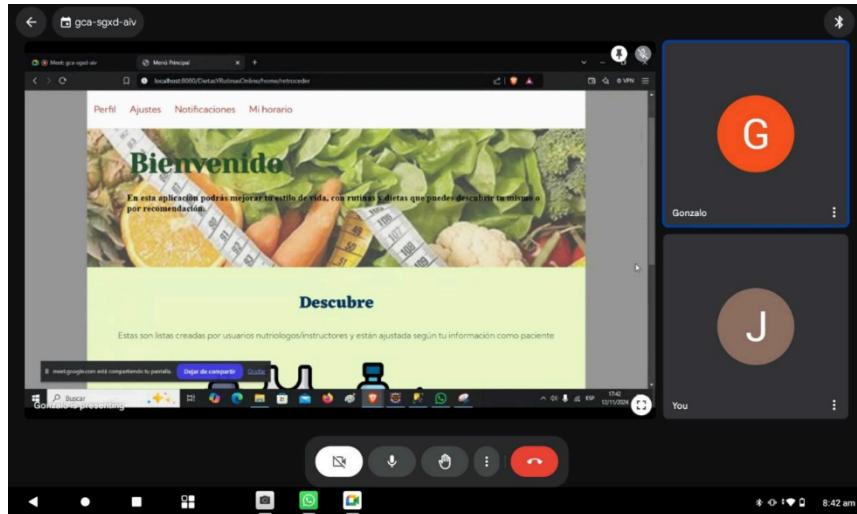
## EVIDENCIA DE LA REUNION REALIZADA EL DIA 12/11/24 (fin de la reunión 20:20)

The screenshot shows a web application interface. On the left, there is a sidebar for 'Tester Nutriologo' with fields for 'Nombre de la Rutina' (Fuerza Inferior 2), 'Tipo' (Deficit), 'Parte del cuerpo' (Tren Inferior), 'Nivel' (Intermedio), and 'Descripción'. The main area displays a table of exercises:

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Fuerza Inferior 2	Deficit	Tren Inferior	Intermedio	Esta rutina está pensada para principiantes que buscan aumentar su masa muscular en el tren inferior. Comienza con la sentadilla con barra para activar los cuádriceps. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice un peso ligero y concentrado en la técnica, especialmente en la sentadilla y el peso muerto. Estos ejercicios son fundamentales para el desarrollo de glúteos y piernas fuertes, así que asegúrese de realizar cada movimiento con control.	<a href="#">Ver detalles</a>
Fuerza Inferior 3	Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Fuerza Inferior 3	Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Rutina de Volumen	Volumen	Tren Superior	Intermedio	Esta rutina es de prueba.	<a href="#">Ver detalles</a>

## SUGERENCIAS DEL NUTRÍÓLOGO (REUNIÓN MARTES 12/11)

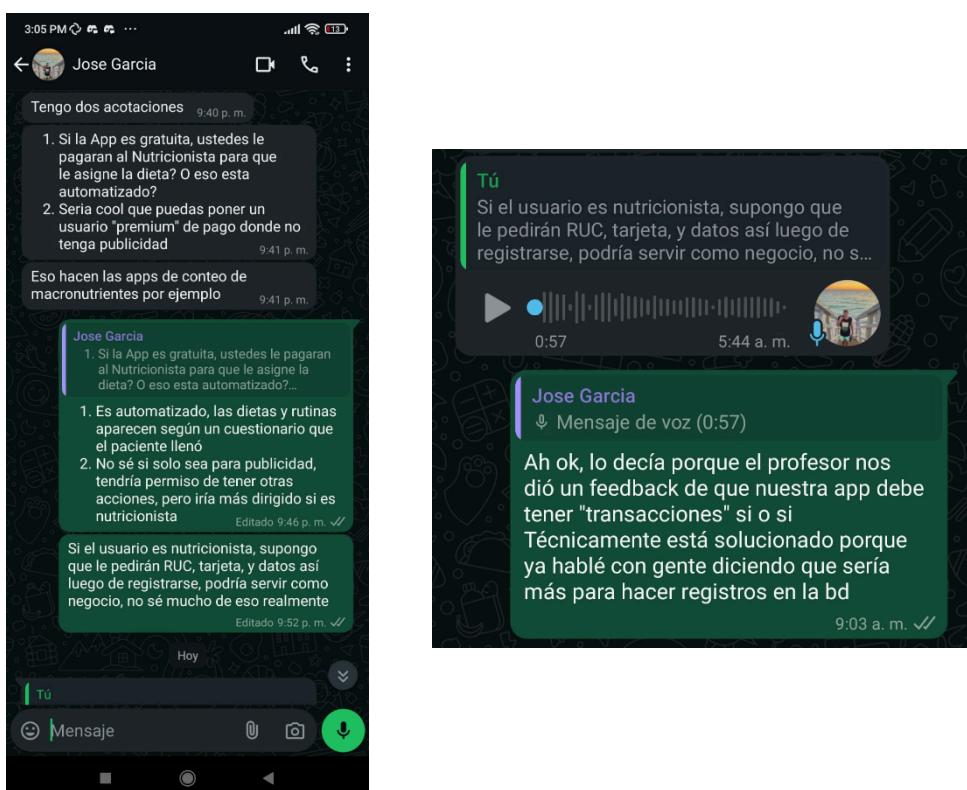
- ✓ Condiciones alimenticias y filtros en dietas: Añadir más condiciones alimenticias para personalizar dietas y permitir filtros escritos al crear dietas, en lugar de mostrarlos visualmente.
- ✓ Recomendaciones para registro de medidas: Incluir guías claras para que los pacientes registren correctamente sus medidas, considerando que el IMC puede ser confuso.
- ✓ Objetivo "mejorar hábitos": Agregar un objetivo para aquellos pacientes que busquen mejorar su salud y hábitos sin enfocarse en déficit o volumen.



## SUSTENTACIÓN DE LA APP WEB:

El nutriólogo también comentó que la forma de mantener la aplicación web es bastante sencilla. Los datos más importantes para él son las credenciales de acceso (usuario y contraseña), ya que no se manejarán transacciones monetarias ni operaciones complicadas. La plataforma servirá principalmente como una herramienta para gestionar a los pacientes, lo cual es suficiente para las necesidades actuales del nutriólogo.

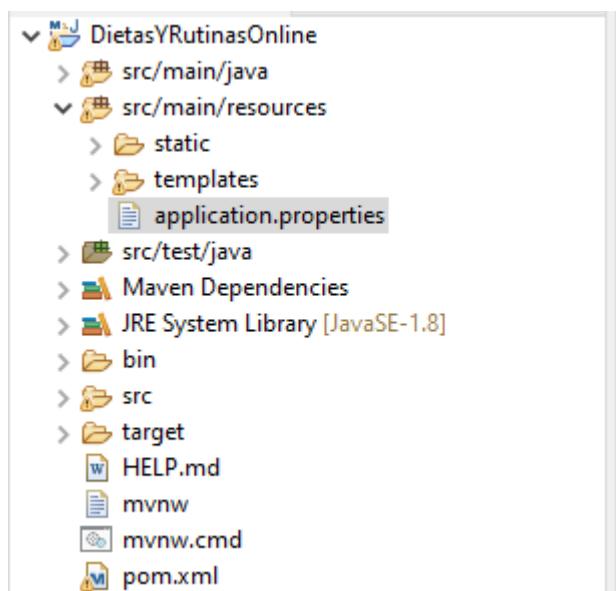
*"Los datos principales que necesito como nutricionista son solo el nombre de usuario y la contraseña (credenciales). Al final, voy a utilizar la plataforma como una herramienta para gestionar a mis pacientes, sin necesidad de realizar transacciones de dinero, cobros o planes. No te preocupes por esos detalles, con que me registre y pueda manejar a mis pacientes dentro del software está bien."*



## ACCESO A LA PLATAFORMA DE DESARROLLO

- **Acceso:** <https://github.com/GonzaloTafur/ProyectoTecnologicoGrupo3.git>

1. Crear una carpeta y extraer ahí el proyecto descargado en el GitHub. A dentro también hay un Word de manual de instrucciones como respaldo.
2. Exportar la carpeta “bd” que está dentro del repositorio, que incluye el query para la creación de la base de datos y las tablas.
3. Cambiar la ruta del application.properties con el nombre del servidor del equipo que haga la prueba.



4. Ejecutar el proyecto

En Eclipse: Click derecho al proyecto importado -> Run As -> 1 Run on Server -> Finish

## 7.5 Sprint Retrospective:

**Fecha y Hora:** 13 nov 2024 10:00 am - 10:40am

**Participantes:**

- ✓ Product owner
- ✓ Equipo de desarrollo

## PROCEDIMIENTO PARA LA RETROSPECTIVA DEL SPRINT 3

En la retrospectiva del Sprint 3, se discutieron varios puntos clave sobre el desarrollo de la aplicación y los ajustes necesarios para mejorar la experiencia tanto de los pacientes como de los nutricionistas. Durante la sesión, se revisaron las observaciones del Product Owner (profesor) y las sugerencias del nutricionista.

A continuación, se presentan las observaciones del Product Owner y las sugerencias del nutricionista, seguidas de los compromisos asumidos y las tareas asignadas para el próximo sprint.

### Observaciones del Product Owner (Profesor)

- Modificación de nombre de usuario: El profesor sugirió que no es recomendable permitir que los nutriólogos o pacientes cambien su nombre de usuario, ya que esto podría generar problemas a largo plazo en la gestión de cuentas.

### Sugerencias del Nutricionista

- Se sugirió añadir más condiciones alimenticias para personalizar las dietas y permitir filtros escritos al momento de crear las dietas. Esto permitirá que las dietas sean más específicas y adaptadas a las necesidades individuales de cada paciente.
- Se discutió la importancia de proporcionar guías claras para que los pacientes registren correctamente sus medidas, especialmente en lo que respecta al IMC, ya que puede resultar confuso para algunos pacientes.
- El nutricionista propuso agregar un nuevo objetivo para pacientes que deseen mejorar su salud y hábitos sin centrarse específicamente en déficit o volumen. Esto podría ser útil para aquellos que buscan un enfoque más integral y saludable.
- Se sugirió optimizar la creación de dietas al añadir más filtros o detalles escritos, permitiendo que los nutricionistas ajusten las dietas de manera más precisa.

**ACUERDOS Y COMPROMISOS**

<b>Sección tratada</b>	<b>Observación</b>	<b>Lo vamos a tomar SI/NO</b>	<b>Sustentación</b>
Condiciones alimenticias y filtros en dietas	Añadir más condiciones alimenticias para personalizar dietas y/o permitir filtros escritos al crear dietas.	SI	Puedo investigar otras condiciones alimenticias existentes o tener el respaldo del stackeholder que tiene información sobre eso. Si no se implementa un filtro por los alimentos insertados, simplemente se añadiran requisitos sean visibles para los pacientes y que les restrinja seguir x dieta.
Recomendaciones para el registro de medidas	Incluir guías claras para el registro de medidas, considerando la confusión que puede generar el IMC.	NO	La idea podría funcionar muy bien si se quiere llegar a ciertos objetivos, sin embargo, por ahora no la tomaré debido a que es mucho calculo para añadir a la lógica y no dejaría tiempo para otras tareas con mayor prioridad, por lo cual se puede dejar para versiones futuras.
Objetivo "Mejorar hábitos"	Agregar un nuevo objetivo que no sea déficit o volumen, sino centrado en mejorar hábitos de vida.	SI	Existirán pacientes que de repente no tengan bien un objetivo fijo y solo prefieren tener un hábito más saludable, por lo que se añadirá una opción general que le permitirá hacer tanto rutinas/dietas de

			deficit como de volumen, igualmente se filtrarán dependiendo de otros datos que tiene registrado el paciente.
Observación del Product Owner	El profesor indicó que no se debe permitir que los usuarios cambien su nombre de usuario para evitar problemas a largo plazo.	SI	Se ajustará la funcionalidad para que los usuarios no puedan modificar su nombre de usuario una vez registrado, garantizando la integridad de las cuentas de usuario.

## NUEVA HU – HU16.1 NOTIFICACIONES PARA EL USUARIO

Durante la reunión, también se discutió la funcionalidad de las notificaciones y se decidió realizar ajustes en la HU16 para incluir las siguientes mejoras:

- ✓ Notificaciones para las reuniones entre el nutricionista y el paciente.
- ✓ Notificaciones sobre "consejos del día" que se enviarán a los pacientes.
- ✓ Notificaciones relacionadas con las actividades programadas en el calendario del paciente.

HU16.1 – Notificaciones para el Usuario (Actualizada)
Como paciente o nutricionista, quiero recibir notificaciones sobre actividades programadas, reuniones y consejos del día en base a mis objetivos, para mantenerme al día con mi plan de salud y recibir recordatorios importantes.
<b>Criterios de aceptación:</b>
Escenario 1 – Notificaciones:  Cuando el usuario ingresa a su e-mail o a la sección de notificaciones en la página, verá nuevas alertas sobre actividades programadas, reuniones con el nutricionista, consejos diarios de salud.

Este ajuste a la HU16.1 permitirá una comunicación más fluida entre los pacientes y nutricionistas, y asegurará que los pacientes se mantengan informados sobre su progreso y próximos pasos.

## 7.6 Product Backlog [actualizado]:

Para estructurar el Product Backlog actualizado, hemos seguido los siguientes pasos:

- **Revisión de los compromisos asumidos durante la Sprint Retrospective:**  
Se discutieron las tareas pendientes y los puntos críticos que surgieron a partir de la retroalimentación del Product Owner (profesor) y del nutricionista.
- **Priorización de las historias de usuario (PBI's):**  
Con base en la importancia de cada tarea para mejorar la funcionalidad de la aplicación y considerando las dependencias entre las historias de usuario, hemos asignado prioridades a cada una.
- **Establecimiento de estimaciones:**  
Se han asignado estimaciones de tiempo a cada historia de usuario, considerando la complejidad de la tarea y los recursos disponibles.

ID	HISTORIA DE USUARIO	PRIORIDAD	ESTIMACION	DEPENDENCIA	SPRINT
HU16.1	Notificaciones para usuario	860	7 días	HU12, HU13	4
HU17	Registro semanal de medidas y seguimiento de progreso	830	6 días	HU04	4
HU19	Recomendación Inicial de Dietas y Rutinas Personalizadas	820	4 días	HU04	4
HU18	Recomendaciones automáticas basadas en datos alterados	800	6 días	HU16	4
HU20	Programar reuniones con el nutriólogo	760	4 días	HU10	4

## PROCEDIMIENTO SEGUIDO PARA LA ACTUALIZACIÓN:

- **Análisis de los resultados de la Sprint Retrospective:** Se discutieron las tareas pendientes y las mejoras que se iban a implementar. Esta revisión permitió priorizar los puntos más urgentes para el siguiente Sprint.
- **Priorización en función de la importancia:** Las historias que afectan directamente la funcionalidad del sistema, como las relacionadas con las notificaciones, la recomendación de dietas, y las reuniones con el nutriólogo, fueron asignadas a una alta prioridad.
- **Ajuste de las historias de usuario (PBI's):** Se modificaron las historias de usuario existentes (como HU16) y se agregaron nuevas historias (como HU16.1) para incorporar las mejoras necesarias.

- **Asignación estimaciones:** Se aseguraron las estimaciones de tiempo para las historias de usuario, con el fin de garantizar que el equipo pueda llevar a cabo el trabajo de manera eficiente y en el tiempo correspondiente a la duración del próximo sprint.

## 8. SPRINT 4

### 8.1 Sprint Backlog:

#### OPTIMIZACIÓN Y CONEXIÓN SALUDABLE

- **Fecha de Inicio del Sprint:** 14 de noviembre del 2024.
- **Fecha de Fin del Sprint:** 4 de diciembre del 2024.
- **Duración del Sprint:** 3 semanas

#### HISTORIAS DE USUARIOS

Este Sprint corresponde del 14/11/24 al 4/12/24, en el que se desarrollaron las historias de usuarios del sprint 4: HU16.1, HU17.1, HU19 y HU20.1

#### INCREMENTO PREVIO (SPRINT 1)

HU01 – Registrar usuario	
Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.	
<b>Criterios de aceptación:</b>	
Escenario 1	
	La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.
	El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.

HU03 – Acceso para usuarios	
Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario	

**Criterios de aceptación:****Escenario 1**

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

**HU02 – Cuestionario para pacientes nuevos**

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

**Criterios de aceptación:****Escenario 1 – Cuestionario**

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

**HU04 – Menú de Inicio**

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

**Criterios de aceptación:****Escenario 1 - Menú de inicio**

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

**INCREMENTO PREVIO (SPRINT 2)**

## HU05 - Insertar rutinas personalizadas

Como nutriólogo, quiero crear y personalizar rutinas de ejercicios agregando descripciones específicas para recomendarlas a mis pacientes registrados.

### Criterios de aceptación:

#### Escenario 1

Cuando el nutriólogo accede a la sección de crear rutinas, se le redirige a una ventana donde puede buscar ejercicios en la base de datos, seleccionarlos y organizarlos en una rutina personalizada. Además, el nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar la rutina según los objetivos específicos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la rutina y recomendarla a sus pacientes registrados.

## HU06 – Insertar dietas personalizadas

Como nutriólogo, quiero crear y personalizar dietas agregando detalles específicos para recomendarlas a mis pacientes registrados.

### Criterios de aceptación:

#### Escenario 1

Cuando el nutriólogo accede a la sección de crear dietas, se le redirige a una ventana donde puede ingresar y organizar planes de alimentación personalizados. El nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar las dietas según objetivos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la dieta y recomendarla a sus pacientes registrados.

## HU07 - Editar rutinas y dietas

Como nutriólogo, quiero poder editar los detalles de las dietas y rutinas para asegurarme de que la información esté actualizada y sea precisa.

### Criterios de aceptación:

## Escenario 1 - Acción editar rutinas

Cuando el nutriólogo selecciona la opción para editar una rutina que él creó, podrá modificar sus datos, incluyendo el tipo de rutina, la categoría, su descripción y los ejercicios incluidos. Estas modificaciones permitirán mantener la base de datos actualizada.

## Escenario 2 - Acción editar dietas

Cuando el nutriólogo selecciona la opción para editar una dieta que él creó, podrá modificar sus datos, incluyendo el tipo de dieta, la descripción, y los alimentos incluidos en esta dieta. Estas modificaciones permitirán mantener la base de datos actualizada.

## HU08 – Ver detalle de una rutina y dieta

Como paciente, quiero visualizar a detalle la dieta y rutina dentro de una lista para tenerla guardarla sea en calendario o por texto.

### Criterios de aceptación:

#### Escenario 1 – Ver detalle de una rutina

Cuando el usuario selecciona la opción para ver el detalle de una rutina, podrá visualizar sus datos, incluyendo el tipo de rutina, el nutriólogo quien lo creó, la categoría, su descripción y una lista de los ejercicios incluidos. También unos botones para descargarlo o guardarlo a tu calendario.

#### Escenario 2 - Ver detalle de una dieta

Cuando el usuario selecciona la opción para ver el detalle de una dieta, podrá visualizar sus datos, incluyendo el tipo de dieta, el nutriólogo quien lo creó, su descripción y una lista de alimentos que debería consumir. También unos botones para descargarlo o guardarlo a tu calendario.

## HU09 – Ver lista de alimentos

Como nutriólogo, quiero ver una lista de alimentos disponibles en la base de datos para poder consultarlos y utilizarlos en la creación de dietas personalizadas para mis pacientes.

### Criterios de aceptación:

## Escenario 1

Cuando el nutriólogo accede a la sección de alimentos, se le muestra una lista de alimentos que incluye el nombre, tipo (desayuno, almuerzo, cena, merienda), y sus nutrientes principales (proteínas, grasas, carbohidratos). El nutriólogo puede buscar alimentos específicos, filtrarlos por tipo o por nutrientes, y seleccionar aquellos que desea incluir en las dietas de sus pacientes.

### HU12 – Ver lista de ejercicios

Como nutriólogo, quiero ver una lista de ejercicios en la base de datos para poder consultarlos y utilizarlos en la creación de rutinas personalizadas para mis pacientes.

#### Criterios de aceptación:

## Escenario 1

Cuando el nutriólogo accede a la sección de ejercicios, se le muestra una lista que incluye el nombre del ejercicio, el grupo muscular al que pertenece, el tipo de ejercicio y una breve descripción. El nutriólogo puede buscar ejercicios específicos, filtrarlos por grupo muscular o tipo de ejercicio, y seleccionar aquellos que desea incluir en las rutinas de sus pacientes.

## INCREMENTO PREVIO (SPRINT 3)

### HU10 – Perfil de nutriólogos

Como nutriólogo, quiero tener acceso a un perfil de usuario para insertar información visible y crear mis rutinas y dietas para que más pacientes puedan visitarme y ponerse en contacto conmigo.

#### Criterios de aceptación:

## Escenario 1 – Perfil de nutriólogo

Cuando el nutriólogo entre a su perfil, podrá visualizar qué rutinas y dietas ha creado junto a un botón para crear otras, y una opción de editar perfil para añadir sus contactos, lugar en donde trabaja y descripción.

## HU11 – Perfil de pacientes

Como paciente, quiero tener acceso un perfil de usuario para visualizar mi actividad y gestionar mi información visible.

### Criterios de aceptación:

Escenario 1 – Perfil de paciente

Cuando el paciente ingrese su perfil, podrá visualizar que rutina o dieta ha tiene guardado, junto a una opción de editar perfil para añadir su descripción y gestionar la información solicitada en el cuestionario.

## HU13 – Ajustes de Usuario

Como usuario, quiero hacerles ajustes de seguridad a mí cuenta para no descuidarla en caso de que se me olvide la contraseña.

### Criterios de aceptación:

Escenario 1 – Ajustes de usuario

Cuando el usuario está en la opción de ajustes, tendrá una opción para actualizar contraseña, y también una para cerrar sesión.

## HU14 - Personalizar calendario de paciente

Como paciente, quiero gestionar un calendario personal para organizar mis rutinas de entrenamiento proporcionadas por el nutriólogo.

### Criterios de aceptación:

Escenario 1

El paciente llega a una opción donde puede acceder a su calendario. El paciente tendrá acceso a un calendario donde podrá agregar sus rutinas de ejercicios para cada día de la semana. Podrá asignar los días de entrenamiento, seleccionar qué rutinas realizará cada día, y programar los días de descanso. Esto le permitirá visualizar y organizar su plan de entrenamiento semanal.

## INCREMENTO ACTUAL (SPRINT 4)

### PROCEDIMIENTO PARA LA ORGANIZACIÓN DEL BACKLOG Y PRIORIZACIÓN DE HISTORIAS DE USUARIO

Durante la planificación del Sprint 4, el equipo de desarrollo realizó una revisión final del Backlog, enfocándose en completar las funcionalidades clave para garantizar el correcto funcionamiento de la aplicación. Este sprint fue diseñado como una etapa esencial para consolidar las bases técnicas y funcionales, asegurando una experiencia completa para los usuarios.

#### c) Revisión Inicial del Backlog

El equipo evaluó las HU restantes, priorizando aquellas que eran fundamentales para el cumplimiento de los objetivos principales de la aplicación. La intención fue cerrar funcionalidades críticas tanto para pacientes como para nutriólogos, preparando el producto para su uso operativo.

#### d) Acuerdos para la Priorización

El equipo consideró los siguientes criterios al priorizar las HU:

- **Funcionalidad esencial:** Se priorizaron las HU necesarias para que la aplicación funcione correctamente, dejando detalles estéticos y técnicos menores para el Sprint 5.
- **Interacción usuario-nutriólogo:** Se seleccionaron funcionalidades que mejoran la comunicación y personalización, esenciales para una experiencia fluida entre los usuarios y el nutriólogo.
- **Optimización del flujo de trabajo:** Se completaron HU dependientes de desarrollos previos, asegurando que todas las funciones estén plenamente integradas antes del ajuste final.

#### e) Selección de Historias de Usuario para el Sprint 3

Durante el Sprint 4, realizamos algunos ajustes en algunas Historias de Usuario para actualizar funcionalidades y adecuarlas a las nuevas necesidades del proyecto. Además, ciertas partes de las HU originales se reemplazaron o modificaron debido a la prioridad de tareas añadidas durante el desarrollo.

**HU16.1 – Notificaciones para el Usuario (Actualizada)**

Como paciente o nutricionista, quiero recibir notificaciones sobre actividades programadas, reuniones y consejos del día en base a mis objetivos, para mantenerme al día con mi plan de salud y recibir recordatorios importantes.

**Criterios de aceptación:**

Escenario 1 – Notificaciones:

Cuando el usuario ingresa a la sección de notificaciones en la página, verá nuevas alertas sobre actividades programadas, reuniones con el nutricionista, consejos diarios de salud.

## HU17.1 – Registro semanal de medidas y seguimiento de progreso

Como paciente, quiero ingresar mis medidas semanales (peso, perímetro de cintura y bíceps), para monitorear mi progreso en función de mis objetivos deportivos.

**Criterios de aceptación:**

Escenario 1

El paciente podrá ingresar sus medidas cada semana para monitorear su progreso. Estos datos se guardarán en su perfil para visualizar el cambio a lo largo del tiempo. Una vez ingresadas las medidas, el usuario no podrá volver a registrar datos hasta pasados 7 días. En caso de intentar ingresar nuevamente, el sistema le mostrará una alerta indicando que debe esperar hasta que transcurra una semana desde su última entrada.

## HU19 – Recomendación Inicial de Dietas y Rutinas Personalizadas

Como paciente recién registrado, quiero que el sistema me recomiende dietas y rutinas adecuadas a mis objetivos (déficit calórico o volumen) y experiencia física previa, para poder iniciar mi programa de forma segura y efectiva, considerando mis capacidades y posibles restricciones alimenticias.

**Criterios de aceptación:**

Escenario 1

El sistema recomendará dietas y rutinas basadas en el cuestionario realizado después del registro.

## HU20.1 – Programar reuniones con el nutriólogo

Como nutriólogo, quiero gestionar reuniones virtuales con mis pacientes, para realizar evaluaciones y brindar seguimiento personalizado.

### Criterios de aceptación:

#### Escenario 1

El nutriólogo podrá programar reuniones grupales o individuales con temas específicos (por ejemplo, "Dietas para principiantes"). Estas reuniones tendrán horarios predefinidos, y el sistema notificará al nutriólogo sobre las confirmaciones de asistencia de los pacientes. Además, podrá visualizar una lista de los asistentes junto con un resumen de su perfil, que incluirá sus dietas, seguimiento de medidas y progreso registrado.

### f) Decisión de Eliminar HU18 - Recomendaciones automáticas basadas en datos alterados

Durante la planificación del Sprint 4, se decidió eliminar la HU18 – Recomendaciones automáticas basadas en datos alterados. Aunque esta funcionalidad, que recomendaría nuevas dietas o rutinas en caso de que los datos del paciente no coincidan con sus objetivos, es valiosa, se determinó que no era esencial para el funcionamiento básico de la aplicación en esta etapa.

La decisión se tomó debido a:

- **Limitaciones de tiempo:** La implementación de esta funcionalidad requería un desarrollo más complejo, que podría haber retrasado la entrega de las funcionalidades principales del sprint.
- **Priorización de tareas críticas:** Se acordó enfocar los esfuerzos en culminar las funcionalidades necesarias para completar la interacción transaccional entre paciente y nutriólogo.

Se planteó la posibilidad de incluir esta HU como una mejora futura para el MVP, ya que representa una característica que puede aportar mucho valor a la experiencia del usuario, pero que no es imprescindible en esta fase inicial.

## HU18 – Recomendaciones automáticas basadas en datos alterados (ELIMINADA)

Como paciente, quiero que el sistema me recomiende nuevas dietas o rutinas en caso de que mis datos (peso, medidas, etc.) no estén alineados con mis objetivos de salud, para mejorar mi progreso.

### Criterios de aceptación:

#### Escenario 1 – Recomendaciones automáticas

Si los datos del paciente no coinciden con sus objetivos, el sistema generará nuevas recomendaciones de dietas o rutinas en su página de inicio.

#### Escenario 2 – Notificación por datos persistentes

Si tras dos semanas los datos siguen desalineados, se enviará una notificación al paciente y al nutriólogo para programar una reunión y revisar el plan de salud.

## 8.2 Sprint Planning:

### AGENDA TRATADA DURANTE LA CEREMONIA DE SPRINT PLANNING:

#### e) Revisión de las Historias de Usuario del Sprint 3:

- Se discutió la prioridad de las HU a trabajar, destacando su relevancia para asegurar el funcionamiento correcto de la aplicación y la interacción entre los usuarios (pacientes y nutriólogos).
- Se detallaron las tareas específicas para cada HU, asegurando que cada tarea requeriría más de 20 minutos de desarrollo.
- Se aclararon los criterios de aceptación para cada historia y se identificaron áreas donde Gonzalo, como programador, necesitaría trabajar en tareas técnicas y de implementación.

#### f) Asignación de responsabilidades:

- Gonzalo Tafur Bermúdez fue asignado como responsable de las tareas técnicas y de programación de las nuevas funcionalidades, siendo el único involucrado en la fase de desarrollo.
- Jasmin no estuvo involucrada en el desarrollo de estas funcionalidades, pero se encargó de planificar las reuniones de seguimiento y verificar la estructura de las funcionalidades para asegurar su correcta alineación con la experiencia del usuario.
- Se distribuyeron las tareas respetando la capacidad de Gonzalo para programar, mientras Jasmin se centró en la planificación de las actividades y la organización de las reuniones.

#### g) Determinación de dependencias y predecesoras:

- Las HU desarrolladas en este Sprint dependen principalmente del cuestionario, de los perfiles de nutriólogo y paciente, ya que estas funcionalidades requieren la información proporcionada por los usuarios para personalizar la experiencia dentro de la aplicación y puede existir una transacción de información entre el nutricionista y el paciente

## h) Aprobación del Sprint Backlog:

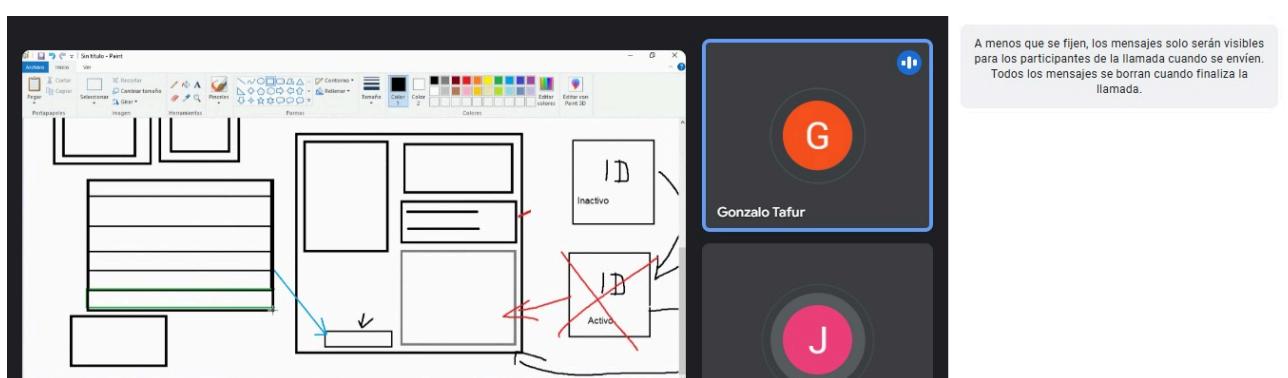
- Se llegó a un acuerdo sobre las tareas a realizar durante el Sprint 4, su duración y el responsable de cada tarea.
- Se priorizaron funcionalidades clave para asegurar la transaccionalidad básica entre los usuarios (pacientes y nutriólogos), con el objetivo de completar las funcionalidades más relevantes para el funcionamiento de la aplicación.

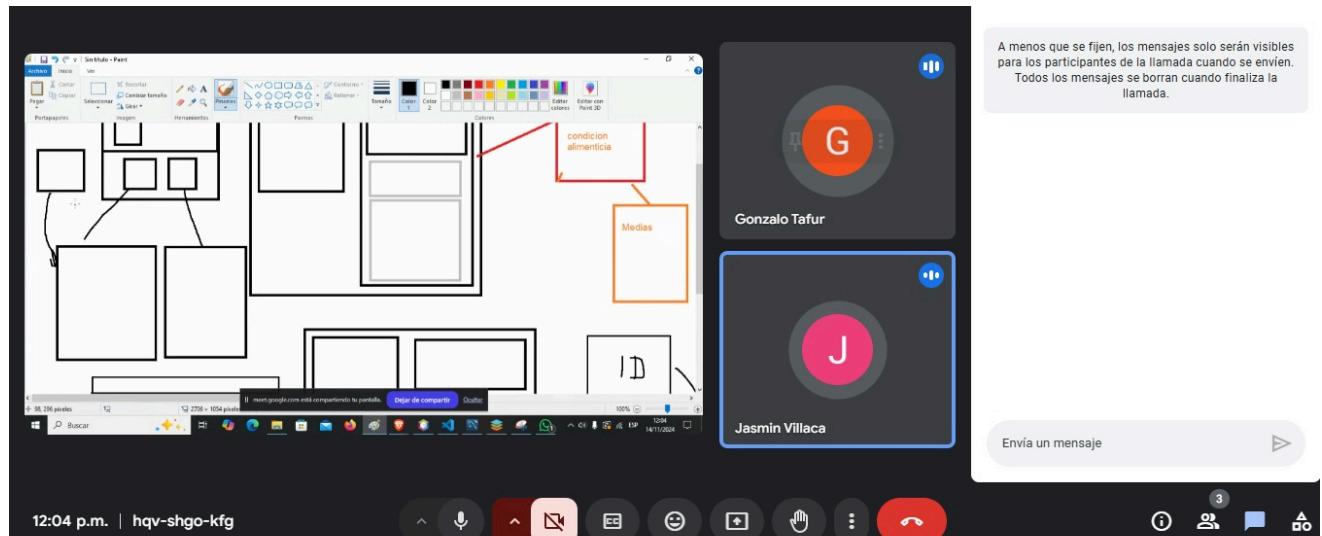
## PROCEDIMIENTO SEGUIDO PARA LA CEREMONIA:

- FECHA: La ceremonia de planificación del Sprint 4 se llevó a cabo el 14/11/2024.
- PARTICIPANTES: Incluyó a todo el equipo
- RESULTADOS DE LA CEREMONIA:

- ✓ **Sprint Backlog:** Se seleccionaron las HU y tareas del Sprint 4.
- ✓ **Duración de las tareas:** Cada tarea fue estimada para ajustarse a la capacidad de Gonzalo en la programación.
- ✓ **Predecesoras:** Las HU desarrolladas en este sprint dependerán del cuestionario y de los perfiles de usuario (paciente, nutriólogo)
- ✓ **Evidencia:** Se verificaron los prototipos y la estructura de las funcionalidades, asegurando que estuvieran alineadas con las necesidades de los usuarios.

## LÓGICA Y ESTRUCTURA DE LA BASE DE DATOS PARA EL REGISTRO DE MEDIDAS





Tareas/Tareas no técnicas	Estimación	Asignación de Trabajo
---------------------------	------------	-----------------------

HU16.1 – Notificaciones para el Usuario		
Crear página de notificaciones para pacientes y nutriólogos	3h	Tafur Bermúdez, Gonzalo
Implementar lógica de notificaciones según actividades programadas, reuniones y consejos diarios	3h	Tafur Bermúdez, Gonzalo
Pruebas de funcionalidad y experiencia de usuario para la visualización de notificaciones	1h	Tafur Bermúdez, Gonzalo

**HU17.1 – Registro semanal de medidas y seguimiento de progreso**

Crear sección dentro del perfil del paciente para ingresar sus medidas semanales (peso, perímetro de cintura y bíceps)	1h	Tafur Bermúdez, Gonzalo
Implementar botón para actualizar o modificar las medidas registradas	5h	Tafur Bermúdez, Gonzalo
Crear sección de "Seguimiento" dentro del perfil, donde se muestra el historial de medidas, incluyendo la fecha de actualización y los valores previos	4h	Tafur Bermúdez, Gonzalo
Implementar validación para evitar registrar datos antes de transcurrir 7 días	2h	Tafur Bermúdez, Gonzalo
Desarrollar alerta de advertencia cuando el paciente intente registrar datos antes de la semana completa	1h	Tafur Bermúdez, Gonzalo
Pruebas de funcionalidad y usabilidad para el registro de medidas, botones de actualización, y sección de seguimiento	1h	Tafur Bermúdez, Gonzalo

**HU19 – Recomendación Inicial de Dietas y Rutinas Personalizadas**

Crear el sistema de recomendación basado en el cuestionario del usuario	1h	Tafur Bermúdez, Gonzalo
Configurar las dietas y rutinas recomendadas según los objetivos del paciente (déficit calórico o volumen) y su experiencia física previa	1h	Tafur Bermúdez, Gonzalo
Implementar el proceso de recomendación automática al completar el cuestionario de registro	2h	Tafur Bermúdez, Gonzalo
Realizar pruebas de funcionalidad y validación de las recomendaciones personalizadas	1h30min	Tafur Bermúdez, Gonzalo

**HU20.1 – Programar reuniones con el nutriólogo**

Crear la interfaz para que el nutriólogo programe reuniones	2h	Tafur Bermúdez, Gonzalo
Implementar opciones para confirmar asistencia de pacientes y visualizar perfiles con dietas y progreso	3h	Tafur Bermúdez, Gonzalo
Configurar el sistema de notificaciones para las reuniones y confirmaciones de asistencia	2h30min	Tafur Bermúdez, Gonzalo
Pruebas de funcionalidad y experiencia de usuario para la programación de reuniones	1h	Tafur Bermúdez, Gonzalo

Durante la ceremonia de Sprint Planning, el equipo aprobó las tareas necesarias para implementar las funcionalidades del Sprint 4. Se priorizó la creación de secciones para el registro semanal de medidas, con validación de tiempos y alertas de advertencia, así como la implementación de la recomendación inicial de dietas y rutinas personalizadas para los pacientes. Además, se trabajó en la integración de notificaciones para mantener a los usuarios informados sobre actividades y reuniones programadas. Estas tareas fueron diseñadas para mejorar la experiencia y el seguimiento de los usuarios en su camino hacia el cumplimiento de sus objetivos.

### KANBAN BOARD JIRA (Contraseña y correo para el acceso)

<https://jasmin.atlassian.net/jira/software/projects/KAN/boards/1>  
[proyecdietasyrutinas@gmail.com](mailto:proyecdietasyrutinas@gmail.com)

DyROtafvill

### 8.3 Daily Scrum:

Durante el Sprint 4, el equipo de desarrollo se enfocó en implementar funcionalidades clave para el seguimiento de progreso de los pacientes, la programación de reuniones con nutriólogos y el sistema de notificaciones. Las historias de usuario trabajadas fueron HU16.1 (Notificaciones para el usuario), HU17.1 (Registro semanal de medidas y seguimiento de progreso), HU19 (Recomendación inicial de dietas y rutinas personalizadas) y HU20.1 (Programar reuniones con el nutriólogo). En las reuniones, que tuvieron una duración de 40 minutos a 1 hora, se discutieron avances, obstáculos y acciones para cumplir con los criterios de aceptación establecidos.

### ¿QUÉ HICE AYER?

Ayer, el equipo avanzó en la implementación de la funcionalidad para el registro de medidas semanales (HU17.1), permitiendo a los pacientes ingresar y visualizar su progreso en medidas clave como peso, perímetro de cintura y bíceps. Se completó la integración de notificaciones (HU16.1), destacando alertas cruciales como recordatorios de actividades programadas para los pacientes y notificaciones para los nutriólogos sobre pacientes registrados a sus reuniones. También se trabajó en la recomendación inicial de dietas y rutinas (HU19), asegurando que el sistema sugiera planes adecuados a los objetivos del paciente. Además, se resolvieron algunos problemas con el flujo de las

reuniones (HU20.1), permitiendo que los nutriólogos puedan confirmar la asistencia de los pacientes y visualizar la lista de participantes.

## ¿QUÉ HARÉ HOY?

Hoy, el equipo se enfocará en pulir las funcionalidades implementadas, asegurando que las notificaciones se muestren correctamente y que el flujo de las reuniones sea fluido para los nutriólogos y pacientes. Se continuará con la mejora del sistema de recomendaciones iniciales, garantizando que las sugerencias sean personalizadas según los datos del paciente. Además, se verificarán los registros de medidas y las alertas para asegurar que la lógica de tiempo y validación esté funcionando correctamente.

## ¿QUÉ IMPEDIMENTOS TENGO?

Un problema surgió con el flujo de las reuniones (HU20.1), ya que inicialmente no se había implementado el botón de confirmación de asistencia y no se mostraba la lista de participantes en la vista inicial. Este problema se resolvió tras investigar cómo debería funcionar el flujo de las reuniones, asegurando que el nutriólogo pueda gestionar correctamente la confirmación de asistencia y visualizar a los participantes.

En cuanto a las notificaciones (HU16.1), la lógica para gestionar las alertas presentó cierta complejidad, ya que se tuvo que ajustar la prioridad de las tareas. Algunas alertas no fueron implementadas, pero las más importantes, como recordatorios de actividades programadas y reuniones, se mantuvieron. Esto implicó que se hicieran algunos ajustes en el código y se simplificaran ciertas funcionalidades, pero el flujo general de las notificaciones está operativo.

## ¿CÓMO RESOLVI EL PROBLEMA?

El equipo trabajó en conjunto para ajustar la lógica de las reuniones y garantizar que el flujo de confirmación de asistencia y visualización de participantes fuera intuitivo y funcional. Además, se revisó el sistema de notificaciones, priorizando las alertas más cruciales para el seguimiento del usuario. Aunque algunas funcionalidades adicionales se eliminaron temporalmente, se aseguraron de que las tareas clave, como los recordatorios y las notificaciones sobre reuniones, fueran entregadas a tiempo y de acuerdo a los requisitos del Product Owner (Profesor)

## EVIDENCIA DE LA REUNIÓN #1 (JUEVES 14/11)

- DURACIÓN: 11:30 am – 12:30 pm
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Definición de funcionalidades iniciales para las HU de Reuniones y Registro de Medidas:

#### ✓ REUNIONES (HU20.1):

En esta primera reunión, se discutió de manera inicial cómo organizar y estructurar la sección de reuniones dentro de la plataforma. La idea era definir el espacio donde aparecerían las reuniones y cómo los pacientes podrían interactuar con esta funcionalidad. No se profundizó en detalles técnicos, pero se acordó que las reuniones tendrían una sección visible para los pacientes y nutriólogos, permitiendo ver cuándo están programadas. La prioridad inicial era definir su ubicación y la estructura general.

#### ✓ REGISTRO DE MEDIDAS (HU17.1):

Se planteó la necesidad de permitir que los pacientes registren sus medidas semanales, tales como peso, perímetro de cintura y bíceps. Se discutió brevemente la forma en que

esta funcionalidad se integraría en el perfil del paciente, sin entrar en detalles sobre validaciones o advertencias en ese momento. La idea era establecer el espacio dentro del perfil donde se registrarían estas medidas.

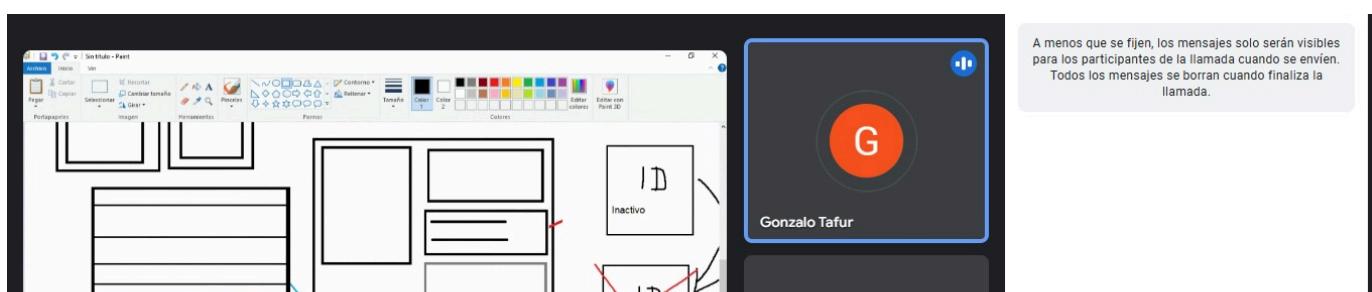
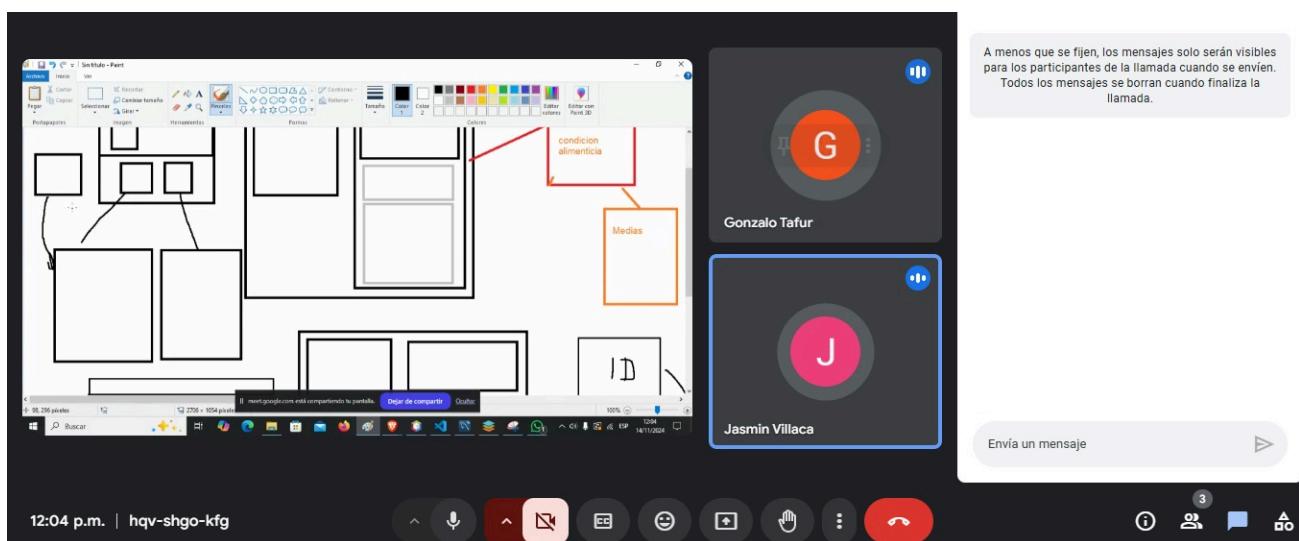
## Estructuración del flujo general de funcionalidades:

- ✓ Se trató de manera inicial el flujo para las funcionalidades de reuniones y registro de medidas, pero sin entrar en detalles técnicos sobre cómo se manejarían los datos o interacciones. El enfoque estuvo en la ubicación y la lógica básica de cómo los pacientes y nutriólogos interactuarían con estas secciones.

## Discusión de otras HU del sprint:

- ✓ Aunque se tocó brevemente el tema de las demás historias de usuario que se implementarían en el sprint, el foco de esta reunión estuvo principalmente en las reuniones y el registro de medidas, sin entrar en un análisis profundo de otras funcionalidades.
- ✓ Se acordó que las funcionalidades de reuniones y medidas se abordarían primero, para luego continuar con el resto de HU una vez que estas dos estuvieran claras en términos de estructura y flujo.

## RESULTADOS DE LA REUNION



## EVIDENCIA DE LA REUNIÓN #2 (DOMINGO 24/11)

- DURACIÓN: 22:00 – 22:53 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Progreso en HU20 – Reuniones:

- ✓ Se presentó el avance en la funcionalidad de las reuniones, específicamente en la estructura y la interfaz de la creación de estas. Inicialmente, se había diseñado una sección para visualizar a los pacientes que ingresaron a las reuniones, pero sin el botón para confirmar asistencia. Solo se mostraba la lista de participantes si estos habían ingresado directamente.
- ✓ Se discutió la necesidad de ajustar este flujo, de modo que los pacientes puedan confirmar su asistencia antes de ingresar a la reunión, mejorando así la interacción y la lógica del proceso. Este cambio fue aprobado y se implementará en la próxima fase de desarrollo.

### Progreso en HU17 – Registro de Medidas:

- ✓ No hubo cambios significativos en la funcionalidad del registro de medidas en esta reunión. La implementación sigue siendo la misma, permitiendo que los pacientes registren sus medidas como peso y perímetros sin modificaciones adicionales por ahora.

### Revisión y ajustes en las notificaciones:

- ✓ Se revisaron las notificaciones dentro de la plataforma, que por el momento eran automáticas, sin un detonante específico para ciertas acciones. Las notificaciones actuales se limitaban a consejos generales, como advertencias sobre nuevas dietas agregadas o actualizaciones generales, pero no había lógica aún para enviar recordatorios más específicos (como recordar al paciente que tiene una rutina programada o que un nutriólogo ha creado una nueva reunión).
- ✓ Se acordó que la lógica de las notificaciones debe mejorar para asegurar que sean más relevantes y útiles, por ejemplo, enviando recordatorios personalizados según el tipo de actividad del usuario (paciente o nutriólogo). Aunque los ajustes no fueron implementados aún, se estableció como una prioridad para las siguientes reuniones.

## RESULTADOS DE LA REUNION

### 1) PROTOTIPO INTERFAZ DE CREACIÓN DE REUNIONES

#### Programar una nueva reunión

Nombre

Fecha  
 dd/mm/aaaa

Hora  
 : :

¿En qué plataforma se va a hacer la reunión?

The screenshot shows a medical application interface. On the left, there's a sidebar with 'Reuniones' (Meetings) and a list of meetings. The main area displays a table of exercises with columns for name, intensity, and volume. To the right, there's a messaging interface with two participants, 'Gonzalo Tafur' and 'Jasmin Villaca', each with a profile icon and a message input field.

**Mensajes en la llamada**

A menos que se fijen, los mensajes solo serán visibles para los participantes de la llamada cuando se envíen. Todos los mensajes se borran cuando finaliza la llamada.

## Pacientes asistidos a la reunión

Reunion	Nombre	Paciente	Biografía	Acciones
Reunión de domingo 19:00	Reunión de domingo	Gonzalo Tafur Bermúdez	Hola, esta es mi biografía	<a href="#">Ver seguimiento</a> <a href="#">Ver su Perfil</a>

Retroceder

2024. Todos los derechos reservados.

## 2) NOTIFICACIONES (IDIEA INICIAL)

[-< Retroceder](#)

### Notificaciones

2024-11-26T19:28:18.267  
**Un nutrólogo a programado una nueva reunión.**  
 2024-11-26T19:03:40.010  
 ¡Una mente sana comienza con un cuerpo sano!  
 2024-11-26T19:03:00.010  
 ¿Qué piensas comer hoy? Recuerda que puedes decidir dentro de nuestro listado de dietas, que filtra según tu información de paciente  
 2024-11-26T19:00:00.010  
 ¿Qué piensas comer hoy? Recuerda que puedes decidir dentro de nuestro listado de dietas, que filtra según tu información de paciente  
 2024-11-26T18:59:40.007  
 ¿Ocupado y full esta semana? nunca esta demás hacer un par de ejercicios para ganar energías  
 2024-11-25T11:51:10.040  
 Se ha creado una nueva rutina para ti. Puedes visitarla en  
 2024-11-25T10:56:00.010  
 ¡Recuerda hacer ejercicio y beber mucha agua!  
 2024-11-25T10:55:40.003  
 ¡Una mente sana comienza con un cuerpo sano!

### 3) REGISTRO DE MEDIDAS

Fecha de modificación	Frecuencia de ejercicios	Condición alimenticia	Peso corporal	Estatura	Perímetro de cintura	Perímetro de cadera	Perímetro de muslo	Perímetro de brazo	Objetivo
2024-11-18T23:21:49.833	1 hora a diario	Nada							Deficit
2024-11-19T11:30:30.283	30 minutos a diario	Nada	73.77	1.67					Deficit
2024-11-19T12:50:15.057	30 minutos a diario	Nada	73.77	1.67					Deficit
2024-11-19T15:31:49.030	1 hora a diario	Nada	70.77	1.67					Deficit
2024-11-19T16:39:00.843	1 hora a diario	Nada	65.00	1.70					Volumen
2024-11-19T17:09:00.413	1 hora a diario	Nada	67.00	1.70					Deficit
2024-11-19T17:46:15.337	30 minutos a diario	Nada	67.45	1.70					Deficit

etroceder

2024. Todos los derechos reservados.

### EVIDENCIA DE LA REUNIÓN #3 (LUNES 02/12)

- DURACIÓN: 22:30 – 23:05 horas
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

#### Ajustes en la funcionalidad de las reuniones:

- ✓ Se revisaron y ajustaron los procesos relacionados con la creación y visualización de reuniones. Inicialmente, el sistema solo mostraba a los pacientes que ingresaron a la reunión, sin opción de confirmación de asistencia.
- ✓ En esta reunión, se incorporó la opción para que los pacientes confirmen su asistencia a las reuniones, lo que permitirá al nutriólogo saber quién está presente y asegurar el flujo adecuado de la sesión.
- ✓ Además, se permitió que los pacientes puedan cancelar su asistencia si es necesario, y el nutriólogo podrá revisar la lista de pacientes que confirmaron su participación.

#### Notificaciones:

- ✓ Se validó el funcionamiento de las notificaciones automáticas, las cuales ya incluyen alertas sobre la creación y actualización de reuniones, y recordatorios de rutinas para los pacientes.
- ✓ Se discutió la lógica detrás de estas notificaciones y se plantearon posibles mejoras para personalizar aún más los mensajes, como asegurarse de que los pacientes reciban notificaciones específicas sobre las actividades programadas para cada día, y que los nutriólogos sean notificados cuando sus pacientes asistan a las reuniones.

## Recomendaciones iniciales basadas en el cuestionario:

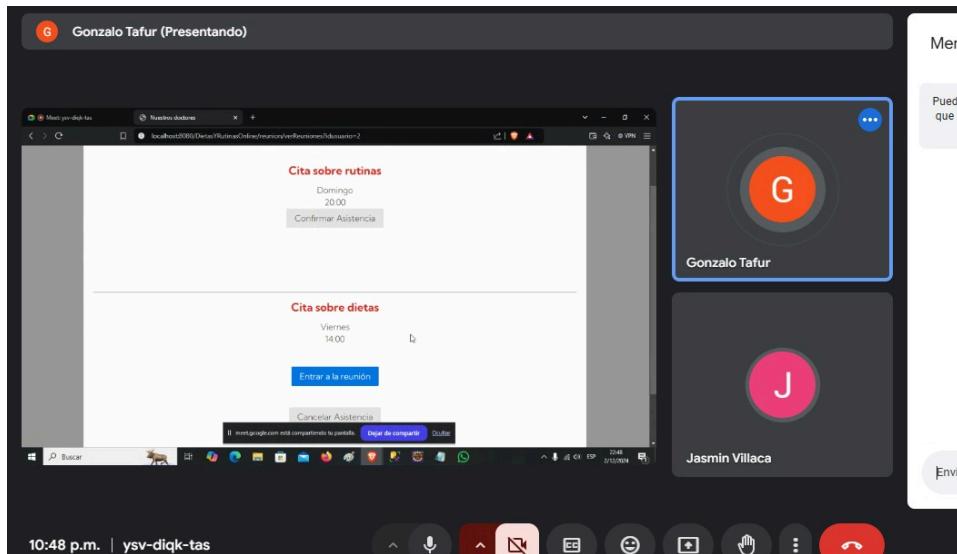
- ✓ Se presentó el avance en la funcionalidad de recomendaciones personalizadas basadas en el cuestionario inicial del paciente.
- ✓ El sistema ahora muestra rutinas y dietas recomendadas de acuerdo con los objetivos (volumen, déficit calórico) y el nivel de entrenamiento (principiante, intermedio, avanzado) especificados por el paciente durante el registro.
- ✓ Estas recomendaciones aparecen de forma automática debajo del calendario, permitiendo a los pacientes ver las opciones más relevantes según sus respuestas en el cuestionario.

## Medidas y alertas:

- ✓ Se mantuvo el diseño original para las medidas, sin cambios importantes por el momento.
- ✓ Se discutió la implementación de una alerta para pacientes que no hayan actualizado sus medidas en más de 7 días, con el fin de mantener un seguimiento constante de su progreso.

## RESULTADOS DE LA REUNION

### 1) REUNIONES ACTUALIZACION



### Programar una nueva reunión

Motivo

Día

Hora

¿En qué plataforma se va a hacer la reunión?

Cita sobre dietas  
Viernes  
14:00  
[Enlace de la reunión](#) [Desactivar Reunion](#)

Reunión de presentación  
MARTES  
18:30  
[Enlace de la reunión](#) [Desactivar Reunion](#)

Cita sobre rutinas  
DOMINGO  
20:00  
[Enlace de la reunión](#) [Desactivar Reunion](#)

Pacientes asistidos [Programar una nueva reunión](#)

## 2) RECOMENDACIONES INICIALES

Bienvenido, paciente nuevo. Antes de ingresar al menú, queremos que empieces eligiendo las dietas y rutinas que realizarás.

### Rutinas

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Definición Superior 1	Deficit	Tren Superior	Principiante	<p>Esta rutina está diseñada para principiantes en déficit calórico. Comienza con el press de banca y realiza repeticiones altas (8-15). Se recomienda un descanso de 2 minutos entre series. La carga de peso debe ser ligera y enfocarse en un mayor número de repeticiones para promover la quema de grasa. Cada ejercicio está orientado a trabajar los músculos del tren superior de manera efectiva mientras se mantiene la técnica adecuada.</p> <p>Esta rutina está pensada</p>	<a href="#">Ver detalles</a>

## 3) NOTIFICACIONES

<- Retroceder

### Notificaciones

2024-12-03T19:29:40.670

El paciente Tester confirmó unirse a la reunión Cita sobre rutinas

2024-12-03T17:25:55.710

El paciente Gonzalo confirmó unirse a la reunión Reunión de presentación

2024-12-03T11:01:51.587

El paciente Tester confirmó unirse a la reunión Reunión de presentación

## 8.4 Sprint Review:

Se presentará un resumen del incremento del producto completado durante este cuarto sprint.

### ALCANCE DEL SPRINT

- HU16.1 - Notificaciones para el Usuario.
- HU17.1 - Registro semanal de medidas y seguimiento de progreso.
- HU19 - Recomendación Inicial de Dietas y Rutinas Personalizadas.
- HU20.1 - Programar reuniones con el nutriólogo.

### RESULTADOS DEL SPRINT

#### ○ NOTIFICACIONES PARA EL USUARIO

Se completó la funcionalidad de notificaciones personalizadas. Ahora, tanto pacientes como nutriólogos reciben alertas sobre actividades programadas, reuniones y consejos diarios de salud según los objetivos personales establecidos.

Las notificaciones se muestran en la página de notificaciones del usuario y están relacionadas con los eventos y tareas programadas, como recordatorios de reuniones y actualizaciones de rutinas.

El sistema ahora garantiza que las notificaciones sean relevantes y se alineen con las preferencias y objetivos del usuario.

#### PACIENTE

Notificaciones	
2024-12-03T19:33:37.947	Un nutriólogo a programado una nueva reunión.
2024-12-03T19:33:09.707	Un nutriólogo a programado una nueva reunión.
2024-12-03T17:24:37.443	Un nutriólogo a programado una nueva reunión.
2024-12-03T17:11:40.333	Hoy tienes una cita programada.
2024-12-03T12:46:40.020	Hoy tienes una cita programada.
2024-12-03T11:00:56.077	Un nutriólogo a programado una nueva reunión.
2024-12-03T11:00:13.597	Un nutriólogo a programado una nueva reunión.
2024-12-02T21:29:40.010	¡Recuerda hacer ejercicio y beber mucha agua!
2024-12-02T21:29:00.013	¡Empieza un nuevo día! No olvides revisar tu horario y las rutinas programadas
2024-12-02T11:40:40.017	¡Hoy tienes una actividad programada! Ve a tu horario de Rutinas

## NUTRIOLOGO

<- Retroceder

### Notificaciones

2024-12-03T19:29:40.670  
El paciente Tester confirmó unirse a la reunión Cita sobre rutinas  
2024-12-03T17:25:55.710  
El paciente Gonzalo confirmó unirse a la reunión Reunión de presentación  
2024-12-03T11:01:51.587  
El paciente Tester confirmó unirse a la reunión Reunión de presentación

## o REGISTRO SEMANAL DE MEDIDAS Y SEGUIMIENTO DE PROGRESO

La funcionalidad para que los pacientes ingresen sus medidas semanales fue finalizada. Los pacientes ahora pueden registrar datos como su peso, perímetro de cintura y bíceps.

El sistema guarda los datos de progreso en el perfil del paciente y muestra cómo ha cambiado con el tiempo.

Se implementó una restricción de 7 días entre registros de medidas para evitar entradas prematuras. En caso de que el paciente intente registrar medidas antes de transcurrir una semana, recibirá una alerta informando que debe esperar.

Información

Debe esperar al menos 7 días antes de poder realizar otra modificación.

### Dieta(s) que estoy haciendo

Nombre	Objetivo	Descripción		Acciones			
Fecha de modificación	Frecuencia de ejercicios	Condición alimenticia	Peso corporal	Estatura	Perímetro de cintura	Perímetro de cadera	Perímetro de muslo
2024-11-27T10:30:38.813	A veces o nada	Lacteos					
2024-12-01T00:20:40.320	A veces o nada	Lacteos					
2024-12-01T00:24:06.197	A veces o nada	Nada					
2024-12-01T09:31:07.940	30 minutos a diario	Lacteos	75.20	1.80			
2024-12-01T11:44:38.023	30 minutos a diario	Lacteos	69.00	1.81			

Retroceder

2024. Todos los derechos reservados.

## ○ RECOMENDACIÓN INICIAL DE DIETAS Y RUTINAS PERSONALIZADAS:

El sistema ahora recomienda dietas y rutinas personalizadas para los pacientes recién registrados, basándose en las respuestas del cuestionario completado tras el registro.

Estas recomendaciones consideran los objetivos del paciente (déficit calórico o volumen) y su experiencia física previa, asegurando que cada paciente reciba un plan adaptado a sus necesidades específicas.

La funcionalidad fue probada y se validó que las recomendaciones se ajustan a los datos ingresados por el paciente en el cuestionario inicial.

### Nuestras recomendaciones iniciales son

Bienvenido, paciente nuevo. Antes de ingresar al menú, queremos que empieces eligiendo las dietas y rutinas que realizarás.

#### Rutinas

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Definición Superior 1	Deficit	Tren Superior	Principiante	<p>Esta rutina está diseñada para principiantes en déficit calórico. Comienza con el press de banca y realiza repeticiones altas (8-15). Se recomienda un descanso de 2 minutos entre series. La carga de peso debe ser ligera y enfocarse en un mayor número de repeticiones para promover la quema de grasa. Cada ejercicio está orientado a trabajar los músculos del tren superior de manera efectiva mientras se mantiene la técnica adecuada.</p> <p>Esta rutina está enfocada en la quema de grasa y la</p>	<a href="#">Ver detalles</a>

#### Diетas

Nombre	Objetivo	Descripción	Acciones
Dieta de deficit	Deficit		<a href="#">Ver detalles</a> <a href="#">Seguir dieta</a>
Dieta #1	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a> <a href="#">Seguir dieta</a>
Dieta #2	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a> <a href="#">Seguir dieta</a>
Dieta #2	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a> <a href="#">Seguir dieta</a>
Dieta #3	Deficit	Esta dieta solo es de prueba	<a href="#">Ver detalles</a> <a href="#">Seguir dieta</a>
		Esta dieta está diseñada para apoyarte en tu objetivo de aumentar masa muscular de forma saludable. Comienza el día con un licuado de	

## ○ PROGRAMAR REUNIONES CON EL NUTRÓLOGO

Los nutriólogos ahora pueden programar reuniones virtuales con sus pacientes, tanto grupales como individuales. Se implementaron horarios predefinidos para las reuniones, y los pacientes podrán confirmar su asistencia.

Una vez confirmada la asistencia, el nutrólogo puede visualizar una lista de los asistentes y revisar un resumen de su perfil, que incluye información sobre sus dietas, seguimiento de medidas y progreso registrado.

Esta funcionalidad facilita la gestión de reuniones y seguimiento personalizado para los nutriólogos.

### Reuniones disponibles

#### Cita sobre dietas

Viernes  
14:00

[Entrar a la reunión](#)

[Cancelar Asistencia](#)

#### Reunión de presentación

MARTES  
18:30

### Reuniones

Cita sobre dietas  
Viernes  
14:00

[Enlace de la reunión](#) [Desactivar Reunion](#)

Reunión de presentación  
MARTES  
18:30

[Enlace de la reunión](#) [Desactivar Reunion](#)

Cita sobre rutinas  
DOMINGO  
20:00

[Enlace de la reunión](#) [Desactivar Reunion](#)

Pacientes asistidos [Programar una nueva reunión](#)

Fue  
Infe

Fue  
Infe

Rut  
pru

Def  
Infe

## Programar una nueva reunión

Motivo

Día

Hora

¿En que plataforma se va ha hacer la reunión?

Zoom

Meet

## OBSERVACIONES DEL PRODUCT OWNER (PROFESOR)

PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo, Marco Aurelio Espinoza Rivera (Profesor)

El profesor menciono un punto importante durante la reunión, no se debe permitir que un nutriólogo programe dos reuniones a la misma hora y el mismo día. Por ejemplo, si un nutriólogo ya tiene una reunión programada para el martes a las 12, no podrá agendar otra para el mismo día y hora. Esto también aplica para los pacientes: no deben poder asistir a dos reuniones con nutriólogos distintos el mismo día a la misma hora. En resumen, solo se puede agendar una reunión por paciente y nutriólogo en el mismo día y hora.

## EVIDENCIA DE LA REUNION REALIZADA EL DIA 12/11/24 (fin de la reunión 20:20)

The screenshot shows a web application interface for dietary and exercise recommendations. At the top, there is a header with the date "31/12/2024" and a link to "Sala del curso". Below the header, the URL "localhost:8080/DietasYRutinasOnline/usuario/grabarCuestionario" is visible. On the right side of the screen, there is a video call window showing a person's face with the name "GONZALO TAFUR BE..." displayed below it. The main content area has a yellow background and displays the following text:

**Nuestras recomendaciones iniciales son**

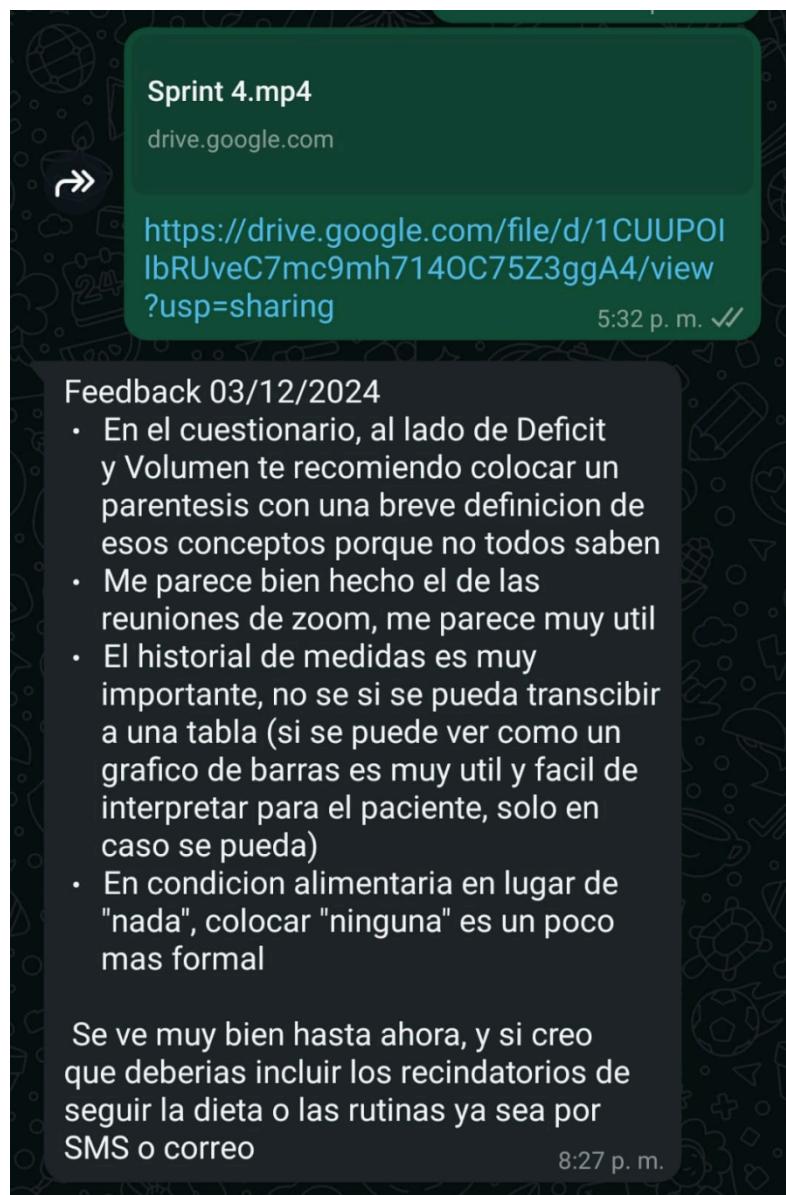
Bienvenido, paciente nuevo. Antes de ingresar al menú, queremos que empieces eligiendo las dietas y rutinas que realizarás.

**Rutinas**

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Fuerza Superior 1	Volumen	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en fase de volumen. Comienza con la prensa de banca para activar los músculos del pecho. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice pesos ligeros para asegurar una técnica adecuada. La rutina trabaja los principales músculos del tronco.	<a href="#">Ver detalles</a>

## SUGERENCIAS DEL NUTRÓLOGO (COMENTARIOS 03/12/2024)

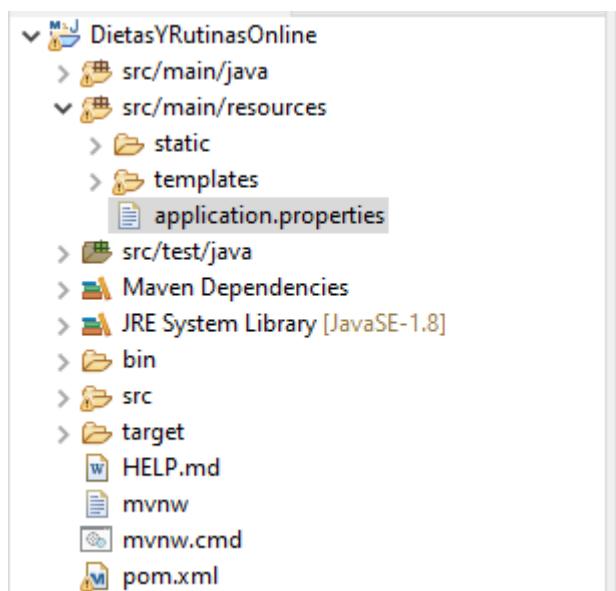
- ✓ En el cuestionario, al lado de Déficit y Volumen te recomiendo colocar un paréntesis con una breve definición de esos conceptos porque no todos saben
- ✓ Me parece bien hecho el de las reuniones de zoom, me parece muy útil
- ✓ El historial de medidas es muy importante, no sé si se pueda transcribir a una tabla (si se puede ver como un gráfico de barras es muy útil y fácil de interpretar para el paciente, solo en caso se pueda)
- ✓ En condición alimentaria en lugar de "nada", colocar "ninguna" es un poco más formal
- ✓ Se ve muy bien hasta ahora, y si creo que se debería incluir los reclinatorios de seguir la dieta o las rutinas ya sea por SMS o correo



## ACCESO A LA PLATAFORMA DE DESARROLLO

- **Acceso:** <https://github.com/GonzaloTafur/ProyectoTecnologicoGrupo3.git>

5. Crear una carpeta y extraer ahí el proyecto descargado en el GitHub. A dentro también hay un Word de manual de instrucciones como respaldo.
6. Exportar la carpeta “bd” que está dentro del repositorio, que incluye el query para la creación de la base de datos y las tablas.
7. Cambiar la ruta del application.properties con el nombre del servidor del equipo que haga la prueba.



8. Ejecutar el proyecto

En Eclipse: Click derecho al proyecto importado -> Run As -> 1 Run on Server -> Finish

## 8.5 Sprint Retrospective:

**Fecha y Hora:** 04 dic 2024 10:00 am - 10:40am

**Participantes:**

- ✓ Product owner
- ✓ Equipo de desarrollo

## PROCEDIMIENTO PARA LA RETROSPECTIVA DEL SPRINT 3

Durante la retrospectiva del Sprint 4, se discutieron los avances del desarrollo de la aplicación, así como las observaciones tanto del Product Owner (profesor) como del nutricionista. A continuación, se detallan las observaciones y sugerencias planteadas por ambas partes, seguidas de los compromisos asumidos y las tareas asignadas para el próximo sprint.

### Observaciones del Product Owner (Profesor)

- Restricciones para agendar reuniones: El profesor destacó que no debe permitirse que un nutriólogo programe dos reuniones para el mismo día y hora, ni que un paciente asista a reuniones con distintos nutriólogos a la misma hora. Solo debe permitirse una reunión por paciente y nutriólogo en el mismo día y hora.

### Sugerencias del Nutricionista

- Definición de Déficit y Volumen en el cuestionario: Se sugirió añadir una breve definición de los términos Déficit y Volumen al lado de estos conceptos en el cuestionario, ya que algunos pacientes podrían no comprenderlos claramente.
- Historial de medidas y visualización gráfica: El nutricionista mencionó la importancia del historial de medidas y sugirió que se presentara en forma de gráfico de barras para que sea más fácil de interpretar para los pacientes. Se indicó que, si es posible, esta visualización puede mejorar la experiencia del usuario.
- Condición alimentaria: Se recomendó cambiar el término "nada" a "ninguna" en la sección de condición alimentaria para dar un tono más formal y profesional.
- Recordatorios de seguimiento: El nutricionista sugirió agregar recordatorios de seguimiento de las dietas y rutinas, a través de SMS o correo electrónico, para motivar a los pacientes a cumplir con su plan.

## ACUERDOS Y COMPROMISOS

Sección tratada	Observación	Lo vamos a tomar SI/NO	Sustentación
Restricciones para reuniones	Implementar que no se permitan reuniones duplicadas para el mismo día y hora, tanto para pacientes como nutriólogos.	SI	Es una funcionalidad esencial para garantizar la integridad y coherencia en el manejo de las reuniones programadas.
Definiciones en cuestionario	Añadir definiciones breves para "Volumen" y "Déficit" en el cuestionario.	SI	Facilitará la comprensión de los términos para los pacientes que no están familiarizados con ellos.
Visualización gráfica del historial	Presentar el historial de medidas como un gráfico de barras para facilitar su interpretación.	NO	Aunque es una buena idea, su implementación requiere más tiempo, por lo que se pospone para una futura actualización.
Cambio en condición alimentaria	Cambiar "nada" a "ninguna" en la sección de condición alimentaria.	SI	Es un ajuste sencillo de realizar, por ello no será complicado realizarlo
Recordatorios de seguimiento	Implementar recordatorios automáticos por SMS o correo para seguimiento de dietas y rutinas.	NO	Aunque útil, su desarrollo requiere más tiempo y recursos, por lo que se pospone para una futura actualización.

## NUEVA HU – HU21 MEJORAS DE LA UI Y EXPERIENCIA DEL USUARIO

Durante la reunión, también se discutió la importancia de mejorar la interfaz de usuario (UI) para garantizar una experiencia más intuitiva y amigable para pacientes y nutricionistas. Se decidió integrar las siguientes mejoras como parte de las actualizaciones de este sprint:

- ✓ **Restricciones claras en reuniones:** Implementar notificaciones y alertas que eviten la programación de reuniones conflictivas, tanto para pacientes como nutricionistas.
- ✓ **Definiciones en el cuestionario inicial:** Añadir explicaciones breves de los conceptos "Volumen" y "Déficit" en el cuestionario de registro.
- ✓ **Optimización visual:** Aplicar una paleta de colores amigable y profesional, junto con un diseño estructurado que facilite la navegación.

### HU21 - Mejoras de la UI y Experiencia del Usuario

Como usuario quiero que la interfaz de la aplicación sea intuitiva, visualmente agradable y fácil de usar, con elementos claramente definidos y accesibles, para que mi experiencia en la plataforma sea satisfactoria y eficiente.

#### Criterios de aceptación:

Escenario 1 – Restricciones de reuniones:

Cuando un paciente confirma su asistencia a una reunión y un nutricionista intenta programar una reunión para la misma fecha, el sistema debe restringir inmediatamente la acción si ya existe otra reunión agendada para el mismo día y hora, evitando conflictos en la programación y asistencia.

Escenario 2 – Definiciones en el cuestionario:

Al completar el cuestionario inicial, los pacientes deben visualizar definiciones breves y claras de los términos "Volumen" (aumentar masa muscular) y "Déficit" (perder grasa), junto a las opciones correspondientes para facilitar su comprensión.

Escenario 3 – Diseño y estructura optimizados:

Cuando los usuarios navegan por la aplicación, la interfaz debe ser clara, organizada y visualmente atractiva, utilizando una paleta de colores amigable que refleje profesionalismo y bienestar, además de contar con elementos como un menú principal, encabezados, iconos y botones destacados que faciliten la navegación y la interacción.

## 8.6 Product Backlog [actualizado]:

Se ha incorporado una nueva historia de usuario al Product Backlog basada en los compromisos asumidos durante la Sprint Retrospective y las mejoras discutidas con el equipo. Esta historia engloba ajustes a la interfaz de usuario y la experiencia del usuario, además de incluir restricciones funcionales necesarias.

ID	HISTORIA DE USUARIO	PRIORIDAD	ESTIMACION	DEPENDENCIA	SPRINT
HU21	Mejoras de la UI y Experiencia del Usuario	740	7 días	HU previas	5

## PROCEDIMIENTO SEGUIDO PARA LA ACTUALIZACIÓN:

- Revisión de resultados y ajustes:** Se evaluaron las tareas pendientes y las recomendaciones del profesor, destacando la necesidad de mejoras en la interfaz y experiencia del usuario para el próximo sprint.
- Establecimiento de estimaciones:** La nueva historia de usuario (HU21) se estimó en 7 días, considerando la complejidad de los cambios en la UI y las pruebas requeridas.

## 9. SPRINT 5: MVP

### 9.1 Sprint Backlog:

#### APLICACIÓN DE RUTINAS Y DIETAS

- Fecha de Inicio del Sprint:** 5 de diciembre del 2024.
- Fecha de Fin del Sprint:** 17 de diciembre del 2024.
- Duración del Sprint:** 2 semanas

#### HISTORIAS DE USUARIOS

Este Sprint corresponde del 5/12/24 al 17/12/24, incluye la presentación de todas las historias de usuario desarrolladas en los sprints anteriores. Además, se abordará la última historia de usuario planificada para este Sprint 5, la HU21.

#### INCREMENTO PREVIO (SPRINT 1)

HU01 – Registrar usuario
Como usuario, quiero registrarme en la aplicación de rutinas y dietas ingresando mis datos personales para acceder a las funcionalidades de la plataforma.
<b>Criterios de aceptación:</b>

## Escenario 1

La interfaz de registro deberá incluir campos para nombre de usuario, contraseña, nombre, apellidos, correo electrónico, sexo y documento de identidad. El sistema deberá redirigir al usuario según seleccione ser paciente o nutriólogo en el proceso de registro.

El formulario deberá conectarse al servidor para almacenar la información del usuario. También deberán realizarse pruebas de usabilidad y funcionalidad del formulario para garantizar una experiencia de usuario adecuada.

## HU03 – Acceso para usuarios

Como usuario, quiero iniciar sesión en la aplicación ingresando mi correo electrónico y contraseña, para acceder a mis datos y funcionalidades personalizadas según mi tipo de usuario

### Criterios de aceptación:

## Escenario 1

La interfaz de inicio de sesión deberá incluir campos para el correo electrónico y la contraseña. El sistema deberá mostrar vistas personalizadas según el tipo de usuario (paciente o nutriólogo) que haya iniciado sesión. El formulario deberá conectarse al servidor para validar la información del usuario.

## HU02 – Cuestionario para pacientes nuevos

Como administrador del sistema, quiero que los nuevos pacientes respondan unas preguntas para visualizar sus resultados.

### Criterios de aceptación:

## Escenario 1 – Cuestionario

Luego de que el paciente termine de completar el formulario de registro, el sistema le dará una serie de preguntas que debe responder (Cuánto pesa, la última vez que hizo dieta, cada cuánto hace ejercicios, si ha sido hospitalizado y de qué, si es alérgico a x comida, etc.)

## HU04 – Menú de Inicio

Como usuario, quiero que en el menú de inicio se me proporcione acceso a las rutinas, dietas, el calendario, entre otras funcionalidades, para así poder navegar y ver detalles.

**Criterios de aceptación:**

Escenario 1 - Menú de inicio

Tras registrarse o iniciar sesión, el paciente será llevado al menú de inicio, donde encontrará una barra de navegación superior con el ícono de usuario y sus notificaciones, además de un banner de bienvenida. Bajo esta barra, habrá opciones para acceder a rutinas, dietas, calendario y otras funcionalidades necesarias.

## INCREMENTO PREVIO (SPRINT 2)

### HU05 - Insertar rutinas personalizadas

Como nutriólogo, quiero crear y personalizar rutinas de ejercicios agregando descripciones específicas para recomendarlas a mis pacientes registrados.

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear rutinas, se le redirige a una ventana donde puede buscar ejercicios en la base de datos, seleccionarlos y organizarlos en una rutina personalizada. Además, el nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar la rutina según los objetivos específicos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la rutina y recomendarla a sus pacientes registrados.

### HU06 – Insertar dietas personalizadas

Como nutriólogo, quiero crear y personalizar dietas agregando detalles específicos para recomendarlas a mis pacientes registrados.

**Criterios de aceptación:**

Escenario 1

Cuando el nutriólogo accede a la sección de crear dietas, se le redirige a una ventana donde puede ingresar y organizar planes de alimentación personalizados. El nutriólogo puede agregar descripciones, objetivos, duración y nivel de dificultad, y categorizar las dietas según objetivos como volumen, recomposición corporal o déficit calórico. Una vez completada, puede guardar la dieta y recomendarla a sus pacientes registrados.

## HU07 - Editar rutinas y dietas

Como nutriólogo, quiero poder editar los detalles de las dietas y rutinas para asegurarme de que la información esté actualizada y sea precisa.

### Criterios de aceptación:

#### Escenario 1 - Acción editar rutinas

Cuando el nutriólogo selecciona la opción para editar una rutina que él creó, podrá modificar sus datos, incluyendo el tipo de rutina, la categoría, su descripción y los ejercicios incluidos. Estas modificaciones permitirán mantener la base de datos actualizada.

#### Escenario 2 - Acción editar dietas

Cuando el nutriólogo selecciona la opción para editar una dieta que él creó, podrá modificar sus datos, incluyendo el tipo de dieta, la descripción, y los alimentos incluidos en esta dieta. Estas modificaciones permitirán mantener la base de datos actualizada.

## HU08 – Ver detalle de una rutina y dieta

Como paciente, quiero visualizar a detalle la dieta y rutina dentro de una lista para tenerla guardarla sea en calendario o por texto.

### Criterios de aceptación:

#### Escenario 1 – Ver detalle de una rutina

Cuando el usuario selecciona la opción para ver el detalle de una rutina, podrá visualizar sus datos, incluyendo el tipo de rutina, el nutriólogo quien lo creó, la categoría, su descripción y una lista de los ejercicios incluidos. También unos botones para descargarlo o guardarlo a tu calendario.

## Escenario 2 - Ver detalle de una dieta

Cuando el usuario selecciona la opción para ver el detalle de una dieta, podrá visualizar sus datos, incluyendo el tipo de dieta, el nutriólogo quien lo creó, su descripción y una lista de alimentos que debería consumir. También unos botones para descargarlo o guardarlo a tu calendario.

### HU09 – Ver lista de alimentos

Como nutriólogo, quiero ver una lista de alimentos disponibles en la base de datos para poder consultarlos y utilizarlos en la creación de dietas personalizadas para mis pacientes.

#### Criterios de aceptación:

##### Escenario 1

Cuando el nutriólogo accede a la sección de alimentos, se le muestra una lista de alimentos que incluye el nombre, tipo (desayuno, almuerzo, cena, merienda), y sus nutrientes principales (proteínas, grasas, carbohidratos). El nutriólogo puede buscar alimentos específicos, filtrarlos por tipo o por nutrientes, y seleccionar aquellos que desea incluir en las dietas de sus pacientes.

### HU12 – Ver lista de ejercicios

Como nutriólogo, quiero ver una lista de ejercicios en la base de datos para poder consultarlos y utilizarlos en la creación de rutinas personalizadas para mis pacientes.

#### Criterios de aceptación:

##### Escenario 1

Cuando el nutriólogo accede a la sección de ejercicios, se le muestra una lista que incluye el nombre del ejercicio, el grupo muscular al que pertenece, el tipo de ejercicio y una breve descripción. El nutriólogo puede buscar ejercicios específicos, filtrarlos por grupo muscular o tipo de ejercicio, y seleccionar aquellos que desea incluir en las rutinas de sus pacientes.

## INCREMENTO PREVIO (SPRINT 3)

## HU10 – Perfil de nutriólogos

Como nutriólogo, quiero tener acceso a un perfil de usuario para insertar información visible y crear mis rutinas y dietas para que más pacientes puedan visitarme y ponerse en contacto conmigo.

### Criterios de aceptación:

Escenario 1 – Perfil de nutriólogo

Cuando el nutriólogo entre a su perfil, podrá visualizar qué rutinas y dietas ha creado junto a un botón para crear otras, y una opción de editar perfil para añadir sus contactos, lugar en donde trabaja y descripción.

## HU11 – Perfil de pacientes

Como paciente, quiero tener acceso un perfil de usuario para visualizar mi actividad y gestionar mi información visible.

### Criterios de aceptación:

Escenario 1 – Perfil de paciente

Cuando el paciente ingrese su perfil, podrá visualizar qué rutina o dieta ha tiene guardado, junto a una opción de editar perfil para añadir su descripción y gestionar la información solicitada en el cuestionario.

## HU13 – Ajustes de Usuario

Como usuario, quiero hacerles ajustes de seguridad a mí cuenta para no descuidarla en caso de que se me olvide la contraseña.

### Criterios de aceptación:

Escenario 1 – Ajustes de usuario

Cuando el usuario está en la opción de ajustes, tendrá una opción para actualizar contraseña, y también una para cerrar sesión.

## HU14 - Personalizar calendario de paciente

Como paciente, quiero gestionar un calendario personal para organizar mis rutinas de entrenamiento proporcionadas por el nutriólogo.

### Criterios de aceptación:

Escenario 1

El paciente llega a una opción donde puede acceder a su calendario. El paciente tendrá acceso a un calendario donde podrá agregar sus rutinas de ejercicios para cada día de la semana. Podrá asignar los días de entrenamiento, seleccionar qué rutinas realizará cada día, y programar los días de descanso. Esto le permitirá visualizar y organizar su plan de entrenamiento semanal.

## INCREMENTO PREVIO (SPRINT 4)

### HU16.1 – Notificaciones para el Usuario (Actualizada)

Como paciente o nutricionista, quiero recibir notificaciones sobre actividades programadas, reuniones y consejos del día en base a mis objetivos, para mantenerme al día con mi plan de salud y recibir recordatorios importantes.

### Criterios de aceptación:

Escenario 1 – Notificaciones:

Cuando el usuario ingresa a la sección de notificaciones en la página, verá nuevas alertas sobre actividades programadas, reuniones con el nutricionista, consejos diarios de salud.

### HU17.1 – Registro semanal de medidas y seguimiento de progreso

Como paciente, quiero ingresar mis medidas semanales (peso, perímetro de cintura y bíceps), para monitorear mi progreso en función de mis objetivos deportivos.

**Criterios de aceptación:**

Escenario 1

El paciente podrá ingresar sus medidas cada semana para monitorear su progreso. Estos datos se guardarán en su perfil para visualizar el cambio a lo largo del tiempo. Una vez ingresadas las medidas, el usuario no podrá volver a registrar datos hasta pasados 7 días. En caso de intentar ingresar nuevamente, el sistema le mostrará una alerta indicando que debe esperar hasta que transcurra una semana desde su última entrada.

## HU19 – Recomendación Inicial de Dietas y Rutinas Personalizadas

Como paciente recién registrado, quiero que el sistema me recomiende dietas y rutinas adecuadas a mis objetivos (déficit calórico o volumen) y experiencia física previa, para poder iniciar mi programa de forma segura y efectiva, considerando mis capacidades y posibles restricciones alimenticias.

**Criterios de aceptación:**

Escenario 1

El sistema recomendará dietas y rutinas basadas en el cuestionario realizado después del registro.

## HU20.1 – Programar reuniones con el nutriólogo

Como nutriólogo, quiero gestionar reuniones virtuales con mis pacientes, para realizar evaluaciones y brindar seguimiento personalizado.

**Criterios de aceptación:**

Escenario 1

El nutriólogo podrá programar reuniones grupales o individuales con temas específicos (por ejemplo, "Dietas para principiantes"). Estas reuniones tendrán horarios predefinidos, y el sistema notificará al nutriólogo sobre las confirmaciones de asistencia de los pacientes. Además, podrá visualizar una lista de los asistentes junto con un resumen de su perfil, que incluirá sus dietas, seguimiento de medidas y progreso registrado.

## INCREMENTO ACTUAL (SPRINT 5)

### PROCEDIMIENTO PARA LA ORGANIZACIÓN Y CONSOLIDACIÓN DEL BACKLOG

Durante la planificación del Sprint 5, realizamos una evaluación final de la aplicación con el objetivo de entregar un producto funcional, intuitivo y visualmente optimizado. Este sprint se centró en corregir carencias identificadas, resolver errores técnicos menores y mejorar la experiencia del usuario a nivel estético y funcional.

#### a) Análisis de las Carencias y Oportunidades de Mejora

En revisiones anteriores y pruebas de usabilidad, se identificaron los siguientes puntos críticos:

- **Interfaz poco intuitiva:** Algunos elementos visuales y de navegación no estaban claramente definidos, afectando la experiencia del usuario.
- **Restricciones no implementadas:** La programación de reuniones no contaba con un sistema efectivo para evitar conflictos de horario.
- **Falta de claridad en el cuestionario inicial:** Los términos clave como "Volumen" y "Déficit" no estaban explicados, lo cual generaba confusión en usuarios nuevos.

#### b) Acuerdos para la Implementación de Mejoras

Con base en los hallazgos anteriores, se decidió incluir una última historia de usuario (HU21) que aborda tres áreas específicas:

- Restricciones en la programación de reuniones, asegurando que no haya conflictos en fechas y horarios.
- Definiciones claras en el cuestionario inicial, facilitando la comprensión de los objetivos "Volumen" y "Déficit".
- Optimización del diseño y estructura visual, implementando mejoras estéticas y funcionales para lograr una interfaz organizada, amigable y profesional.

#### HU21 - Mejoras de la UI y Experiencia del Usuario

Como usuario quiero que la interfaz de la aplicación sea intuitiva, visualmente agradable y fácil de usar, con elementos claramente definidos y accesibles, para que mi experiencia en la plataforma sea satisfactoria y eficiente.

#### Criterios de aceptación:

## Escenario 1 – Restricciones de reuniones:

Cuando un paciente confirma su asistencia a una reunión y un nutricionista intenta programar una reunión para la misma fecha, el sistema debe restringir inmediatamente la acción si ya existe otra reunión agendada para el mismo día y hora, evitando conflictos en la programación y asistencia.

## Escenario 2 – Definiciones en el cuestionario:

Al completar el cuestionario inicial, los pacientes deben visualizar definiciones breves y claras de los términos "Volumen" (aumentar masa muscular) y "Déficit" (perder grasa), junto a las opciones correspondientes para facilitar su comprensión.

## Escenario 3 – Diseño y estructura optimizados:

Cuando los usuarios navegan por la aplicación, la interfaz debe ser clara, organizada y visualmente atractiva, utilizando una paleta de colores amigable que refleje profesionalismo y bienestar, además de contar con elementos como un menú principal, encabezados, iconos y botones destacados que faciliten la navegación y la interacción.

## 9.2 Sprint Planning:

### AGENDA TRATADA DURANTE LA CEREMONIA DE SPRINT PLANNING:

#### a) Revisión de la última Historia de Usuario (HU21):

- Se discutió la HU21 – Mejoras de la UI y Experiencia del Usuario, enfocada en optimizar la interfaz y mejorar la experiencia del usuario final.
- Se identificaron las tareas específicas de la HU, como:
  - ✓ Restricciones en la programación de reuniones.
  - ✓ Definiciones claras de términos en el cuestionario inicial.
  - ✓ Optimización del diseño visual y la estructura de la aplicación.
- Además, se planificó la implementación de la encriptación de contraseñas como una mejora técnica adicional para garantizar la seguridad de los datos de los usuarios.

#### b) Asignación de responsabilidades:

- Gonzalo Tafur Bermúdez fue asignado como el único responsable de la implementación de la HU21 y la encriptación de contraseñas. Su rol incluyó tareas técnicas, de desarrollo y ajustes finales de la aplicación.
- Jasmin se encargó de verificar que los criterios de aceptación y las mejoras visuales estuvieran alineadas con las expectativas de los usuarios y organizó las reuniones de seguimiento.

#### c) Determinación de dependencias y predecesoras:

- La HU21 no tuvo dependencias directas de HU anteriores, pero sí requirió que las funcionalidades básicas estuvieran completamente operativas para realizar ajustes finales.
- La implementación de la encriptación de contraseñas dependió del correcto funcionamiento del módulo de inicio de sesión y registro de usuarios.

## d) Aprobación del Sprint Backlog:

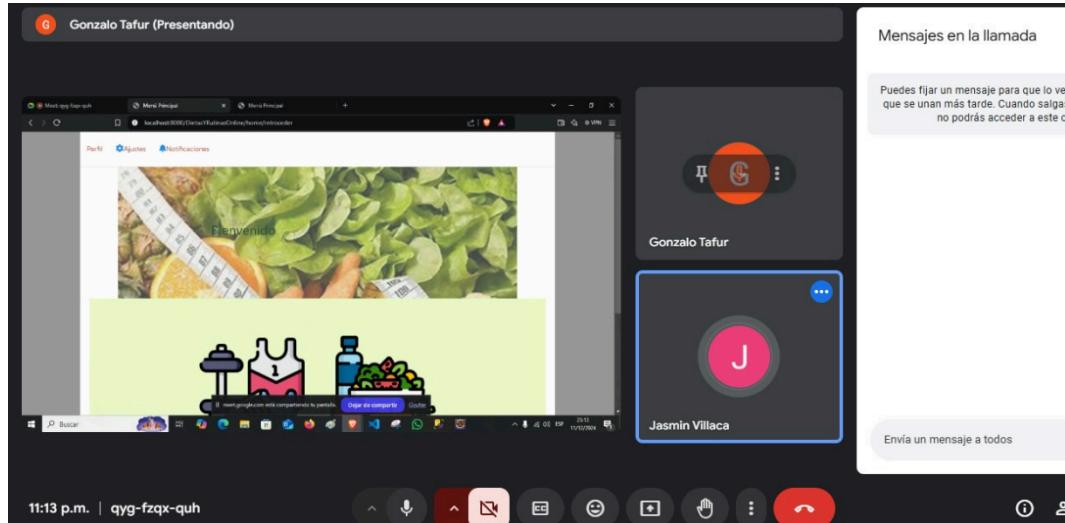
- Se llegó a un acuerdo sobre la realización de la HU21 y la tarea de encriptación de contraseñas.
- Se estimó que ambas tareas se completarían dentro del Sprint 5, consolidando así el producto final en su versión MVP.

## PROCEDIMIENTO SEGUIDO PARA LA CEREMONIA:

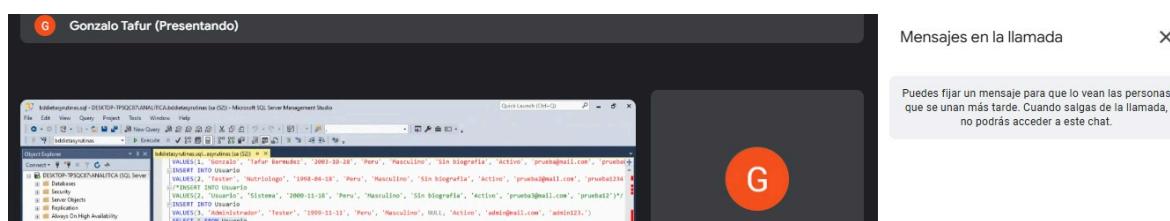
- FECHA: La ceremonia de planificación del Sprint 5 se llevó a cabo el 11/12/2024.
- PARTICIPANTES: Incluyó a todo el equipo
- RESULTADOS DE LA CEREMONIA:

- ✓ **Sprint Backlog:** Se seleccionó la HU21 y la tarea adicional de encriptación de contraseñas.
- ✓ **Responsabilidad:** Gonzalo asumió la ejecución completa de las tareas.
- ✓ **Evidencia:** Se verificó que las mejoras técnicas y visuales se alinearan con las necesidades del usuario final, asegurando una experiencia funcional, intuitiva y segura.

## EVIDENCIA DE LA REUNION



## EVIDENCIA DE LAS PRUEBAS DE FUNCIONAMIENTO DE ENCRIPCIONACION



Tareas/Tareas no técnicas	Estimación	Asignación de Trabajo
<b>HU21 - Mejoras de la UI y Experiencia del Usuario</b>		
Implementar restricciones para la programación de reuniones cuando existe una reunión en conflicto	1 hora	Tafur Bermúdez, Gonzalo
Cambiar la navegabilidad de las listas y que estas incluyan una barra despegable.	2 horas	Tafur Bermúdez, Gonzalo
Optimizar el diseño y estructura de la aplicación (colores, menús, iconos, botones, organización)	2 horas	Tafur Bermúdez, Gonzalo
Realizar pruebas de experiencia de usuario y ajustes finales en la interfaz	1 hora	Tafur Bermúdez, Gonzalo
Implementar la encriptación de contraseñas para reforzar la seguridad del sistema	3 horas	Tafur Bermúdez, Gonzalo

### KANBAN BOARD JIRA (Contraseña y correo para el acceso)

<https://jasmin.atlassian.net/jira/software/projects/KAN/boards/1>  
[proyecdietasyrutinas@gmail.com](mailto:proyecdietasyrutinas@gmail.com)

DyROtafvill

### 9.3 Daily Scrum:

Durante el Sprint 5, el equipo de desarrollo se enfocó en implementar las mejoras finales de la aplicación, con especial atención a la HU21 (Mejoras de la UI y Experiencia del Usuario) y la implementación de encriptación de contraseñas utilizando Spring Security. Las reuniones diarias, con

una duración de 40 minutos a 1 hora, sirvieron para revisar los avances, asignar tareas y garantizar el cumplimiento de los criterios de aceptación establecidos.

## ¿QUÉ HICE AYER?

Ayer, el equipo completó las tareas relacionadas con la HU21, enfocándose en optimizar la interfaz de usuario y mejorar la experiencia general. Se implementaron las siguientes mejoras:

- Restricciones para evitar conflictos en la programación de reuniones.
- Visualización clara de las definiciones de Volumen y Déficit en el cuestionario inicial.
- Optimización del diseño de la aplicación (colores, menús, botones e iconos) para una mejor navegación.

Además, se trabajó en la implementación de la encriptación de contraseñas mediante Spring Security, cumpliendo con una solicitud previa del profesor para reforzar la seguridad del sistema.

## ¿QUÉ HARÉ HOY?

Hoy, el equipo se enfocará en realizar pruebas exhaustivas para asegurar que las mejoras de la interfaz de usuario funcionen correctamente en todos los flujos de la aplicación. Además, se validará la lógica de las restricciones en la programación de reuniones para garantizar que no existan conflictos. Se llevará a cabo la verificación de la encriptación de contraseñas, asegurando que cumpla con los estándares de seguridad esperados. Finalmente, se realizarán ajustes menores en los detalles visuales para ofrecer una experiencia de usuario fluida y profesional.

## ¿QUÉ IMPEDIMENTOS TENGO?

No se presentaron inconvenientes durante la implementación de las tareas del Sprint 5. La encriptación de contraseñas, aunque fue una tarea pendiente solicitada previamente por el profesor, se implementó sin contratiempos utilizando Spring Security como dependencia principal.

## EVIDENCIA DE LA REUNIÓN #1 (JUEVES 14/11)

- DURACIÓN: 22:30 am – 23:13 pm
- PARTICIPANTES: Jasmin Xiomi Villaca Mamani, Tafur Bermúdez Gonzalo
- TEMAS TRATADOS:

### Actualización y mejoras de diseño:

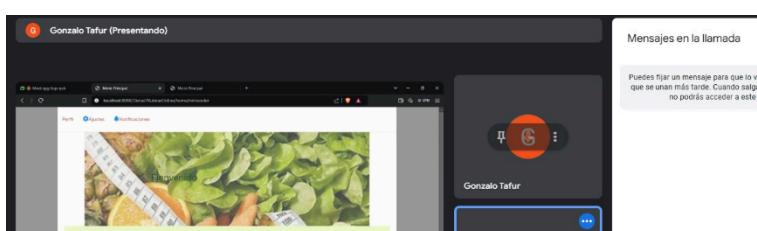
Se revisaron propuestas para optimizar el diseño de la aplicación, evaluando mejoras en los iconos, paletas de colores y detalles visuales. El objetivo fue mejorar la experiencia de usuario, asegurando una interfaz más intuitiva y profesional.

### Implementación de la encriptación de contraseñas con Spring Security:

Se discutió y planificó la implementación de la encriptación de contraseñas utilizando la dependencia Spring Security. Esta tarea había sido solicitada previamente por el profesor y se destacó su importancia para garantizar la seguridad y protección de los datos de los usuarios.

## RESULTADOS DE LA REUNION

### EVIDENCIA DE LA REUNION



## ENCRIPCION

src/main/java

```

63      <!-- https://mvnrepository.com/artifact/org.springframework.security/spring-security-core -->
64      <dependency>
65          <groupId>org.springframework.boot</groupId>
66          <artifactId>spring-boot-starter-security</artifactId>
67          <version>2.7.18</version>
68      </dependency>
69
70      <dependency>
71          <groupId>org.springframework.security</groupId>
72          <artifactId>spring-security-crypto</artifactId>
73      </dependency>
74
75

```

src/main/resources

```

3@ import org.springframework.context.annotation.Bean;@
4
5  @Configuration
6  public class SecurityConfig {
7
8      @Bean
9      public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
10          http
11              .authorizeRequests()
12                  .anyRequest().permitAll()
13                  .and()
14                      .csrf().disable();
15          return http.build();
16      }
17
18      @Bean
19      public PasswordEncoder passwordEncoder() {
20          return new BCryptPasswordEncoder();
21      }
22
23
24
25
26
27
28
29
30

```

Results											Messages	
o	idtipousu	nomusuario	apeusuario	fechanacimiento	nacionalidad	sexo	biografia	estusuario	correo	password		
1	TY	JGW		2003-03-01	Ecuador	Masculino	NULL	Activo	prueba524@mail.com	\$2a\$10\$XoQ/p8JNmpb60E9uHm0/XJwzie18FWj2t490COVYQLZ...		
1	Tests	EQW		2002-07-31	Paraguay	Masculino	NULL	Activo	ggg42@mail.com	\$2a\$10\$wGUjY9z5qcorEdO/0/Pp.nFvkZsi0xVDK9OU0zuLflb...		
2	HDRT	VV		2009-04-02	NULL	Femenino	Nutriólogo con 10 años...	Activo	test221@mail.com	\$2a\$10\$/Q/32R8XB852elBu3Bn80bhEl3jq9Dow.PHOAu.0x655...		
NULL	AJ	RT		2000-07-19	Otro	Masculino	NULL	Activo	prueba532@mail.com	\$2a\$10\$3xBMLeVZAqBseEyLOCE30BSg15WHSwND7b7gnXHosi...		
NULL	YE	HF		2001-03-09	Otro	Masculino	NULL	Activo	64ghj@mail.com	\$2a\$10\$2TcpJceOUpq6/KgpOpEtekC/WZG4/QWWlj5LevrKqJC...		
2	GD	HTB		2003-03-07	Otro	Masculino	NULL	Activo	tttw@mail.com	\$2a\$10\$NjsUawTpgl5fwca7SNOpto905M2L/v/k9EHGOhlwS.3Y...		
1	RGF	FVBN		2002-07-17	España	Masculino	NULL	Activo	test33111@mail.com	\$2a\$10\$MAQlaDe4yMUWlbRPYO7hWuZTN4.HnyeWW55FaE1R...		
1	HDER	GBD		2004-03-04	Chile	Femenino	NULL	Activo	test2143@mail.com	\$2a\$10\$Z.wQtBhncNg3wBlocCdseWfio2jgUhzloCPM.oQBizvCLG...		
1	Maria Ana	Lopez Gutierrez		2003-02-17	Perú	Femenino	NULL	Activo	maria.lopez@gmail...	\$2a\$10\$.wE1UoW7wZjD1jrfZXq5jPOV0DeXQPdil2qcLzEOgXunk...		
2	Alejandro	Ramirez Lopez		1995-08-09	NULL	Masculino	Experiencia Consultori...	Activo	alejandro.ramirez@...	\$2a\$10\$EC2/PlsnzebpRv90UceJ7.uATkNsQCzGIL3XxvVm8km0v...		
1	Lucia	Mamani Flores		2006-04-12	Perú	Femenino	NULL	Activo	lucia.mamani@gmail...	\$2a\$10\$AhQEMooTN.bf(0MqJ.YbuYcALiZfrReZTRFinRpckKG...		
2	Aldair	Gonzales San...		1994-09-01	NULL	Masculino	15 años de experiencia...	Activo	aldair.gonzalez@gm...	\$2a\$10\$Z3M.vVNJuFc15pWeCknfuP4XbMbjv2FEBSWBXCO0eSe...		
1	Luis	Rodriguez Go...		2001-07-13	Perú	Masculino	NULL	Activo	luisrod41248@gmail...	\$2a\$10\$DPKnSFHCe2KIK3yiv/oSebJM3nxLU57oh9FLUpY6VUTa...		

## EVIDENCIA DE LA REUNIÓN #2 (MARTES 11/12)

- DURACIÓN: 13:00 – 13:54: horas
- PARTICIPANTES: Tafur Bermúdez Gonzalo, Nutriólogo
- TEMAS TRATADOS:

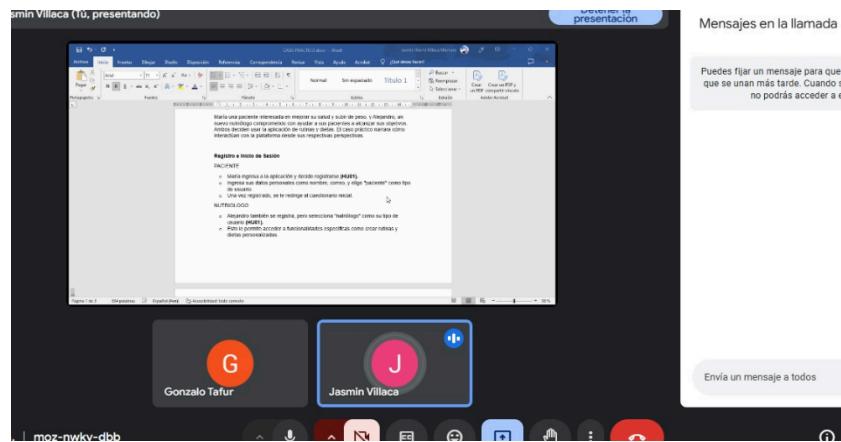
### Revisión de diseño y funcionalidades:

- ✓ Se revisaron los cambios en el diseño de la plataforma, incluyendo una nueva barra deslizadora para facilitar la visualización de rutinas y dietas. Además, se discutieron ajustes en la paleta de colores para mejorar la usabilidad y la estética de la aplicación.
- ✓ También se abordó la creación de un caso práctico para evidenciar el correcto funcionamiento de cada funcionalidad. Este caso será grabado en un video demostrativo

que mostrará cómo las funcionalidades (como el registro de medidas, notificaciones y reuniones) operan de manera conjunta, sirviendo como prueba de que todo está funcionando correctamente.

## RESULTADOS DE LA REUNION

### CASO PRACTICO CREACION



### 9.4 Sprint Review:

## FUNCIONALIDADES COMPLETAS DE LA APLICACIÓN (MVP)

### o INTERFAZ DE REGISTRO PARA USUARIOS

En la siguiente imagen, se puede visualizar la interfaz de registro diseñada para los usuarios. Esta sección permite a los nuevos usuarios ingresar sus credenciales y datos personales, como el rol o tipo de usuario (paciente y nutriólogo) nombres, apellidos, correo electrónico, fecha de nacimiento, nacionalidad y sexo. Al completar este formulario, ya tendrán acceso a la aplicación para comenzar su recorrido hacia un estilo de vida más saludable.

## Regístrate

¿Eres paciente o nutriólogo?

Paciente

Nutriólogo

Nombres

Apellidos

Fecha nacimiento  CALENDAR

Nacionalidad

Sexo

Correo Electronico

Contraseña

**Enviar**

### ○ CUESTIONARIO PARA PACIENTES NUEVOS

En esta imagen, se presenta el cuestionario que los nuevos pacientes deben completar tras registrarse. Este cuestionario recopila información clave sobre la salud del paciente, incluyendo peso, historial de dietas, frecuencia de ejercicio y alergias. Los datos recolectados son esenciales para personalizar las rutinas y dietas, garantizando una atención adecuada y efectiva. En caso que no sabe o no se acuerda de su información, puede saltar el cuestionario e ir al menú, pero puede llenarlo en otro momento actualizándolo en su perfil o por notificaciones.

## Cuestionario

Este cuestionario nos ayudara a saber más de ti.  
También puede editarlo para hacer seguimiento.

¿Con que frecuencia haces ejercicios?

30 minutos a diario

¿Tienes una condición alimenticia?

Ninguna

¿Tienes un objetivo?

Volumen

*Si aún no sabe sus medidas, puede saltarse esta parte y luego llenarlas en otro momento.*

Peso corporal

72

Estatura

1.74

Perímetro de cintura

Perímetro de cadera

Perímetro de muslo

Perímetro de brazo/bicep

**Grabar**

### o INTERFAZ DE INICIO DE SESIÓN PARA USUARIOS

En la siguiente imagen, se puede visualizar la interfaz de inicio de sesión diseñada para los usuarios. Esta sección permite a los usuarios ingresar sus credenciales de correo electrónico y contraseña. Al completar esto, los usuarios pueden acceder a la aplicación y comenzar su recorrido hacia un estilo de vida más saludable.

## Iniciar Sesión

Correo Electronico

Contraseña

**Ingresar**

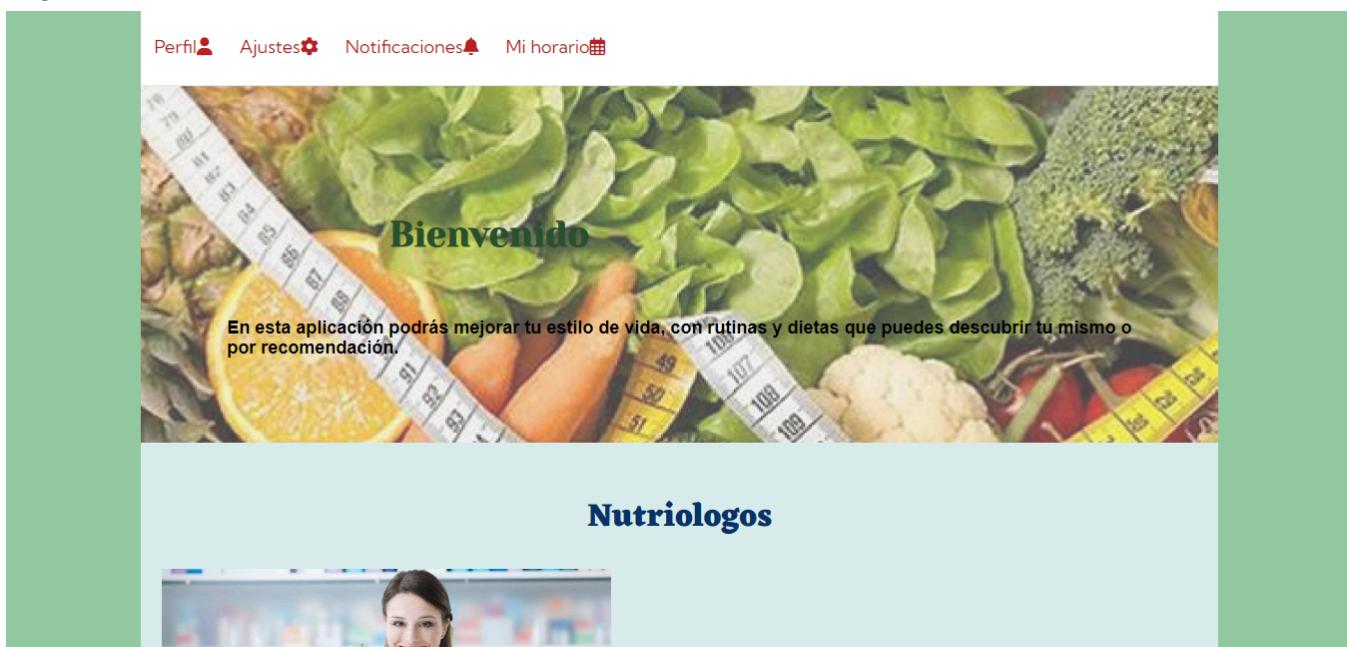
[¿Eres nuevo aquí? Regístrate](#)

2024. Todos los derechos reservados.

### ○ MENÚ DE INICIO DE LA APLICACIÓN

Aquí se muestra la interfaz del menú de inicio de la aplicación. Después de registrarse o iniciar sesión, los usuarios encontrarán una barra de navegación superior con acceso a las diferentes funcionalidades, incluyendo rutinas, dietas y calendario. Esta organización permite a los pacientes navegar de manera sencilla y eficiente, mejorando su experiencia en la aplicación.

#### PACIENTE



## NUTRÍOLOGO



### ○ INSERCIÓN DE RUTINAS PERSONALIZADAS

En la siguiente imagen, se visualiza la interfaz donde el nutriólogo puede crear rutinas personalizadas. Los nutriólogos pueden seleccionar ejercicios de la base de datos, categorizarlos según el objetivo (volumen o déficit calórico), añadir descripciones específicas, definir el nivel de dificultad y duración. Además, de poder añadir recomendaciones en la descripción para los pacientes sobre los tiempos de descanso, el peso sugerido y otras pautas importantes para la correcta realización de las rutinas.

## Insertar nueva rutina

Nombre de rutina

Tipo de rutina

Rutina de Deficit

Parte del cuerpo

Tren Superior

Nivel

Principiante

Descripción

Ejercicios

Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción	Seleccionar
Remo con mancuerna	Espalda	Tren Superior	4	12 a 15	Con una mancuerna en una mano y apoyando la otra en un banco, lleva la mancuerna hacia el abdomen. Controla la bajada y mantén la espalda recta.	<input type="checkbox"/>
Curl con mancuernas ligeras	Bíceps	Tren Superior	4	12 a 15	Con las palmas hacia arriba, flexiona los codos levantando las mancuernas hacia los hombros. Usa un peso ligero y repeticiones altas para mayor resistencia.	<input checked="" type="checkbox"/>
Fondos en paralelas	Tríceps	Tren Superior	4	12 a 15	Con las manos en las barras paralelas, baja el cuerpo flexionando los codos y sube estirando los brazos. Mantén las piernas extendidas.	<input type="checkbox"/>

## o INSERCIÓN DE DIETAS PERSONALIZADAS

En esta imagen, se presenta el cuestionario que los nuevos pacientes deben completar tras registrarse. Este cuestionario recopila información clave sobre la salud del paciente, incluyendo peso, historial de dietas, frecuencia de ejercicio y alergias. Los datos recolectados son esenciales para personalizar las rutinas y dietas, garantizando una atención adecuada y efectiva. En caso que no sabe o no se acuerda de su información, puede saltar el cuestionario e ir al menú, pero puede llenarlo en otro momento actualizándose en su perfil o por notificaciones.

## Insertar nueva dieta

Nombre

Objetivo

Deficit

Descripción

¿Qué alimento incluye?

Alimento	Nutrientes	Tipo	Descripción	Seleccionar
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.	<input checked="" type="checkbox"/>
Pan con Palta y Huevo	20g de proteína, 50g de carbohidratos, 18g de grasas	Desayuno	Pan integral con palta (50g) y huevo frito o cocido (2 unidades). Rico en grasas saludables y proteínas.	<input type="checkbox"/>
Arroz con Pollo	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias y aceite vegetal.	<input type="checkbox"/>
Lentejas con Carne	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteínas para volumen.	<input type="checkbox"/>

## o VER DETALLE DE UNA RUTINA Y DIETA

Esta sección permite a los pacientes visualizar a detalle las rutinas y dietas asignadas o recomendadas. En el caso de las rutinas, los usuarios pueden ver si están enfocadas en volumen o déficit, el grupo muscular que trabajan (tren superior, tren inferior o cuerpo completo), y además, se incluye un apartado de recomendaciones específicas para la ejecución: tiempos de descanso entre series, cargas sugeridas, y el número de repeticiones. Para las dietas, se muestra la lista de alimentos, las porciones y un botón para añadir la dieta a su calendario.

## Definición Inferior 1

Tipo de Rutina: Deficit

Descripción: Esta rutina está enfocada en la quema de grasa y la tonificación. Comienza con el Hip Thrust para activar los glúteos. Descansa de 2 minutos entre series. Utiliza pesos ligeros y enfócate en un rango de repeticiones más alto (10 -15). La combinación de Hip Thrust y zancadas es excelente para trabajar los glúteos y las piernas, lo que resulta en una rutina efectiva para principiantes que buscan un déficit calórico.

Nivel: Principiante

Creado por: [HDRT.VV](#)

### Ejercicios

Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción
Hip thrust repeticiones altas	Glúteos	Tren Inferior	4	12 a 15	Eleva las caderas contrayendo los glúteos, pero con un peso moderado y más repeticiones para aumentar la quema calórica.
Zancadas hacia Atrás	Glúteos	Tren Inferior	4	10 a 12	Da un paso hacia atrás, baja en zancada hasta que la rodilla casi toca el suelo, cambia de pierna.
Elevaciones de Talón con Disco de Peso	Pantorrillas	Tren Inferior	4	10 a 12	Coloca un disco de peso sobre los muslos, justo encima de las rodillas, mientras estás sentado en un banco. Eleva los talones lo más alto posible, mantén un segundo y baja lentamente.
Curl de piernas en		Tren	2	8 a 10	Siéntete en la máquina con las piernas detrás del

## Nutrición Vital

Objetivo: Volumen

Descripción: Esta dieta está pensada para quienes buscan ganar masa muscular de forma saludable. Comienza el día con un batido de plátano y avena y una tortilla de huevo con queso fresco y espinacas , brindando energía y proteínas. Una media mañana, disfruta de yogur con semillas de chía y frutas , que aporta fibra y grasas saludables.

Creado por: [HDRT.VV](#)

### Condición

Carne

### Alimentos incluidos

Alimento	Nutrientes	Tipo	Descripción
Batido de Plátano y Avena	15g de proteína, 70g de carbohidratos, 10g de grasas	Desayuno	Batido con leche (250 ml), plátano (1 unidad), avena (50g). Aporta energía y favorece el crecimiento muscular.
Lentejas con Carne	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteínas para volumen.
Tortilla de Huevo con Queso Fresco y Espinacas	15 g de proteína, 8 g de carbohidratos, 10 g de grasas	Desayuno	Tortilla hecha con 3 huevos, 50 g de queso fresco y 100 g de espinacas salteadas. Alta en proteínas y nutrientes.
Ensalada de Monastros	22 g de proteína, 20 g de		Mezcla de 100 g de lentejas y 100 g de garbanzos con 150 g de

### VER LISTA DE EJERCICIOS

La lista de ejercicios disponible para los nutriólogos contiene el nombre del ejercicio, el grupo muscular, y una breve descripción de cómo realizarlo. Esta sección está optimizada para que los nutriólogos puedan buscar fácilmente los ejercicios que necesitan al momento de crear nuevas rutinas para sus pacientes. Además de que se añadió un filtrado para los ejercicios

Ejercicios					
Rutinas	Ejercicios				
Biceps	Buscar				
Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción
Remo con barra	Espalda	Tren Superior	3	6 a 8	Inclina el torso hacia adelante con la barra en las manos. Sube la barra hacia el abdomen y baje controladamente. Mantén la espalda recta durante el ejercicio.
Remo con mancuerna	Espalda	Tren Superior	4	12 a 15	Con una mancuerna en una mano y apoyando la otra en un banco, lleva la mancuerna hacia el abdomen. Controla la bajada y mantén la espalda recta.
Remo en máquina con peso	Espalda	Tren Superior	3	8 a 10	Siéntate en la máquina de remo, sujetas las manijas y jala hacia la cintura, manteniendo la espalda recta. Acuéstate boca abajo en un banco inclinado con mancuernas en las

## ○ VER LISTA DE ALIMENTOS

Se presenta una lista de alimentos clasificados por tipo de comida (desayuno, almuerzo, cena, merienda) y con información detallada sobre los nutrientes principales (proteínas, grasas y carbohidratos). Esto ayuda a los nutriólogos a seleccionar los alimentos adecuados para cada paciente y garantizar que las dietas sean balanceadas y adaptadas a los objetivos de cada usuario. Se añadió una opción de filtrado.

Alimentos			
Alimento	Nutrientes	Tipo	Descripción
Papa Rellena con Carne	22g de proteína, 50g de carbohidratos, 12g de grasas	Cena	Papas rellenas con carne molida de res (100g), con cebolla, ajo, y especias. Aporta calorías para recuperación nocturna.
Tallarines Verdes con Pollo	28g de proteína, 90g de carbohidratos, 18g de grasas	Cena	Pasta (200g) con pesto de albahaca, espinacas y pollo a la plancha (150g). Alto en carbohidratos y proteínas.
Sopa de Verduras con Pollo	15g de proteína, 30g de carbohidratos, 5g de grasas	Cena	Sopa a base de verduras (zapallo, zanahoria, espinaca) y pechuga de pollo desmenuzada. Ligero y bajo en calorías.
Tortilla de Claras con Espinacas	12g de proteína, 10g de carbohidratos, 3g de grasas	Cena	Tortilla hecha con claras de huevo (4) y espinacas. Ideal para una cena ligera y alta en proteinas.
Tortilla de espinacas, queso y Yuca	22 g de proteína, 20 g de carbohidratos, 14 g de grasas	Cena	Tortilla preparada con 150 g de espinacas frescas, 100 g de queso bajo en grasa y 3 huevos (150 g). Servida con 100 g de yuca cocida.

## ○ EDICIÓN DE RUTINAS Y DIETAS

Se implementó la funcionalidad para que los nutriólogos editen tanto rutinas como dietas. Pueden actualizar descripciones, cambiar los ejercicios o alimentos incluidos y ajustar las recomendaciones de acuerdo con los progresos del paciente. Esta opción asegura que la información esté siempre actualizada y alineada con los objetivos actuales del paciente.

Editar mi rutina																											
Nombre de rutina	<input type="text" value="Fuerza Inferior 3"/>																										
Tipo de rutina	<input style="border: none; background-color: #007bff; color: white; padding: 5px; margin-right: 10px;" type="button" value="Rutina de Deficit"/> <input style="border: 1px solid #007bff; padding: 5px;" type="button" value="Rutina de Deficit"/> <input style="border: 1px solid #007bff; padding: 5px;" type="button" value="Rutina de Volumen"/>																										
Nivel	<input style="border: none; background-color: #007bff; color: white; padding: 5px; margin-right: 10px;" type="button" value="Principiante"/> <input style="border: 1px solid #007bff; padding: 5px;" type="button" value="Principiante"/>																										
Descripción	<input type="text" value="Esta rutina solo es de prueba"/>																										
Ejercicios	<table border="1"> <thead> <tr> <th>Nombre del Ejercicio</th> <th>Grupo Muscular</th> <th>Tipo</th> <th>Número de Series</th> <th>Número de Repeticiones</th> <th>Descripción</th> <th>Seleccionar</th> </tr> </thead> <tbody> <tr> <td>Press de banca</td> <td>Pecho</td> <td>Tren Superior</td> <td>3</td> <td>6 a 8</td> <td>Acuéstate en un banco plano con los pies en el suelo. Baja la barra hasta el pecho y empujala hacia arriba hasta estirar los brazos. Mantén control en todo el movimiento.</td> <td><input checked="" type="checkbox"/></td> </tr> <tr> <td>Press inclinado</td> <td>Pecho</td> <td>Tren Superior</td> <td>3</td> <td>6 a 8</td> <td>En un banco inclinado, baja la barra o mancuernas hacia el pecho superior. Empuja hacia arriba contrayendo el pecho. Controla el movimiento para evitar lesiones.</td> <td><input checked="" type="checkbox"/></td> </tr> </tbody> </table>						Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción	Seleccionar	Press de banca	Pecho	Tren Superior	3	6 a 8	Acuéstate en un banco plano con los pies en el suelo. Baja la barra hasta el pecho y empujala hacia arriba hasta estirar los brazos. Mantén control en todo el movimiento.	<input checked="" type="checkbox"/>	Press inclinado	Pecho	Tren Superior	3	6 a 8	En un banco inclinado, baja la barra o mancuernas hacia el pecho superior. Empuja hacia arriba contrayendo el pecho. Controla el movimiento para evitar lesiones.	<input checked="" type="checkbox"/>
Nombre del Ejercicio	Grupo Muscular	Tipo	Número de Series	Número de Repeticiones	Descripción	Seleccionar																					
Press de banca	Pecho	Tren Superior	3	6 a 8	Acuéstate en un banco plano con los pies en el suelo. Baja la barra hasta el pecho y empujala hacia arriba hasta estirar los brazos. Mantén control en todo el movimiento.	<input checked="" type="checkbox"/>																					
Press inclinado	Pecho	Tren Superior	3	6 a 8	En un banco inclinado, baja la barra o mancuernas hacia el pecho superior. Empuja hacia arriba contrayendo el pecho. Controla el movimiento para evitar lesiones.	<input checked="" type="checkbox"/>																					

## Editar mi dieta

Nombre de dieta

Fuerza y sabor

Objetivo

Deficit ▾

Descripción

Esta dieta está diseñada para potenciar tu masa muscular de manera saludable y efectiva. Comienza el día con un batido de yogur y frutas, acompañado de pan con palta y huevo, que proporciona una mezcla equilibrada de proteínas y carbohidratos. Una media mañana, disfruta de yogur con semillas de chía y frutas, que te ofrecerá fibra y grasas saludables para mantenerte saciado. Para el almuerzo, un delicioso lomo saltado con arroz y papas, que combina proteínas magras y carbohidratos energéticos.

Alimentos

Alimento	Nutrientes	Tipo	Descripción	Seleccionar
Pan con Palta y Huevo	carbohidratos, 18g de grasas	Desayuno	Un integral con palta (80g) y huevo frito o cocido (2 huevos). Rico en grasas saludables y proteínas.	<input type="checkbox"/>
Arroz con Pollo	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Plato principal. Contiene arroz (200g), pollo (150g), arvejas, zanahorias y aceite vegetal.	<input type="checkbox"/>
Lentejas con Carne	30g de proteína, 80g de carbohidratos, 15g de grasas	Almuerzo	Lentejas (200g) acompañadas con carne de res (150g) y arroz blanco. Rica en fibra, carbohidratos y proteínas para volumen.	<input type="checkbox"/>

## ○ PERFIL DE NUTRIOLOGOS

El nutriólogo ahora tiene acceso a su perfil donde puede ver todas las rutinas y dietas que ha creado. Además, se ha implementado un botón para crear nuevas rutinas y dietas. También se incluye una opción para editar su perfil, donde puede añadir detalles como la descripción personal, su lugar de trabajo, y sus datos de contacto. Esta funcionalidad permite al nutriólogo gestionar su presencia en la plataforma y hacerla más accesible a los pacientes.

<- Retroceder

Avatar

Tester Nutriologo

Sin biografía

Reuniones

Cita sobre rutinas  
DOMINGO  
20:00

Tips para mejorar tu rutina

VER MÁS

Rutinas creadas

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Fuerza Inferior 2	Deficit	Tren Inferior	Intermedio	Esta rutina está pensada para principiantes que buscan aumentar su masa muscular en el tren inferior. Comienza con la sentadilla con barra para activar los cuádriceps. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice un peso ligero y concentrado en la técnica, especialmente en la sentadilla y el peso muerto. Estos ejercicios son fundamentales para el desarrollo de glúteos y piernas fuertes, así que asegúrese de realizar cada movimiento con control.	<a href="#">Ver detalles</a>
Fuerza Inferior 3	Rutina de Deficit	Tren Inferior	Avanzado	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>
Rutina de prueba 1	Volumen	Tren Superior	Intermedio	Esta rutina solo es de prueba	<a href="#">Ver detalles</a>

## ○ PERFIL DE PACIENTES

Los pacientes ahora tienen su propio perfil donde pueden ver las rutinas y dietas que han seguido. Se les permite editar su perfil para incluir una descripción personal y actualizar la información relacionada con el cuestionario de salud. Esta funcionalidad brinda al paciente un mayor control sobre su información y le permite gestionar sus datos dentro de la plataforma.

The screenshot displays a patient profile interface. On the left, there's a sidebar with a back button, an avatar placeholder, and a bio section for "Gonzalo Tafur Bermúdez" stating "Hola, esta es mi biografía". It includes edit buttons for profile and biography, and details like birth date (2003-10-28), nationality (Otro), and gender (Masculino). Below this is a "Medidas de seguimiento" section with a "Ver seguimiento" button. The main content area starts with an "Información" header and a table titled "Dieta(s) que estoy haciendo". The table has columns for Nombre (Nutrición Vital), Objetivo (Volumen), Descripción (a diet for muscle gain), and Acciones (links to "Ver detalles" and "Dejar de seguir Dieta"). Below the table is a "Seguir otras dietas" link. Further down, there are sections for "Objetivo nutricional" (Mejorar Hábitos), "Frecuencia de ejercicios" (1 hora a diario), "Condición alimenticia" (Ninguna), "Peso corporal" (78.45), "Estatura" (1.72), "Perímetro de cintura" (50.00), "Perímetro de cadera", "Perímetro de muslo" (29.00), and "Perímetro de brazo/bicep". At the bottom is an "Editar mi información de paciente" button.

## ○ AJUSTES DE USUARIO

Se ha implementado la opción de ajustes de usuario, donde los usuarios pueden cambiar su contraseña si la olvidan y también tienen la opción de cerrar sesión. Esto proporciona un nivel adicional de seguridad y control sobre las cuentas de usuario.

<- Retroceder

## Ajustes de cuenta

**Cambiar contraseña**

Nueva contraseña

Confirmar contraseña

**Guardar**

**Cambiar Correo electrónico**

Correo Electronico

**Guardar**

¿Quieres cerrar sesión?

**Cerrar Sesión**

### o PERSONALIZAR CALENDARIO DE PACIENTE

Los pacientes ahora pueden gestionar un calendario personal donde pueden agregar las rutinas de ejercicios proporcionadas por el nutriólogo. El calendario permite asignar días específicos de entrenamiento, seleccionar las rutinas para esos días y programar los días de descanso. Esta funcionalidad facilita que los pacientes sigan sus programas de entrenamiento y se organicen de manera eficiente.

Horario de rutinas

Programa tus tiempos para hacer rutinas a tu gusto.

Fuerza Inferior 2

[Ver detalles](#)

Descanso por serie:  
Descanso por ejercicio:

LUNES

[Eliminar de tu horario](#)

Periodo: Tarde

Definición Inferior 3

[Ver detalles](#)

Descanso por serie:  
Descanso por ejercicio:

VIERNES

[Eliminar de tu horario](#)

Periodo: Mañana

Inserte un nuevo horario

Rutina:

Fuerza Inferior 1

Día:

Miércoles

Periodo de día:

Medio día

Descanso por ejercicio:

No especificado

Descanso por serie:

No especificado

Guardar nuevo horario

## Rutinas

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción
Fuerza Superior 1	Volumen	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en fase de volumen. Comienza con una serie de ejercicios de flexión que trabajan los principales músculos del tren superior, favoreciendo el desarrollo de la fuerza y la resistencia. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice pesos que permitan realizar 8 a 12 repeticiones de cada ejercicio.

### o NOTIFICACIONES PARA EL USUARIO

Se completó la funcionalidad de notificaciones personalizadas. Ahora, tanto pacientes como nutriólogos reciben alertas sobre actividades programadas, reuniones y consejos diarios de salud según los objetivos personales establecidos.

Las notificaciones se muestran en la página de notificaciones del usuario y están relacionadas con los eventos y tareas programadas, como recordatorios de reuniones y actualizaciones de rutinas.

El sistema ahora garantiza que las notificaciones sean relevantes y se alineen con las preferencias y objetivos del usuario.

### PACIENTE

<- Retroceder

### Notificaciones

2024-12-18T00:00:06.017

Han pasado 7 días desde tu última edición, aprovecha en editarlas si lo necesitas.  
2024-12-18T00:00:06.017

### NUTRIOLOGO

## Notificaciones

**2024-12-16T23:57:51.627**

El paciente Tests confirmó unirse a la reunión Tips de dietas

**2024-12-16T15:47:43.517**

El paciente Gonzalo confirmó unirse a la reunión Tips de dietas

**2024-12-11T14:02:12.897**

El paciente Gonzalo confirmó unirse a la reunión Cita sobre rutinas

**2024-12-03T19:29:40.670**

El paciente Tester confirmó unirse a la reunión Cita sobre rutinas

## o REGISTRO SEMANAL DE MEDIDAS Y SEGUIMIENTO DE PROGRESO

La funcionalidad para que los pacientes ingresen sus medidas semanales fue finalizada. Los pacientes ahora pueden registrar datos como su peso, perímetro de cintura y bíceps.

El sistema guarda los datos de progreso en el perfil del paciente y muestra cómo ha cambiado con el tiempo.

Se implementó una restricción de 7 días entre registros de medidas para evitar entradas prematuras. En caso de que el paciente intente registrar medidas antes de transcurrir una semana, recibirá una alerta informando que debe esperar.

Historial de seguimiento de medidas

Fecha de modificación	Frecuencia de ejercicios	Peso corporal	Estatura	Perímetro de cintura	Perímetro de cadera	Perímetro de muslo	Perímetro de brazo
2024-12-17T22:29:08.897	1 hora a diario	78.45	1.72	50.00		29.00	
2024-12-03T17:26:59.997	30 minutos a diario	80.45	1.71				
2024-12-03T11:59:28.410	30 minutos a diario	79.00	1.66				
2024-11-19T17:46:15.337	30 minutos a diario	67.45	1.70				
2024-11-19T17:09:00.413	1 hora a diario	67.00	1.70				
2024-11-19T16:39:00.843	1 hora a diario	65.00	1.70				
2024-11-19T16:39:00.843	1 hora a diario	70.77	1.67				

Retroceder



## Información

Debe esperar al menos 7 días antes de poder realizar otra modificación.

## ○ RECOMENDACIÓN INICIAL DE DIETAS Y RUTINAS PERSONALIZADAS:

El sistema ahora recomienda dietas y rutinas personalizadas para los pacientes recién registrados, basándose en las respuestas del cuestionario completado tras el registro.

Estas recomendaciones consideran los objetivos del paciente (déficit calórico o volumen) y su experiencia física previa, asegurando que cada paciente reciba un plan adaptado a sus necesidades específicas.

La funcionalidad fue probada y se validó que las recomendaciones se ajustan a los datos ingresados por el paciente en el cuestionario inicial.

**Nuestras recomendaciones iniciales son**

Bienvenido, paciente nuevo. Antes de ingresar al menú, queremos que empieces eligiendo las dietas y rutinas que realizaras.

<b>Objetivo nutricional:</b>	Volumen
<b>Frecuencia de ejercicios:</b>	30 minutos a diario
<b>Condición alimenticia:</b>	Ninguna
<b>Peso corporal:</b>	72
<b>Estatura:</b>	1.74
<b>Perímetro de cintura:</b>	
<b>Perímetro de cadera:</b>	
<b>Perímetro de muslo:</b>	
<b>Perímetro de brazo/bicep:</b>	

**Rutinas**

Nombre de la Rutina	Tipo	Parte del cuerpo	Nivel	Descripción	Acciones
Fuerza Superior 1	Volumen	Tren Superior	Principiante	Esta rutina está diseñada para principiantes en fase de volumen. Comienza con la prensa de banca para activar los músculos del pecho. Se recomienda un descanso de 3 a 4 minutos entre series. Utilice pesos ligeros para asegurar una técnica adecuada. La rutina trabaja los	<a href="#">Ver detalles</a>

**Dietas**

Nombre	Objetivo	Descripción	Acciones
Energía sostenible	Volumen	Esta dieta está diseñada para ayudarte a aumentar tu masa muscular de forma saludable y equilibrada. Comienza tu día con un nutritivo bowl de quinua con leche y frutos secos, acompañado de tostadas integrales con mantequilla de maní y plátano, que te ofrecerá una combinación ideal de proteínas, carbohidratos y grasas saludables. Una media	<a href="#">Ver detalles</a>

- **PROGRAMAR REUNIONES CON EL NUTRÍÓLOGO**

Los nutriólogos ahora pueden programar reuniones virtuales con sus pacientes, tanto grupales como individuales. Se implementaron horarios predefinidos para las reuniones, y los pacientes podrán confirmar su asistencia.

Una vez confirmada la asistencia, el nutriólogo puede visualizar una lista de los asistentes y revisar un resumen de su perfil, que incluye información sobre sus dietas, seguimiento de medidas y progreso registrado.

Esta funcionalidad facilita la gestión de reuniones y seguimiento personalizado para los nutriólogos.

## Reuniones

Cita sobre rutinas  
DOMINGO  
20:00  
[Enlace de la reunión](#) [Desactivar Reunion](#)

---

Tips para mejorar tu rutina  
VIERNES  
18:00  
[Enlace de la reunión](#) [Desactivar Reunion](#)

---

Tips de dietas  
MARTES  
11:00  
[Enlace de la reunión](#) [Desactivar Reunion](#)

---

Pacientes asistidos [Programar una nueva reunión](#)

## Programar una nueva reunión

Motivo

Día

 ▼

Hora

 ⌚

¿En que plataforma se va ha hacer la reunión?

[Zoom](#) selected

[Meet](#)

[---](#)

[Guardar Reunión](#)

## Reuniones disponibles

### Cita sobre rutinas

DOMINGO  
20:00

[Entrar a la reunión](#)

[Cancelar Asistencia](#)

### Tips para mejorar tu rutina

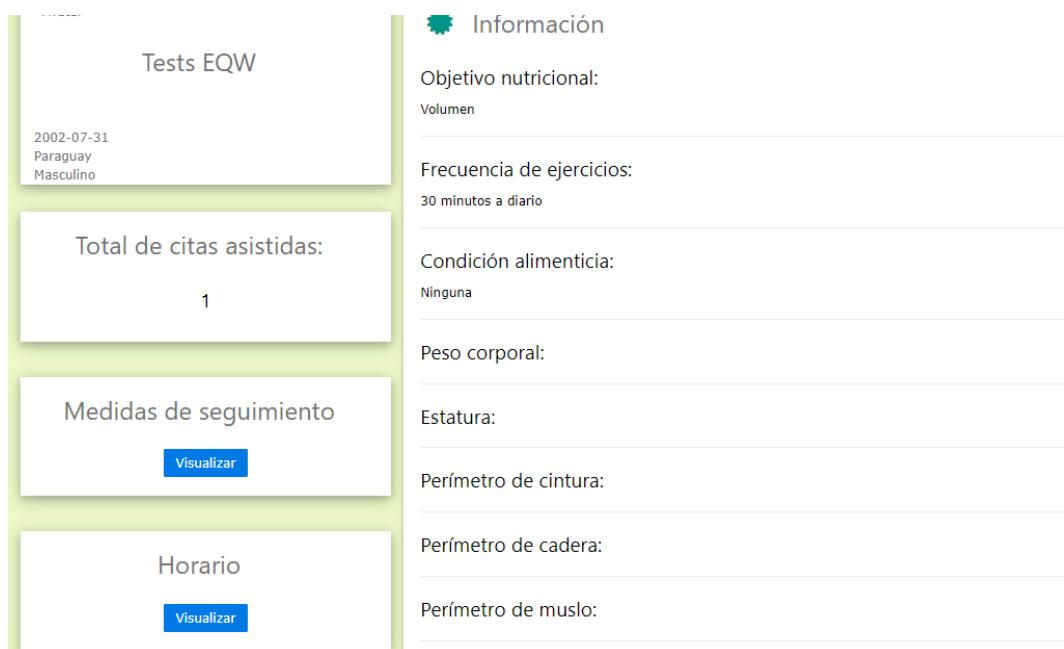
VIERNES  
18:00

[Confirmar Asistencia](#)

## Pacientes asistidos a la reunión

Reunión	Hora y día	Paciente	Se unió en:	Estado	Historial de medidas	Perfil
Tips de dietas	MARTES 11:00	Tests EQW	2024-12-16T23:57:51.600	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>
Tips de dietas	MARTES 11:00	Gonzalo Tafur Bermúdez	2024-12-16T15:47:43.357	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>
Cita sobre rutinas	DOMINGO 20:00	Gonzalo Tafur Bermúdez	2024-12-11T14:02:12.857	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>
Cita sobre rutinas	DOMINGO 20:00	Tester Paciente	2024-12-03T19:29:40.613	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>
Reunión de presentación	MARTES 18:30	Gonzalo Tafur Bermúdez	2024-12-03T17:25:55.673	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>
Reunión de presentación	MARTES 18:30	Tester Paciente	2024-12-03T11:01:51.560	Activo	<a href="#">Ver</a>	<a href="#">Ver</a>

2024. Todos los derechos reservados.



The screenshot shows a mobile application interface. On the left, there's a sidebar with three main sections: "Tests EQW" (with a date 2002-07-31 and location Paraguay/Masculino), "Total de citas asistidas: 1" (with a "Visualizar" button), and "Horario" (with a "Visualizar" button). The main content area on the right is titled "Información" and contains the following fields:

- Objetivo nutricional: Volumen
- Frecuencia de ejercicios: 30 minutos a diario
- Condición alimenticia: Ninguna
- Peso corporal:
- Estatura:
- Perímetro de cintura:
- Perímetro de cadera:
- Perímetro de muslo:

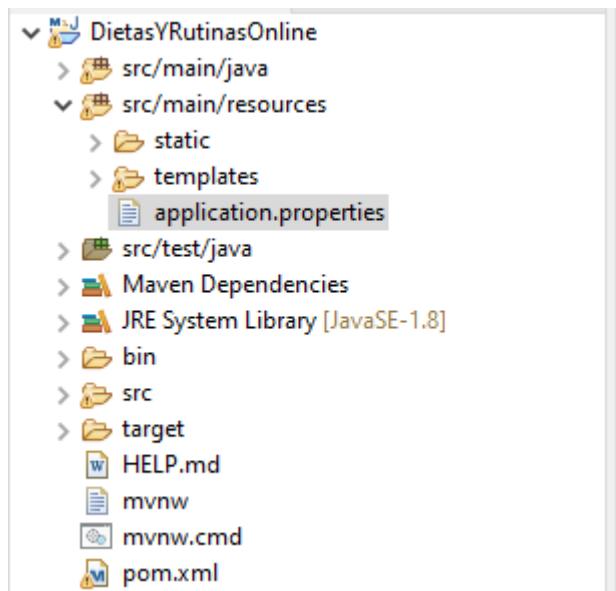
- **MEJORAS DE LA UI Y EXPERIENCIA DEL USUARIO**

Se implementaron mejoras significativas en la interfaz de usuario para hacer la aplicación más intuitiva y visualmente atractiva. Esto incluyó la restricción de reuniones duplicadas, un mejor diseño en la creación de dietas y rutinas con una tabla desplegable de alimentos y ejercicios, y una optimización general del diseño para mejorar la navegación y la accesibilidad.

## ACCESO A LA PLATAFORMA DE DESARROLLO

Acceso: <https://github.com/GonzaloTafur/ProyectoTecnologicoGrupo3>

9. Crear una carpeta y extraer ahí el proyecto descargado en el GitHub. A dentro también hay un Word de manual de instrucciones como respaldo.
10. Exportar la carpeta “bd” que está dentro del repositorio, que incluye el query para la creación de la base de datos y las tablas.
11. Cambiar la ruta del application.properties con el nombre del servidor del equipo que haga la prueba.



## 12. Ejecutar el proyecto

En Eclipse: Click derecho al proyecto importado -> Run As -> 1 Run on Server -> Finish

## 9.5 Sprint Retrospective:

### ANÁLISIS CRÍTICO DEL TRABAJO REALIZADO

#### a) Alcance final (MVP)

- ✓ **Gestión de usuarios:** Registro, acceso, y perfiles personalizados para pacientes y nutriólogos.
- ✓ **Rutinas y dietas personalizadas:** Creación, edición y visualización de rutinas y dietas adaptadas a los objetivos de los usuarios.
- ✓ **Seguimiento del progreso:** Registro de medidas físicas, con seguimiento semanal del progreso.
- ✓ **Recomendaciones personalizadas:** El sistema recomienda rutinas y dietas basadas en las respuestas del paciente a un cuestionario inicial.
- ✓ **Programación de reuniones:** Los nutriólogos pueden gestionar reuniones con los pacientes para seguimiento.

#### b) Lo que funcionó bien:

- ✓ La plataforma ha demostrado ser fluida y fácil de usar, con una distribución adecuada de tareas tanto para pacientes como para nutriólogos.
- ✓ La interfaz es intuitiva, lo que permite a los usuarios navegar sin complicaciones.

- ✓ Conexión y funcionalidades principales (registro, inicio de sesión, visualización de rutinas y dietas) se completaron correctamente, alcanzando el objetivo de facilitar la gestión de dietas y rutinas.

## c) Áreas de mejora:

- ✓ El proceso de personalización para pacientes con condiciones especiales (alergias, diabetes, etc.) no está completamente implementado. Esto podría ser una mejora importante para ampliar el alcance del proyecto y adaptarlo a más tipos de usuarios.
- ✓ Índice de Masa Corporal (IMC) no fue considerado en el sistema como indicador adicional al registrar las medidas. Este es un dato relevante para evaluar el progreso de los pacientes en general, aunque no sea tan adecuado para deportistas o usuarios con objetivos estéticos.
- ✓ Vinculación con Google: Aunque los usuarios pueden ingresar con su correo, la integración completa con Gmail aún no se ha realizado. Esta dependencia podría mejorar la experiencia de usuario al vincular sus cuentas con la plataforma de manera más directa.

## FEEDBACK DEL NUTRÍÓLOGO (14/12/2024)

### a) Positivos:

- ✓ La plataforma cumple con las expectativas funcionales y de usabilidad.
- ✓ Buen flujo de trabajo entre pacientes y nutriólogos.

### b) Áreas a mejorar:

- ✓ Incluir dietas específicas para pacientes con condiciones médicas (alergias, diabetes, etc.).
- ✓ Añadir un sistema de valoración de rutinas y dietas por parte de los pacientes, lo cual podría ayudar a mejorar el servicio y generar más interacción.
- ✓ Recomendaciones automáticas: Implementar una funcionalidad que sugiera cambios en dietas o rutinas si los datos del paciente (como peso) no se alinean con sus objetivos.
- ✓ Notificaciones mejoradas: Añadir enlaces directos a las reuniones y otras funcionalidades desde las notificaciones para facilitar el acceso.

## FUTURAS MEJORAS PROPUESTAS

### a) Mejoras funcionales:

- ✓ Recomendaciones automáticas: Implementar un sistema que sugiera ajustes a la dieta o rutinas si el progreso de un paciente no sigue sus objetivos, especialmente cuando hay alteraciones en las medidas del usuario.
- ✓ Valoración de rutinas y dietas: Crear un sistema de feedback donde los pacientes puedan calificar las rutinas y dietas proporcionadas por los nutriólogos, permitiendo mejorar la calidad del servicio.

- ✓ Exportar listas a Excel: Debido a que los nutriólogos también hacen sus anotaciones de dietas y rutinas en Excel, se creará la opción de exportar listas para que los usuarios puedan llevarlo y editarlo en todo momento.

## b) Mejoras técnicas:

- ✓ Integración con Google: Vincular la plataforma con el correo electrónico de Google para facilitar la autenticación y mejorar la interacción del usuario con la plataforma, como en la parte de notificaciones.
- ✓ Integración con Zoom: Vincular la aplicación con la API de Zoom para que un nutrólogo pueda integrar su cuenta desde ahí y que los pacientes puedan tener un acceso más rápido a la sala.
- ✓ Gráfico de progreso: Implementar un gráfico de barras interactivo para mostrar el historial de medidas de los pacientes, lo que facilitaría la interpretación de su progreso de una manera visual.

## c) Mejoras administrativas:

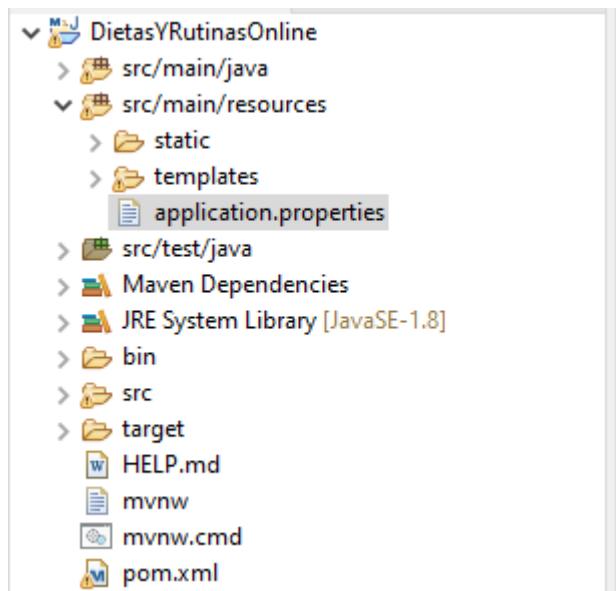
- ✓ Interfaz de Administrador: Desarrollar una sección de administración que permita gestionar registros de nutrólogos, incluyendo la verificación de certificados, experiencia y solicitud de aprobación. Esto podría mejorar el control de calidad y la seguridad del sistema.
- ✓ Visualización de transacciones: Implementar una sección de administración para visualizar transacciones y gestionar el sistema de manera más eficiente.

## 10. DESPLIEGUE DEL PRODUCTO

### ACCESO AL REPOSITORIO

Acceso: <https://github.com/GonzaloTafur/ProyectoTecnologicoGrupo3.git>

1. Crear una carpeta y extraer ahí el proyecto descargado en el GitHub. A dentro también hay un Word de manual de instrucciones como respaldo.
2. Exportar la carpeta “bd” que está dentro del repositorio, que incluye el query para la creación de la base de datos y las tablas.
3. Cambiar la ruta del application.properties con el nombre del servidor del equipo que haga la prueba.

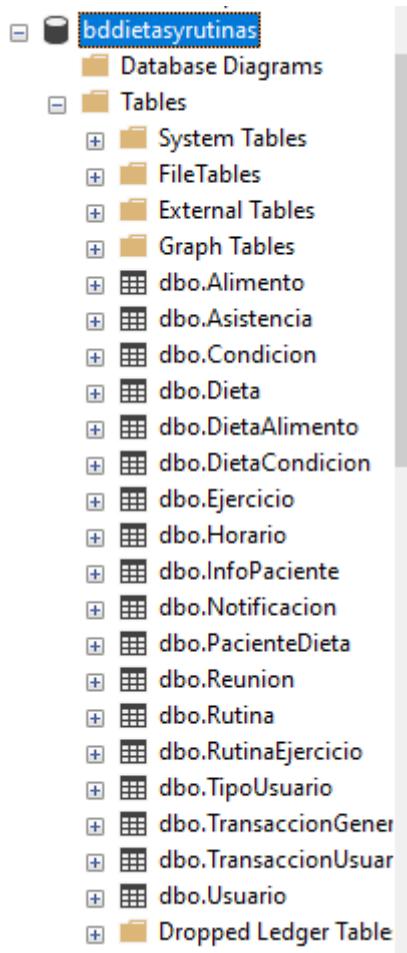


#### 4. Ejecutar el proyecto

En Eclipse: Click derecho al proyecto importado -> Run As -> 1 Run on Server -> Finish

## DOCUMENTACION TECNICA

### Base de datos



## Creación de la BD y tablas

```
USE master
GO

DROP DATABASE IF exists bddietasyrutinas
CREATE DATABASE bddietasyrutinas
GO

USE bddietasyrutinas
GO

CREATE TABLE TipoUsuario(
    idtipousu INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomtipousu VARCHAR(20),
    esttipousu VARCHAR(20),
)
```

```
CREATE TABLE Usuario(
    idusuario INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idtipousu INT,
    nomusuario VARCHAR(20),
    apeusuario VARCHAR(50),
    fechanacimiento DATE,
    nacionalidad VARCHAR(20),
    sexo VARCHAR(10),
    biografia VARCHAR(200),
    estusuario VARCHAR(20),
    correo VARCHAR(50),
    password VARCHAR(255),
    FOREIGN KEY (idtipousu) REFERENCES TipoUsuario(idtipousu)
)

CREATE TABLE Ejercicio (
    idejercicio INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomejercicio VARCHAR(60),
    grupomuscular VARCHAR(30),
    tipoejercicio VARCHAR(30),
    series INT,
    repeticiones VARCHAR(10),
    descejercicio VARCHAR(300),
)

CREATE TABLE Rutina (
    idrutina INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomrutina VARCHAR(30),
    tiporutina VARCHAR(30),
    parteCuerpo VARCHAR(30),
    nivel VARCHAR(30),
    descriutura VARCHAR(700),
    estrutura VARCHAR(20),
    idusuario INTEGER,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario)
)

CREATE TABLE RutinaEjercicio (
    id_rutina INTEGER NOT NULL,
    id_ejercicio INTEGER NOT NULL,
    PRIMARY KEY (id_rutina, id_ejercicio),
    FOREIGN KEY (id_rutina) REFERENCES Rutina(idrutina),
    FOREIGN KEY (id_ejercicio) REFERENCES Ejercicio(idejercicio)
)
```

```
CREATE TABLE Reunion (
    idreunion INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idusuario INT,
    motivo VARCHAR(60),
    /*fecha DATE,*/
    diaSemana VARCHAR(20),
    horaInicio TIME,
    enlace VARCHAR(300),
    estreunion VARCHAR(20),
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
)
CREATE TABLE Asistencia (
    idasistencia INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idreunion INT,
    idusuario INT,
    fecasistencia DATETIME,
    estados VARCHAR(20),
    estasistencia VARCHAR(20),
    FOREIGN KEY (idreunion) REFERENCES Reunion(idreunion),
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
)
CREATE TABLE Horario (
    idhorario INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idusuario INT,
    idrutina INT,
    diaSemana VARCHAR(10),
    idperiodo INT,
    periodo VARCHAR(10),
    esthorario VARCHAR(20),
    tiempoDescanso INT,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
    FOREIGN KEY (idrutina) REFERENCES Rutina(idrutina),
)
CREATE TABLE Alimento (
   idalimento INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomalimento VARCHAR(80),
    nutrientes VARCHAR(100),
    tipoalimento VARCHAR(30),
    descalimento VARCHAR(300)
)
CREATE TABLE Condicion (
    idcondicion INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomcondicion VARCHAR(30),
    estcondicion VARCHAR(20)
)
```

```
CREATE TABLE Dieta (
    iddieta INTEGER IDENTITY (1, 1) PRIMARY KEY,
    nomdieta VARCHAR(60),
    objdieta VARCHAR(60),
    descdieta VARCHAR(900),
    estdieta VARCHAR(20),
    idusuario INT,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario)
)

CREATE TABLE DietaAlimento (
    id_dieta INTEGER NOT NULL,
    id_alimento INTEGER NOT NULL,
    PRIMARY KEY (id_dieta, id_alimento),
    FOREIGN KEY (id_dieta) REFERENCES Dieta(iddieta),
    FOREIGN KEY (id_alimento) REFERENCES Alimento(idalimento)
)

CREATE TABLE DietaCondicion (
    id_dieta INTEGER NOT NULL,
    id_condicion INTEGER NOT NULL,
    PRIMARY KEY (id_dieta, id_condicion),
    FOREIGN KEY (id_dieta) REFERENCES Dieta(iddieta),
    FOREIGN KEY (id_condicion) REFERENCES Condicion(idcondicion)
)

CREATE TABLE InfoPaciente (
    idinfopaciente INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idusuario INT,
    frecEjercicios VARCHAR(30),
    condicion VARCHAR(30),
    pesoCorporal DECIMAL(5, 2),
    estatura DECIMAL(5, 2),
    perimCintura DECIMAL(5, 2),
    perimCadera DECIMAL(5, 2),
    perimMuslo DECIMAL(5, 2),
    perimBrazo DECIMAL(5, 2),
    objetivo VARCHAR(20),
    estinfo VARCHAR(20),
    fechainfo DATETIME,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
)

CREATE TABLE PacienteDieta (
    id_paciente INTEGER NOT NULL,
    id_dieta INTEGER NOT NULL,
    PRIMARY KEY (id_paciente, id_dieta),
    FOREIGN KEY (id_paciente) REFERENCES InfoPaciente(idinfopaciente),
    FOREIGN KEY (id_dieta) REFERENCES Dieta(iddieta)
)

CREATE TABLE TransaccionGeneral (
    idtrangeneral INTEGER IDENTITY (1, 1) PRIMARY KEY,
```

```
CREATE TABLE TransaccionGeneral (
    idtrangeneral INTEGER IDENTITY (1, 1) PRIMARY KEY,
    fechatrans DATETIME,
    desctrans VARCHAR(100),
    idusuario INT,
    idrutina INT,
    iddieta INT,
    idhorario INT,
    idinfopaciente INT,
    idreunion INT,
    idasistencia INT,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
    FOREIGN KEY (idrutina) REFERENCES Rutina(idrutina),
    FOREIGN KEY (iddieta) REFERENCES Dieta(iddieta),
    FOREIGN KEY (idhorario) REFERENCES Horario(idhorario),
    FOREIGN KEY (idinfopaciente) REFERENCES InfoPaciente(idinfopaciente),
    FOREIGN KEY (idreunion) REFERENCES Reunion(idreunion),
    FOREIGN KEY (idasistencia) REFERENCES Asistencia(idasistencia)
)

CREATE TABLE TransaccionUsuario (
    idtransusu INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idusuario INT,
    idtipousu INT,
    fecharegistro DATETIME,
    fechalogin DATETIME,
    fechalogout DATETIME,
    cambioPassword DATETIME,
    cambioCorreo DATETIME,
    fperfilactual DATETIME,
    FOREIGN KEY (idusuario) REFERENCES Usuario(idusuario),
    FOREIGN KEY (idtipousu) REFERENCES TipoUsuario(idtipousu),
)

CREATE TABLE Notificacion (
    idnoti INTEGER IDENTITY (1, 1) PRIMARY KEY,
    idtrangeneral INT,
    rolnoti VARCHAR(20),
    mensaje VARCHAR(300),
    estnoti VARCHAR(20),
    diaSemana VARCHAR(20),
    timestamp DATETIME,
    FOREIGN KEY (idtrangeneral) REFERENCES TransaccionGeneral(idtrangeneral),
)
```

## Conexión a la base de datos

```
1 spring.application.name=DietasYRutinasOnline
2
3 spring.datasource.driverClassName=com.microsoft.sqlserver.jdbc.SQLServerDriver
4 spring.datasource.url=jdbc:sqlserver://localhost:1433;databaseName=bdddietasyrutinas
5 spring.datasource.username=sa
6 spring.datasource.password=sa
7 spring.jpa.show-sql=true
8 spring.jpa.properties.hibernate.format_sql = true
9
10## Hibernate Properties
11# The SQL dialect makes Hibernate generate better SQL for the chosen database
12spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.SQLServer2012Dialect
13
14# Hibernate ddl auto (create, create-drop, validate, update)
15spring.jpa.hibernate.ddl-auto = none
16
17spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
18
19|
```

## Entidades:

- **Usuario**

```
16 import javax.persistence.ManyToOne;
17 import javax.persistence.ManyToOne;
18 import javax.persistence.OneToMany;
19 import javax.persistence.Table;
20
21 @Entity
22 @Table(name="Usuario")
23 public class Usuario {
24
25@  @Id
26 @GeneratedValue(strategy = GenerationType.IDENTITY)
27 @Column(name = "idusuario")
28 private int idusuario;
29
30@  @ManyToOne
31 @JoinColumn(name = "idtipousu")
32 private TipoUsuario tipousuario;
33
34@  @Column(name = "nomusuario")
35 private String nombres;
36
37@  @Column(name = "apeusuario")
38 private String apellidos;
39
40@  @Column(name = "fechanacimiento")
41 private Date fechanacimiento;
42
43@  @Column(name = "nacionalidad")
44 private String nacionalidad;
45
46@  @Column(name = "sexo")
47 private String sexo;
48
49@  @Column(name = "biografia")
50 private String biografia;
51
52@  @Column(name = "estusuario")
53 private String estado;
```

```
55@    @Column(name = "correo")
56    private String correo;
57
58@    @Column(name = "password")
59    private String password;
60
61
62@    public int getIdusuario() {
63        return idusuario;
64    }
65
66@    public void setIdusuario(int idusuario) {
67        this.idusuario = idusuario;
68    }
69
70@    public TipoUsuario getTipousuario() {
71        return tipousuario;
72    }
73
74@    public void setTipousuario(TipoUsuario tipousuario) {
75        this.tipousuario = tipousuario;
76    }
77
78@    public String getNombres() {
79        return nombres;
80    }
81
82@    public void setNombres(String nombres) {
83        this.nombres = nombres;
84    }
85
86@    public String getApellidos() {
87        return apellidos;
88    }
89
90@    public void setApellidos(String apellidos) {
91        this.apellidos = apellidos;
92    }
93
94@    public Date getFechanacimiento() {
95        return fechanacimiento;
96    }
97
98@    public void setFechanacimiento(Date fechanacimiento) {
99        this.fechanacimiento = fechanacimiento;
100    }
101
102@    public String getNacionalidad() {
103        return nacionalidad;
104    }
105
106@    public void setNacionalidad(String nacionalidad) {
107        this.nacionalidad = nacionalidad;
108    }
109
110@    public String getSexo() {
111        return sexo;
112    }
113
114@    public void setSexo(String sexo) {
115        this.sexo = sexo;
116    }
117
118@    public String getBiografia() {
119        return biografia;
120    }
121
122@    public void setBiografia(String biografia) {
123        this.biografia = biografia;
124    }
```

```
125  
126@     public String getEstado() {  
127         return estado;  
128     }  
129  
130@     public void setEstado(String estado) {  
131         this.estado = estado;  
132     }  
133  
134@     public String getCorreo() {  
135         return correo;  
136     }  
137  
138@     public void setCorreo(String correo) {  
139         this.correo = correo;  
140     }  
141  
142@     public String getPassword() {  
143         return password;  
144     }  
145  
146@     public void setPassword(String password) {  
147         this.password = password;  
148     }  
149  
150  
151 }
```

- **Tipo Usuario (Rol)**

```
1 package com.vietasyskutinasuniline.entity;
2
3+ import javax.persistence.Column;
4
5 @Entity
6 @Table(name="TipoUsuario")
7 public class TipoUsuario {
8
9     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    @Column(name = "idtipousu")
12    private int idtipousu;
13
14    @Column(name = "nomtipousu")
15    private String nomtipousu;
16
17    @Column(name = "esttipousu")
18    private String esttipousu;
19
20    public int getIdtipousu() {
21        return idtipousu;
22    }
23
24    public void setIdtipousu(int idtipousu) {
25        this.idtipousu = idtipousu;
26    }
27
28    public String getNomtipousu() {
29        return nomtipousu;
30    }
31
32    public void setNomtipousu(String nomtipousu) {
33        this.nomtipousu = nomtipousu;
34    }
35
36    public String getEsttipousu() {
37        return esttipousu;
38    }
39
40    public void setEsttipousu(String esttipousu) {
41        this.esttipousu = esttipousu;
42    }
43
44
45    }
```

- **Ejercicio**

```
1 package com.DietasYRutinasOnline.entity;
2
3+ import java.util.List;
4
5 @Entity
6 @Table(name="Ejercicio")
7 public class Ejercicio {
8
9     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    @Column(name = "idejercicio")
12    private int idejercicio;
13
14    @Column(name = "nombreejercicio")
15    private String nombre;
16
17    @Column(name = "grupomuscular")
18    private String grupomuscular;
19
20    @Column(name = "tipoejercicio")
21    private String tipo;
22
23    @Column(name = "series")
24    private int series;
25
26    @Column(name = "repeticiones")
27    private String repeticiones;
28
29    @Column(name = "descejercicio")
30    private String descripcion;
31
32    @ManyToMany(mappedBy = "ejercicio")
33    private List<Rutina> rutina;
34
35    public int getIdejercicio() {
36        return idejercicio;
37    }
38
39    public void setIdejercicio(int idejercicio) {
40        this.idejercicio = idejercicio;
41    }
42
43    public String getNombre() {
44        return nombre;
45    }
```

- **Rutina**

```
1 package com.DietasYRutinasOnline.entity;
2
3④ import java.util.List;⑤
4
5
6 @Entity
7 @Table(name="Rutina")
8 public class Rutina {
9
10    @Id
11    @GeneratedValue(strategy = GenerationType.IDENTITY)
12    @Column(name = "idrutina")
13    private int idrutina;
14
15    @Column(name = "nomrutina")
16    private String nombre;
17
18    @Column(name = "tiporutina")
19    private String tipo;
20
21    @Column(name = "partecuerpo")
22    private String parteCuerpo;
23
24    @Column(name = "nivel")
25    private String nivel;
26
27    @Column(name = "descripción")
28    private String descripción;
29
30    @Column(name = "estrutura")
31    private String estructura;
32
33    @ManyToMany
34    @JoinTable(
35        name = "Rutinaejercicio",
36        joinColumns = @JoinColumn(name = "id_rutina", referencedColumnName = "idrutina"),
37        inverseJoinColumns = @JoinColumn(name = "id_ejercicio", referencedColumnName = "idejercicio"))
38    private List<Ejercicio> ejercicio;
39
40    @ManyToOne
41    @JoinColumn(name = "idusuario")
42    private Usuario nutriologo;
43
44    public int getIdrutina() {
45        return idrutina;
46    }
47
48    public void setIdrutina(int idrutina) {
49        this.idrutina = idrutina;
50    }
51
52    public String getNombre() {
53        return nombre;
54    }
55
56    public void setNombre(String nombre) {
57        this.nombre = nombre;
58    }
59
60    public String getTipo() {
61        return tipo;
62    }
63
64    public void setTipo(String tipo) {
65        this.tipo = tipo;
66    }
67
68    public String getParteCuerpo() {
69        return parteCuerpo;
70    }
71
72    public void setParteCuerpo(String parteCuerpo) {
73        this.parteCuerpo = parteCuerpo;
74    }
75
76    public String getNivel() {
77        return nivel;
78    }
79
80    public void setNivel(String nivel) {
81        this.nivel = nivel;
82    }
83
84    public String getDescripción() {
85        return descripción;
86    }
87
88    public void setDescripción(String descripción) {
89        this.descripción = descripción;
90    }
91
92    public String getEstructura() {
93        return estructura;
94    }
95
96    public void setEstructura(String estructura) {
97        this.estructura = estructura;
98    }
99
100   public List<Ejercicio> getEjercicio() {
101       return ejercicio;
102   }
103
104   public void setEjercicio(List<Ejercicio> ejercicio) {
105       this.ejercicio = ejercicio;
106   }
107
108   public Usuario getNutriologo() {
109       return nutriologo;
110   }
111
112   public void setNutriologo(Usuario nutriologo) {
113       this.nutriologo = nutriologo;
114   }
115 }
```

- **Alimento**

```
1 package com.dietasykutinasunline.entity;
2
3④ import java.util.List;[]
12
13 @Entity
14 @Table(name="Alimento")
15 public class Alimento {
16
17④     @Id
18     @GeneratedValue(strategy = GenerationType.IDENTITY)
19     @Column(name = "idalimento")
20     private int idalimento;
21
22④     @Column(name="nomalimento")
23     private String nombre;
24
25④     @Column(name="nutrientes")
26     private String nutrientes;
27
28④     @Column(name="tipoalimento")
29     private String tipo;
30
31④     @Column(name="descalimento")
32     private String descripcion;
33
34④     @ManyToMany(mappedBy = "alimento")
35     private List<Dieta> dieta;
36
37④     public int getIdalimento() {
38         return idalimento;
39     }
40
41④     public void setIdalimento(int idalimento) {
42         this.idalimento = idalimento;
43     }
44
45④     public String getNombre() {
46         return nombre;
47     }
48
49④     public void setNombre(String nombre) {
50         this.nombre = nombre;
51     }
52
53④     public String getNutrientes() {
```

- **Dieta**

```
3④ import java.util.List;⑤
16
17 @Entity
18 @Table(name="Dieta")
19 public class Dieta {
20
21④   @Id
22     @GeneratedValue(strategy = GenerationType.IDENTITY)
23     @Column(name = "iddieta")
24     private int iddieta;
25
26④   @Column(name="nomdieta")
27     private String nombre;
28
29④   @Column(name="objdieta")
30     private String objetivo;
31
32④   @Column(name="descdieta")
33     private String descripcion;
34
35④   @Column(name="estdieta")
36     private String estado;
37
38④   @ManyToOne
39     @JoinColumn(name = "idusuario")
40     private Usuario nutriologo;
41
42④   @ManyToMany(cascade = CascadeType.ALL)
43     @JoinTable(
44       name = "DietaAlimento",
45       joinColumns = @JoinColumn(name = "id_dieta", referencedColumnName ="iddieta"),
46       inverseJoinColumns = @JoinColumn(name = "id_alimento", referencedColumnName ="idalimento"))
47     private List<Alimento> alimento;
48
49④   @ManyToMany(cascade = CascadeType.ALL)
50     @JoinTable(
51       name = "DietaCondicion",
52       joinColumns = @JoinColumn(name = "id_dieta", referencedColumnName ="iddieta"),
53       inverseJoinColumns = @JoinColumn(name = "id_condicion", referencedColumnName ="idcondicion"))
54     private List<Condicion> condicion;
55
56④   @ManyToMany(mappedBy = "dieta")
57     private List<InfoPaciente> paciente;
```

- **Condición**

```
1 package com.DietaSYRutinasOnline.entity;
2
3④ import java.util.List;⑤
4
5 @Entity
6 @Table(name="Condicion")
7 public class Condicion {
8
9     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    @Column(name = "idcondicion")
12    private int idcondicion;
13
14    @Column(name="nomcondicion")
15    private String nombre;
16
17    @Column(name="estcondicion")
18    private String estado;
19
20    @ManyToMany(mappedBy = "condicion")
21    private List<Dieta> dieta;
22
23    public int getIdcondicion() {
24        return idcondicion;
25    }
26
27    public void setIdcondicion(int idcondicion) {
28        this.idcondicion = idcondicion;
29    }
30
31    public String getNombre() {
32        return nombre;
33    }
34
35    public void setNombre(String nombre) {
36        this.nombre = nombre;
37    }
38
39    public String getEstado() {
40        return estado;
41    }
42
43    public void setEstado(String estado) {
44        this.estado = estado;
45    }
46
47    public String toString() {
48        return "Condicion{" +
49            "idcondicion=" + idcondicion +
50            ", nombre='" + nombre + '\'' +
51            ", estado='" + estado + '\'' +
52            '}';
53}
```

- **InfoPaciente**

```
1 package com.DiétasYRutinasOnline.entity;
2
3④ import java.math.BigDecimal;⑤
4
5 ⑥ @Entity
6 ⑦ @Table(name="InfoPaciente")
7 ⑧ public class InfoPaciente {
8
9    ⑨     @Id
10    ⑪     @GeneratedValue(strategy = GenerationType.IDENTITY)
11    ⑫     @Column(name = "idinfopaciente")
12     ⑬     private int idinfopaciente;
13
14    ⑭     @ManyToOne(cascade = CascadeType.ALL)
15     ⑮     @JoinColumn(name = "idusuario")
16     ⑯     private Usuario paciente;
17
18    ⑰     @Column(name="frecEjercicios")
19     ⑱     private String frecEjercicios;
20
21    ⑲     @Column(name="condicion")
22     ⑳     private String condicion;
23
24    ⑳     @Column(name="pesoCorporal")
25     ㉑     private BigDecimal pesoCorporal;
26
27    ㉒     @Column(name="estatura")
28     ㉓     private BigDecimal estatura;
29
30    ㉔     @Column(name="perimCintura")
31     ㉕     private BigDecimal perimCintura;
32
33    ㉖     @Column(name="perimCadera")
34     ㉗     private BigDecimal perimCadera;
35
36    ㉘     @Column(name="perimMuslo")
37     ㉙     private BigDecimal perimMuslo;
38
39    ㉚     @Column(name="perimBrazo")
40     ㉛     private BigDecimal perimBrazo;
41
42    ㉜     @Column(name="objetivo")
43     ㉝     private String objetivo;
44
45    ㉞
```

```
59
60    @Column(name="estinfo")
61    private String estado;
62
63    @Column(name="fechainfo")
64    private LocalDateTime fecha;
65
66    @ManyToMany(cascade = CascadeType.ALL)
67    @JoinTable(
68        name = "PacienteDieta",
69        joinColumns = @JoinColumn(name = "id_paciente", referencedColumnName ="idinfopaciente"),
70        inverseJoinColumns = @JoinColumn(name = "id_dieta", referencedColumnName ="iddieta"))
71    private List<Dieta> dieta;
72
73    public int getIdinfopaciente() {
74        return idinfopaciente;
75    }
76
77    public void setIdinfopaciente(int idinfopaciente) {
78        this.idinfopaciente = idinfopaciente;
79    }
80
81    public Usuario getPaciente() {
82        return paciente;
83    }
84
85    public void setPaciente(Usuario paciente) {
86        this.paciente = paciente;
87    }
88
89    public String getFrecEjercicios() {
90        return frecEjercicios;
91    }
92
93    public void setFrecEjercicios(String frecEjercicios) {
94        this.frecEjercicios = frecEjercicios;
95    }
96
97    public String getCondicion() {
98        return condicion;
99    }
100
101   public void setCondicion(String condicion) {
102       this.condicion = condicion;
103   }
```

- Horario

```
1 package com.DietasYRutinasOnline.entity;
2
3④ import java.time.LocalDateTime;[]
4
5
6
7
8
9 @Entity
10 @Table(name="Horario")
11 public class Horario {
12
13    @Id
14    @GeneratedValue(strategy = GenerationType.IDENTITY)
15    @Column(name = "idhorario")
16    private int idhorario;
17
18    @OneToOne
19    @JoinColumn(name = "idusuario")
20    private Usuario paciente;
21
22    @ManyToOne
23    @JoinColumn(name = "idrutina")
24    private Rutina rutina;
25
26    @Column(name="diaSemana")
27    private String dia;
28
29    @Column(name="periodo")
30    private String periodo;
31
32    @Column(name="esthorario")
33    private String estado;
34
35    @Column(name="descansoSerie")
36    private String descasSerie;
37
38    @Column(name="descansoEjer")
39    private String descasEjercicio;
40
41
42
43
44
45
46
47
48
49
50
51    public int getIdhorario() {
52        return idhorario;
53    }
54
55    public void setIdhorario(int idhorario) {
56        this.idhorario = idhorario;
57    }
58
59
```

- Reunión

```
1 package com.DietasYRutinasOnline.entity;
2
3
4④ import java.sql.Date;[]
5
6
7
8
9 @Entity
10 @Table(name="Reunion")
11 public class Reunion {
12
13    @Id
14    @GeneratedValue(strategy = GenerationType.IDENTITY)
15    @Column(name = "idreunion")
16    private int idreunion;
17
18    @ManyToOne
19    @JoinColumn(name = "idusuario")
20    private Usuario nutriologo;
21
22    @Column(name="motivo")
23    private String motivo;
24
25    @Column(name="diaSemana")
26    private String dia;
27
28    @Column(name="horaInicio")
29    private LocalDateTime hora;
30
31    @Column(name="enlace")
32    private String enlace;
33
34    @Column(name="estreunion")
35    private String estado;
36
37
38
39
40
41
42
43
44
45
46
```

```
4/
48@    public int getIdreunion() {
49        return idreunion;
50    }
51
52@    public void setIdreunion(int idreunion) {
53        this.idreunion = idreunion;
54    }
55
56@    public Usuario getNutriologo() {
57        return nutriologo;
58    }
59
60@    public void setNutriologo(Usuario nutriologo) {
61        this.nutriologo = nutriologo;
62    }
63
64@    public String getMotivo() {
65        return motivo;
66    }
67
68@    public void setMotivo(String motivo) {
69        this.motivo = motivo;
70    }
71
72@    public String getDia() {
73        return dia;
74    }
75
76@    public void setDia(String dia) {
77        this.dia = dia;
78    }
79
80@    public LocalTime getHora() {
81        return hora;
82    }
83
84@    public void setHora(LocalTime hora) {
85        this.hora = hora;
86    }
87
88@    public String getEnlace() {
89        return enlace;
90    }
91
92@    public void setEnlace(String enlace) {
93        this.enlace = enlace;
94    }
95
96@    public String getEstado() {
97        return estado;
98    }
99
100@   public void setEstado(String estado) {
101        this.estado = estado;
102    }
103
104 }
105
```

- **Asistencia**

```
1 package com.DietasYRutinasOnline.entity;
2
3④ import java.time.LocalDateTime;⑤
17
18 @Entity
19 @Table(name="Asistencia")
20 public class Asistencia {
21
22④     @Id
23     @GeneratedValue(strategy = GenerationType.IDENTITY)
24     @Column(name = "idasistencia")
25     private int idasistencia;
26
27④     @ManyToOne(cascade = CascadeType.ALL)
28     @JoinColumn(name = "idreunion")
29     private Reunion reunion;
30
31④     @ManyToOne(cascade = CascadeType.ALL)
32     @JoinColumn(name = "idusuario")
33     private Usuario paciente;
34
35④     @Column(name="fecasistencia")
36     private LocalDateTime fecha;
37
38④     @Column(name="estasistencia")
39     private String estado;
40
41④     public int getIdasistencia() {
42         return idasistencia;
43     }
44
45④     public void setIdasistencia(int idasistencia) {
46         this.idasistencia = idasistencia;
47     }
48
49④     public Reunion getReunion() {
50         return reunion;
51     }
```

- **Transacción**

```
3@ import java.time.LocalDateTime;_
13
14 @Entity
15 @Table(name="TransaccionGeneral")
16 public class Transaccion {
17
18@     @Id
19     @GeneratedValue(strategy = GenerationType.IDENTITY)
20     @Column(name = "idtransgeneral")
21     private int idregistro;
22
23@     @Column(name="fechatrans")
24     private LocalDateTime fecha;
25
26@     @Column(name="tipotrans")
27     private String tipo;
28
29@     @ManyToOne
30     @JoinColumn(name = "idusuario")
31     private Usuario usuario;
32
33@     @ManyToOne
34     @JoinColumn(name = "idrutina")
35     private Rutina rutina;
36
37@     @ManyToOne
38     @JoinColumn(name = "iddieta")
39     private Dieta dieta;
40
41@     @ManyToOne
42     @JoinColumn(name = "idhorario")
43     private Horario horario;
44
45@     @ManyToOne
46     @JoinColumn(name = "idinfopaciente")
47     private InfoPaciente infopaciente;
48
49@     @ManyToOne
50     @JoinColumn(name = "idreunion")
51     private Reunion reunion;
52
53@     @ManyToOne
54     @JoinColumn(name = "idasistencia")
55     private Asistencia asistencia;
--
```

- **TransaccionUsuario**

```
1 package com.DietaRutinasOnline.entity;
2
3+ import java.sql.Date;■
4
5 @Entity
6 @Table(name="TransaccionUsuario")
7 public class TransaccionUsuario {
8
9@     @Id
10    @GeneratedValue(strategy = GenerationType.IDENTITY)
11    @Column(name = "idtransaccion")
12    private int idtransaccion;
13
14@    @Column(name="fecharegistro")
15    private LocalDateTime registro;
16
17@    @Column(name="fechalogin")
18    private LocalDateTime login;
19
20@    @Column(name="fechalogout")
21    private LocalDateTime logout;
22
23@    @Column(name="cambioPassword")
24    private LocalDateTime cambioPassword;
25
26@    @Column(name="cambioCorreo")
27    private LocalDateTime cambioCorreo;
28
29@    @ManyToOne
30    @JoinColumn(name = "idusuario")
31    private Usuario usuario;
32
33@    @ManyToOne
34    @JoinColumn(name = "idtipousu")
35    private Tipousuario tipo;
36
37    public int getIdtransaccion() {
38        return idtransaccion;
39    }
40
41    public void setIdtransaccion(int idtransaccion) {
42        this.idtransaccion = idtransaccion;
43    }
44
45    public LocalDateTime getRegistro() {
46
47    }
```

- Notificación

```

1 package com.DietasYRutinasOnline.entity;
2
3④ import java.time.LocalDateTime;[]
4
5
6
7
8 @Entity
9 @Table(name="Notificacion")
10 public class Notificacion {
11
12     @Id
13     @GeneratedValue(strategy = GenerationType.IDENTITY)
14     @Column(name = "idnoti")
15     private int idnoti;
16
17     @ManyToOne
18     @JoinColumn(name = "idtrangeneral")
19     private Transaccion transaccion;
20
21     @Column(name="rolnoti")
22     private String rol;
23
24     @Column(name="mensaje")
25     private String mensaje;
26
27     @Column(name="estnoti")
28     private String estado;
29
30     @Column(name="timestamp")
31     private LocalDateTime timestamp;
32
33     @Column(name="diaSemana")
34     private String dia;
35
36     public int getIdnoti() {
37         return idnoti;
38     }
39
40     public void setIdnoti(int idnoti) {
41         this.idnoti = idnoti;
42     }
43
44     public Transaccion getTransaccion() {
45         return transaccion;
46     }
47
48     public void setTransaccion(Transaccion transaccion) {
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
948
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1096
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2218
2219
2220
2221
2222
2
```

```
1 package com.DietasYRutinasOnline.repository;
2
3+ import java.util.List;[]
4
5
6 @Repository
7 public interface TipoUsuarioRepository extends JpaRepository<TipoUsuario, Integer>{
8     TipoUsuario findByIdtipousu(int idtipousu);
9     TipoUsuario findByNomtipousu(String nomtipousu);
10 }
11
12
13
14
15
```

- **InfoPacienteRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3+ import java.util.List;[]
4
5
6 @Repository
7 public interface InfoPacienteRepository extends JpaRepository<InfoPaciente, Integer>{
8     InfoPaciente findByIdinfopaciente(int idinfopaciente);
9
10    List<InfoPaciente> findByPaciente(Usuario paciente);
11
12    InfoPaciente findByPacienteAndEstado(Usuario paciente, String estado);
13    List<InfoPaciente> findByEstado(String estado);
14    InfoPaciente findByDietaAndPaciente(Dieta dieta, Usuario paciente);
15
16 }
17
18
19
20
21 }
```

- **EjercicioRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3+ import java.util.List;[]
4
5
6 @Repository
7 public interface EjercicioRepository extends JpaRepository<Ejercicio, Integer>{
8     List<Ejercicio> findByGrupomuscular(String grupomuscular);
9
10
11    @Query("SELECT DISTINCT e.grupomuscular FROM Ejercicio e")
12    List<String> findDistinctGrupomuscular();
13
14
15 }
16
17
18
19
20
21
22
23
24
25
26
27 }
```

- **RutinaRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3+ import java.awt.print.Pageable;[]
4
5
6 @Repository
7 public interface RutinaRepository extends JpaRepository<Rutina, Integer>{
8     List<Rutina> findByEstado(String estado);
9
10
11    List<Rutina> findByNutriologo(Usuario nutriologo);
12    Rutina findByIdrutina(int idrutina);
13    List<Rutina> findByTipo(String tipo);
14    List<Rutina> findByNivel(String nivel);
15    List<Rutina> findByNivelAndTipo(String nivel, String tipo);
16
17
18    @Query("SELECT r FROM Rutina r WHERE r.nivel IN :niveles AND r.tipo = :tipo")
19    List<Rutina> findByNivelesAndTipo(@Param("niveles") List<String> niveles, String tipo);
20
21
22
23
24
25
26
27 }
```

- **AlimentoRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3 import java.util.List;
4
5
6 @Repository
7 public interface AlimentoRepository extends JpaRepository<Alimento, Integer>{
8     List<Alimento> findByTipo(String tipo);
9     @Query("SELECT DISTINCT a.tipo FROM Alimento a")
10    List<String> findDistinctTipos();
11
12 }
13
```

- **CondicionRepository**

```
1
2
3 import java.util.List;
4
5
6 @Repository
7 public interface CondicionRepository extends JpaRepository<Condicion, Integer>{
8     List<Condicion> findByEstado(String estado);
9     Condicion findByNombre(String nombre);
10 }
11
```

- **DietaRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3 import java.util.List;
4
5
6
7 public interface DietaRepository extends JpaRepository<Dieta, Integer>{
8     List<Dieta> findByEstado(String estado);
9     Dieta findByIdDieta(int iddieta);
10    List<Dieta> findByNutriologo(Usuario nutriologo);
11    List<Dieta> findByObjetivo(String objetivo);
12
13    List<Dieta> findByCondicion(Condicion condicion);
14    List<Dieta> findByCondicionNot(Condicion condicion);
15    List<Dieta> findByCondicionAndObjetivo(Condicion condicion, String objetivo);
16    List<Dieta> findByCondicionNotAndObjetivo(Condicion condicion, String objetivo);
17
18 }
19
```

- **HorarioRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3 import java.sql.Time;
4
5
6
7 @Repository
8 public interface HorarioRepository extends JpaRepository<Horario, Integer>{
9     List<Horario> findByPacienteAndEstado(Usuario paciente, String estado);
10    Horario findByPaciente(Usuario paciente);
11    Horario findByPacienteAndDiaAndEstado(Usuario paciente, String dia, String estado);
12    Horario findByPacienteAndDiaAndPeriodoAndEstado(Usuario paciente, String dia, String periodo, String estado);
13    Horario findByIdHorario(int idhorario);
14
15    List<Horario> findByDiaAndEstado(String dia, String estado);
16
17 }
```

- **ReunionRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3④ import java.util.List;[]
4
5 @Repository
6 public interface ReunionRepository extends JpaRepository<Reunion, Integer>{
7     List<Reunion> findByNutriologoAndEstado(Usuario nutriologo, String estado);
8     List<Reunion> findByDiaAndEstado(String dia, String estado);
9     Reunion findByIdreunion(int idreunion);
10
11     List<Reunion> findByNutriologo(Usuario nutriologo);
12     Reunion findByNutriologoAndDiaAndEstado(Usuario nutriologo, String dia, String estado);
13 }
14
15 }
```

- **AsistenciaRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3④ import java.util.List;[]
4
5 @Repository
6 public interface AsistenciaRepository extends JpaRepository<Asistencia, Integer>{
7     List<Asistencia> findByReunionAndEstado(Reunion reunion, String estado);
8
9     Asistencia findByPacienteAndReunion(Usuario paciente, Reunion reunion);
10    Asistencia findByPacienteAndReunionAndEstado(Usuario paciente, Reunion reunion, String estado);
11
12    Asistencia findByPaciente(Usuario paciente);
13    List<Asistencia> findByPacienteAndEstado(Usuario paciente, String estado);
14    List<Asistencia> findByPacienteAndReunionInAndEstado(Usuario paciente, List<Reunion> reuniones, String estado);
15
16    Long countByPacienteAndEstado(Usuario paciente, String estado);
17 }
18
19 }
```

- **TransUsuarioRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3④ import org.springframework.data.jpa.repository.JpaRepository;
4 import org.springframework.stereotype.Repository;
5
6 import com.DietasYRutinasOnline.entity.TransaccionUsuario;
7
8 @Repository
9 public interface TransUsuarioRepository extends JpaRepository<TransaccionUsuario, Integer> {
10 }
11
12 }
```

- **TransaccionRepository**

```
1 package com.DietasYRutinasOnline.repository;
2
3④ import java.util.List;[]
4
5 @Repository
6 public interface TransaccionRepository extends JpaRepository<Transaccion, Integer> {
7     List<Transaccion> findByTipoAndReunionIn(String tipo, List<Reunion> reunion);
8     List<Transaccion> findByTipoAndAsistenciaIn(String tipo, List<Asistencia> asistencia);
9
10 }
11
12 }
```

## Controladores:

- **UsuarioController**

```
3④ import java.time.LocalDateTime;⑤
40
41
42 @Controller
43 @RequestMapping("/usuario")
44 public class UsuarioController {
45
46④    @Autowired
47    UsuarioRepository usuarioRepository;
48
49④    @Autowired
50    TipoUsuarioRepository tipoUsuarioRepository;
51
52④    @Autowired
53    RutinaRepository rutinaRepository;
54
55④    @Autowired
56    DietaRepository dietaRepository;
57
58④    @Autowired
59    InfoPacienteRepository infoPacienteRepository;
60
61④    @Autowired
62    TransaccionRepository transaccionRepository;
63
64④    @Autowired
65    TransUsuarioRepository transUsuarioRepository;
66
67④    @Autowired
68    CondicionRepository condicionRepository;
69
70④    @Autowired
71    private PasswordEncoder passwordEncoder;
72
```

### - Iniciar sesión

```
1③
74④    @RequestMapping(value="/validarUsuario", method= RequestMethod.POST)
75    public String validarUsuario(
76        HttpServletRequest request,
77        @RequestParam("correo")String correo,
78        @RequestParam("password")String password,
79        Model model) {
80
81        Usuario objUsuario = usuarioRepository.findByCorreo(correo);
82        if (objUsuario!=null && passwordEncoder.matches(password, objUsuario.getPassword())) {
83            HttpSession sesion = request.getSession();
84            sesion.setAttribute("usuario", objUsuario);
85
86            TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
87            if(vistaUsuario!=null && vistaUsuario.getNomtipousu().equals("Paciente")) {
88                model.addAttribute("esPaciente", true);
89
90                TransaccionUsuario objTransUsuario = new TransaccionUsuario();
91                objTransUsuario.setLogin(LocalDateTime.now());
92                objTransUsuario.setUsuario(objUsuario);
93                objTransUsuario.setTipo(vistaUsuario);
94                transUsuarioRepository.save(objTransUsuario);
95
96                return "menu";
97            }
98
99            TransaccionUsuario objTransUsuario = new TransaccionUsuario();
100            objTransUsuario.setLogin(LocalDateTime.now());
101            objTransUsuario.setUsuario(objUsuario);
102            objTransUsuario.setTipo(vistaUsuario);
103            transUsuarioRepository.save(objTransUsuario);
104
105            return "menu";
106        }
107        model.addAttribute("error", "El correo electrónico y la contraseña no coinciden. Intenta de nuevo.");
108        return "index";
109    }
110
```

```

</head>
<body>
    <main>
        <section>
            <h2>Iniciar Sesión</h2>
            <form class="pure-form pure-form-stacked"
                th:action="@{/usuario/validarUsuario}"
                method="POST">

                <label for="aligned-email">Correo Electrónico</label>
                <input type="email" id="aligned-email" name="correo" required="" />

                <label for="aligned-password">Contraseña</label>
                <input type="password" id="aligned-password" name="password" required="" />

                <span class="span" th:if="${error}" th:text="${error}"></span>

                <input type="submit" class="pure-button pure-button-primary" value="Ingresar">
            </form>

            <br><br>
            <form th:action="@{/usuario/registrarCuenta}" method="POST">
                <button type="submit" class="btn-link">
                    ¿Eres nuevo aquí? Regístrate
                </button>
            </form>
        </section>
    </main>

    <footer>
        <p class="p-footer">2024. Todos los derechos reservados.</p>
    </footer>
</body>
</html>

```

## - Registrar Usuario

```

@PostMapping("/registrarCuenta")
public String registrarCuenta(HttpServletRequest request, Model model) {
    List<TipoUsuario> listaTiposUsuario = tipoUsuarioRepository.findAll();
    model.addAttribute("listaTiposUsuario", listaTiposUsuario);

    Usuario objUsuario = new Usuario();
    model.addAttribute("objUsuario", objUsuario);

    return "registrar";
}

@PostMapping("/grabarUsuario")
public String grabarUsuario(
    HttpServletRequest request,
    @RequestParam("correo") String correo,
    @ModelAttribute("objUsuario") Usuario objUsuario,
    @ModelAttribute("nomtipousu") String nomtipousu,
    Model model) {

    try {
        Usuario correoExiste = usuarioRepository.findByCorreo(correo);
        if (correoExiste!=null) []
            model.addAttribute("error", "Hubo un error al llenar correo electrónico, intente de nuevo.");

        List<TipoUsuario> listaTiposUsuario = tipoUsuarioRepository.findAll();
        model.addAttribute("listaTiposUsuario", listaTiposUsuario);

        model.addAttribute("objUsuario", objUsuario);
        return "registrar";
    }
    String encryptedPassword = passwordEncoder.encode(objUsuario.getPassword());
    objUsuario.setPassword(encryptedPassword);

    objUsuario.setEstado("Activo");
    usuarioRepository.save(objUsuario);

    HttpSession sesion = request.getSession();
    sesion.setAttribute("usuario", objUsuario);

    TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
    boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
    model.addAttribute("esPaciente", esPaciente);
}

```

```

        if(esPaciente) {
            InfoPaciente objCuestionario = new InfoPaciente();
            model.addAttribute("objCuestionario", objCuestionario);

            TransaccionUsuario objTransUsuario = new TransaccionUsuario();
            objTransUsuario.setUsuario(objUsuario);
            objTransUsuario.setTipo(vistaUsuario);
            transUsuarioRepository.save(objTransUsuario);

            return "cuestionario";
        }

        TransaccionUsuario objTransUsuario = new TransaccionUsuario();
        objTransUsuario.setRegistro(LocalDateTime.now());
        objTransUsuario.setUsuario(objUsuario);
        objTransUsuario.setTipo(vistaUsuario);
        transUsuarioRepository.save(objTransUsuario);

        return "menu";
    }
    catch(Exception ex) {
        return "registrar";
    }
}

```

## VISTA

```

<section>
    <h2>Registrate</h2>
    <form class="pure-form pure-form-stacked"
        th:action="@{/usuario/grabarUsuario}"
        th:object="${objUsuario}"
        method="POST" >

        <label>¿Eres paciente o nutriólogo?</label>
        <div th:each="tipos : ${listaTiposUsuario}">
            <label th:for="'tipoUsuario-' + ${tipos.idtipousu}"
                th:text="${tipos.nomtipousu}">Tipo de Usuario</label>
            <input type="radio"
                th:field="*{tipousuario.idtipousu}"
                th:value="${tipos.idtipousu}"
                id="tipoUsuario-[${tipos.idtipousu}]" />
        </div>

        <label for="nombres_apellidos">Nombres</label>
        <input
            type="text"
            id="multi-first-name"
            th:field="*{nombres}"
            required="">

        <label for="nombres_apellidos">Apellidos</label>
        <input
            type="text"
            id="multi-last-name"
            th:field="*{apellidos}"
            required="">
    </form>

```

```

<label for="nombres_apellidos">Fecha nacimiento</label>
<input
    type="date"
    id="multi-fecha-nacimiento"
    th:field="*{fechanacimiento}"
    required="" />

<label for="percentage">Nacionalidad</label>
<select id="percentage" name="nacionalidad">
    <option th:value="Otro" selected="selected">Otro</option>
    <option th:value="'Argentina'">Argentina</option>
    <option th:value="'Bolivia'">Bolivia</option>
    <option th:value="'Chile'">Chile</option>
    <option th:value="'Colombia'">Colombia</option>
    <option th:value="'Ecuador'">Ecuador</option>
    <option th:value="'España'">España</option>
    <option th:value="'Mexico'">Mexico</option>
    <option th:value="'Paraguay'">Paraguay</option>
    <option th:value="'Perú'">Perú</option>
    <option th:value="'Uruguay'">Uruguay</option>
    <option th:value="'Venezuela'">Venezuela</option>
</select>

<label for="percentage">Sexo</label>
<select id="percentage" name="sexo">
    <option th:value="Masculino" selected="selected">Masculino</option>
    <option th:value="Femenino">Femenino</option>
</select>

<label for="nombres_apellidos">Correo Electronico</label>
<input
    type="email"
    id="multi-email"
    class="pure-u-23-24"
    th:field="*{correo}"
    required="" />

<label for="nombres_apellidos">Contraseña</label>
<input
    type="password"
    id="aligned-password"
    th:field="*{password}"
    minlength="8"
    required=""
    />

<span class="span" th:if="${error}" th:text="${error}"/>

<input type="submit" value="Enviar">

</form>
</section>
</main>

```

## - Cuestionario (PACIENTE)

```

@PostMapping("/grabarCuestionario")
public String grabarCuestionario(
    HttpSession sesion,
    @ModelAttribute("objCuestionario") InfoPaciente objCuestionario,
    Model model) {

    Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
    if (objUsuario==null) {
        Usuario usuRegistrado = usuarioRepository.findById(objUsuario.getIdusuario()).orElse(null);
        objCuestionario.setPaciente(usuRegistrado);
        objCuestionario.setEstado("Activo");
        objCuestionario.setFecha(LocalDateTime.now());
        infoPacienteRepository.save(objCuestionario);

        TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
        boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
        model.addAttribute("esPaciente", esPaciente);
    }
}

```

## (Método que se usará cuando un paciente quiere consultar las listas de dietas y rutinas)

```
201  /* RECOMENDACIÓN INICIAL */
202
203  /* INFORMACION DE PACIENTE */
204  model.addAttribute("miInfo", objCuestionario);
205
206  /* RUTINAS */
207  List<Rutina> listaRutinas;
208  listaRutinas = rutinaRepository.findByEstado("Activo");
209  model.addAttribute("listaRutinas", listaRutinas);
210
211  if(objCuestionario!=null && objCuestionario.getObjetivo().equals("Deficit")) {
212      listaRutinas = rutinaRepository.findByTipo("Deficit");
213      model.addAttribute("listaRutinas", listaRutinas);
214
215      if(objCuestionario.getFrecEjercicios().equals("A veces o nada") || objCuestionario.getFrecEjercicios().equals("Semanalmente")) {
216
217          listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Deficit");
218          model.addAttribute("listaRutinas", listaRutinas);
219      }
220      else if(objCuestionario.getFrecEjercicios().equals("30 minutos a diario")) {
221
222          List<String> niveles = Arrays.asList("Principiante", "Intermedio");
223          listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
224          model.addAttribute("listaRutinas", listaRutinas);
225      }
226
227  }
228
229  else if(objCuestionario!=null && objCuestionario.getObjetivo().equals("Volumen")) {
230      listaRutinas = rutinaRepository.findByTipo("Volumen");
231      model.addAttribute("listaRutinas", listaRutinas);
232
233      if(objCuestionario.getFrecEjercicios().equals("A veces o nada") || objCuestionario.getFrecEjercicios().equals("Semanalmente")) {
234
235          listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Volumen");
236          model.addAttribute("listaRutinas", listaRutinas);
237      }
238      else if(objCuestionario.getFrecEjercicios().equals("30 minutos diario")) {
239
240          List<String> niveles = Arrays.asList("Principiante", "Intermedio");
241          listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Volumen");
242          model.addAttribute("listaRutinas", listaRutinas);
243
244      }
245
246  else if (objCuestionario!=null){
247      if(objCuestionario.getFrecEjercicios().equals("A veces o nada") || objCuestionario.getFrecEjercicios().equals("Semanalmente")) {
248
249          listaRutinas = rutinaRepository.findByNivel("Principiante");
250          model.addAttribute("listaRutinas", listaRutinas);
251      }
252      else if(objCuestionario.getFrecEjercicios().equals("30 minutos a diario")) {
253
254          List<String> niveles = Arrays.asList("Principiante", "Intermedio");
255          listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
256          model.addAttribute("listaRutinas", listaRutinas);
257      }
258  }
259
260  /* DIETAS */
261  List<Dieta> listaDiетas;
262  listaDiетas = dietaRepository.findByEstado("Activo");
263  model.addAttribute("listaDiетas", listaDiетas);
264
265  if(objCuestionario!=null && objCuestionario.getObjetivo().equals("Deficit")) {
266      listaDiетas = dietaRepository.findByObjetivo("Deficit");
267      model.addAttribute("listaDiетas", listaDiетas);
268
269      if(objCuestionario.getCondicion().equals("Lacteos")) {
270          Condicion lacteos = condicionRepository.findByNombre("Lacteos");
271          model.addAttribute("lacteos", lacteos);
272
273          listaDiетas = dietaRepository.findByCondicionNot(lacteos);
274          model.addAttribute("listaDiетas", listaDiетas);
275      }
276
277      else if(objCuestionario.getCondicion().equals("Gluten")) {
278          Condicion gluten = condicionRepository.findByNombre("Gluten");
279
280          listaDiетas = dietaRepository.findByCondicionAndObjetivo(gluten, "Deficit");
281          model.addAttribute("listaDiетas", listaDiетas);
282      }
283
284      else if(objCuestionario.getCondicion().equals("Vegano")) {
285          Condicion carne = condicionRepository.findByNombre("Carne");
286          model.addAttribute("carne", carne);
287  }
```

```
306     Condicion gluten = condicionRepository.findByNombre("Gluten");
307
308     listaDietas = dietaRepository.findByCondicionNotAndObjetivo(gluten, "Volumen");
309     model.addAttribute("listaDietas", listaDietas);
310 }
311
312 else if(objCuestionario.getCondicion().equals("Vegano")) {
313     Condicion carne = condicionRepository.findByNombre("Carne");
314     model.addAttribute("carne", carne);
315
316     listaDietas = dietaRepository.findByCondicionNot(carne);
317     model.addAttribute("listaDietas", listaDietas);
318 }
319
320 else if (objCuestionario!=null){
321
322     if(objCuestionario.getCondicion().equals("Lacteos")) {
323         Condicion lacteos = condicionRepository.findByNombre("Lacteos");
324         model.addAttribute("lacteos", lacteos);
325
326         listaDietas = dietaRepository.findByCondicionNot(lacteos);
327         model.addAttribute("listaDietas", listaDietas);
328     }
329
330     else if(objCuestionario.getCondicion().equals("Gluten")) {
331         Condicion gluten = condicionRepository.findByNombre("Gluten");
332         model.addAttribute("gluten", gluten);
333
334         listaDietas = dietaRepository.findByCondicionNot(gluten);
335         model.addAttribute("listaDietas", listaDietas);
336     }
337
338     else if(objCuestionario.getCondicion().equals("Vegano")) {
339         Condicion carne = condicionRepository.findByNombre("Carne");
340         model.addAttribute("carne", carne);
341
342         listaDietas = dietaRepository.findByCondicionNot(carne);
343         model.addAttribute("listaDietas", listaDietas);
344     }
345 }
346
347 return "recomendacion_inicial";
348
349
350 }
```

```
<h2>Cuestionario</h2>
<form class="pure-form pure-form-stacked"
      th:action="@{/usuario/grabarCuestionario}"
      th:object="${objCuestionario}"
      method="POST" >

    <span class="span">Este cuestionario nos ayudara a saber más de ti.<br>
        También puede editarlo para hacer seguimiento.
    </span>

    <input type="hidden" name="usuRegistrado" th:value="${usuRegistrado}"/>
    <span class="span" th:if="${exito}" th:text="${usuRegistrado}"></span>

    <label for="percentage">¿Con qué frecuencia haces ejercicios?</label>
    <select id="percentage" name="frecEjercicios">
        <option th:value="A veces o nada!" selected="selected">A veces o nada!</option>
        <option th:value="30 minutos a diario">30 minutos a diario</option>
        <option th:value="1 hora a diario">1 hora a diario</option>
        <option th:value="1 hora algunos dias">1 hora algunos días</option>
        <option th:value="2 horas o más">2 horas o más</option>
        <option th:value="Semanalmente">Semanalmente</option>
    </select>

    <label for="percentage">¿Tienes una condición alimenticia?</label>
    <select id="percentage" name="condicion">
        <option th:value="Ninguna" selected="selected">Ninguna</option>
        <option th:value="Gluten">Gluten</option>
        <option th:value="Lacteos">Lacteos</option>
        <option th:value="Vegano">Vegano</option>
        <option th:value="Otra">Otra</option>
    </select>

    <label for="mensaje">¿Tienes un objetivo?</label>
    <select id="percentage" name="objetivo">
        <option th:value="Volumen" selected="selected">Volumen</option>
        <option th:value="Deficit">Deficit</option>
        <option th:value="Mejorar Habitos">Ninguno, solo cambiar y mejorar mis hábitos</option>
    </select>
```

```

<br>
<span class="span">Si aún no sabe sus medidas, puede saltarte esta parte y luego llenarlas en otro momento.</span>

<label for="nombres_apellidos">Peso corporal</label>
<input type="number" id="multi-Last-name" th:field="*{pesocorporal}" step="0.01" />

<label for="estatura">Estatura</label>
<input type="number" id="multi-Last-name" th:field="*{estatura}" step="0.01"/>

<label for="nombres_apellidos">Perímetro de cintura</label>
<input type="number" id="multi-Last-name" th:field="*{perimCintura}" step="0.01" />

<label for="nombres_apellidos">Perímetro de cadera</label>
<input type="number" id="multi-Last-name" th:field="*{perimCadera}" step="0.01" />

<label for="nombres_apellidos">Perímetro de muslo</label>
<input type="number" id="multi-Last-name" th:field="*{perimMuslo}" step="0.01" />

<label for="nombres_apellidos">Perímetro de brazo/bicep</label>
<input type="number" id="multi-Last-name" th:field="*{perimBrazo}" step="0.01" />

<input type="submit" value="Grabar">

</form>
</section>

```

- **Cierre de sesión**

```

@GetMapping("/cerrarSesion")
public String cerrarSesion(HttpSession sesion, Model model) {
    Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");

    TransaccionUsuario objTransUsuario = new TransaccionUsuario();
    objTransUsuario.setLogout(LocalDateTime.now());
    objTransUsuario.setUsuario(objUsuario);
    objTransUsuario.setTipo(objUsuario.getTipousuario());
    transUsuarioRepository.save(objTransUsuario);

    sesion.invalidate();
    return "index";
}

```

- **Actualizar contraseña**

```

@RequestMapping(value = "/nuevaContraseña", method = RequestMethod.POST)
public String nuevaContraseña(
    HttpSession sesion,
    @ModelAttribute("objUsuario") Usuario objUsuario,
    @RequestParam(value="repetirPassword") String repetirPassword,
    Model model) {

    Usuario newPassword = (Usuario) sesion.getAttribute("usuario");
    if (newPassword!=null && objUsuario.getPassword().equals(repetirPassword)) {
        String encryptedPassword = passwordEncoder.encode(objUsuario.getPassword());
        newPassword.setPassword(encryptedPassword);
        usuarioRepository.save(newPassword);

        sesion.invalidate();
        return "index";
    }
    model.addAttribute("errorcontraseña", "Hubo un error al actualizar la nueva contraseña.");
    return "seguridad";
}

```

- **Cambiar correo electrónico**

```

    @RequestMapping(value = "/nuevaContraseña", method = RequestMethod.POST)
    public String nuevaContraseña(
        HttpSession sesion,
        @ModelAttribute("objUsuario") Usuario objUsuario,
        @RequestParam(value="repetirPassword") String repetirPassword,
        Model model) {

        Usuario nuevaPassword = (Usuario) sesion.getAttribute("usuario");
        if (nuevaPassword!=null && objUsuario.getPassword().equals(repetirPassword)) {
            String encryptedPassword = passwordEncoder.encode(objUsuario.getPassword());
            nuevaPassword.setPassword(encryptedPassword);
            usuarioRepository.save(nuevaPassword);

            sesion.invalidate();
            return "index";
        }
        model.addAttribute("errorcontraseña", "Hubo un error al actualizar la nueva contraseña.");
        return "seguridad";
    }

```

## VISTA DE LAS TRES FUNCIONALIDADES

```

<h2>Ajustes de cuenta</h2>
<br>

<h3>Cambiar contraseña</h3>
<form class="pure-form pure-form-stacked"
      th:action="@{/usuario/nuevaContraseña}"
      th:object="${objUsuario}"
      method="POST">

    <!-- <label for="aligned-email">Correo Electronico</label>
    <input type="email" id="aligned-email" name="correo" required="" />-->

    <label for="aligned-password">Nueva contraseña</label>
    <input type="password" id="aligned-password" th:field="*{password}" minlength="8" required="">

    <label for="aligned-password">Confirmar contraseña</label>
    <input type="password" id="aligned-password" name="repetirPassword" minlength="8" required="">

    <span class="span" th:if="${error}" th:text="${error}"></span>

    <input type="submit" value="Guardar">
</form>

<h3>Cambiar Correo electronico</h3>
<form class="pure-form pure-form-stacked"
      th:action="@{/usuario/cambiarCorreo}"
      th:object="${objUsuario}"
      method="POST">

    <label for="aligned-email">Correo Electronico</label>
    <input type="email" id="aligned-email" th:field="*{correo}" required="">

    <span class="span" th:if="${errorcorreo}" th:text="${errorcorreo}"></span>

    <input type="submit" value="Guardar">
</form>

<h3>¿Quieres cerrar sesión?</h3>
<p><a type="button"
       class="pure-button pure-button-secondary"
       style="background-color: red; color: white"
       th:href="@{/usuario/cerrarsesion}">Cerrar Sesión</a></p>

```

- Editar Perfil

```
1 @PostMapping("/actualizarPerfil")
2 public String actualizarPerfil(
3     HttpSession sesion,
4     @ModelAttribute("objUsuario") Usuario objUsuario,
5     Model model) {
6
7     Usuario perfilActual = (Usuario) sesion.getAttribute("usuario");
8     if (objUsuario!=null) {
9         perfilActual.setNombres(objUsuario.getNombres());
10        perfilActual.setNacionalidad(objUsuario.getNacionalidad());
11        perfilActual.setBiografia(objUsuario.getBiografia());
12        usuarioRepository.save(perfilActual);
13
14        TipoUsuario vistaUsuario = perfilActual.getTipousuario();
15        boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
16        model.addAttribute("esPaciente", esPaciente);
17    }
18    model.addAttribute("actualizado", "Su perfil se actualizó con éxito");
19    model.addAttribute("objUsuario", perfilActual);
20
21    InfoPaciente miInfo = infoPacienteRepository.findByPacienteAndEstado(perfilActual, "Activo");
22    model.addAttribute("miInfo", miInfo);
23
24    List<Rutina> misRutinas = rutinaRepository.findByNutriologo(perfilActual);
25    model.addAttribute("misRutinas", misRutinas);
26
27    List<Dieta> misDietas = dietaRepository.findByNutriologo(perfilActual);
28    model.addAttribute("misDietas", misDietas);
29
30    return "menu";
31 }
32
33
34
35
36
37
38
39
40
41
42
43
44
```

19@ <body>
20@ <main>
21@ <section>
22@ <h2>Editar Perfil</h2>
23@ <form class="pure-form pure-form-stacked"
24@ th:action="@{/usuario/actualizarPerfil}"
25@ th:object="\${objUsuario}"
26@ method="POST" >
27
28 <label for="nombres\_apellidos">Nombres</label>
29 <input
30 type="text"
31 id="multi-first-name"
32 th:field="#{nombres}"
33 required="" />
34
35
36 <label for="mensaje">Biografía</label>
37 <textarea id="mensaje" th:field="\*{biografia}"></textarea>
38
39 <input type="submit" value="Guardar">
40
41 </form>
42 </section>
43 </main>
44

- **HomeController**

```
1 package com.DietaRutinasOnline.controller;
2
3④ import java.awt.print.Pageable;⑤
4
5
6 @Controller
7 @RequestMapping("/home")
8 public class HomeController {
9
10    @Autowired
11    UsuarioRepository usuarioRepository;
12
13    @Autowired
14    RutinaRepository rutinaRepository;
15
16    @Autowired
17    DietaRepository dietaRepository;
18
19    @Autowired
20    TipoUsuarioRepository tipoUsuarioRepository;
21
22    @Autowired
23    InfoPacienteRepository infoPacienteRepository;
24
25    @Autowired
26    HorarioRepository horarioRepository;
27
28    @Autowired
29    ReunionRepository reunionRepository;
30
31    @Autowired
32    AsistenciaRepository asistenciaRepository;
33
34    @Autowired
35    NotificacionRepository notificacionRepository;
36
37    @Autowired
38    CondicionRepository condicionRepository;
39
40    @Autowired
41    TransaccionRepository transaccionRepository;
42
```

## - Ver Perfil

```
279④     @GetMapping("/verPerfil")
280     public String verPerfil(
281         HttpSession sesion,
282         Model model) {
283         Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
284         if (objUsuario!=null) {
285             TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
286             boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
287             model.addAttribute("esPaciente", esPaciente);
288         }
289         model.addAttribute("objUsuario", objUsuario);
290
291         /* SI ES PACIENTE*/
292         InfoPaciente miInfo = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
293         model.addAttribute("miInfo", miInfo);
294
295         /* SI ES NUTRIOLOGO */
296         List<Reunion> objReunion = reunionRepository.findByNutriologoAndEstado(objUsuario, "Activo");
297         model.addAttribute("objReunion", objReunion);
298
299         List<Rutina> misRutinas = rutinaRepository.findByNutriologo(objUsuario);
300         model.addAttribute("misRutinas", misRutinas);
301
302         List<Dieta> misDietas = dietaRepository.findByNutriologo(objUsuario);
303         model.addAttribute("misDietas", misDietas);
304
305         return "perfil";
306     }
307
308 }
```

## - Ajustes de cuenta

```

309     @GetMapping("/ajustesCuenta")
310     public String ajustesCuenta(HttpServletRequest sesion, Model model){
311         Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
312         if (objUsuario!=null) {
313             model.addAttribute("objUsuario", objUsuario);
314             return "usuario/seuridad";
315         }
316         else {
317             return "index";
318         }
319     }

```

- **Ver nutriólogos (PACIENTE)**

```

@GetMapping("/verNutriologos")
public String verNutriologos(
    HttpServletRequest sesion,
    @ModelAttribute("objNutriologo") Usuario objNutriologo,
    Model model) {
    Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
    if (objUsuario!=null) {
        TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
        boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
        model.addAttribute("esPaciente", esPaciente);
    }
    TipoUsuario rolNutriologo = tipoUsuarioRepository.findByNomtipousu("Nutriologo");

    List<Usuario> listaNutriologos = usuarioRepository.findByTipousuarioAndEstado(rolNutriologo, "Activo");
    model.addAttribute("listaNutriologos", listaNutriologos);

    return "usuario/lista_nutriologos";
}

<section style="background-color: #fff">
    <h2>Nuestros doctores</h2>

    <div>
        <table class="pure-table">
            <thead>
                <tr>
                    <th>Nombre</th>
                    <th>Apellido</th>
                    <th>Biografía</th>
                    <th>Reuniones</th>
                    <th></th>
                </tr>
            </thead>
            <tbody class="pure-table-border">
                <tr th:each="nutriologo: ${listaNutriologos}">
                    <td><span th:text="${nutriologo.nombres}"></span></td>
                    <td><span th:text="${nutriologo.apellidos}"></span></td>
                    <td><span th:text="${nutriologo.biografia}"></span></td>
                    <td><a th:href="@{/reunion/verReuniones(idusuario=${nutriologo.idusuario})}">Ver Reuniones</a></td>
                    <td><a th:href="@{/reunion/perfilNutriologo(idusuario=${nutriologo.idusuario})}">Ver su Perfil</a></td>
                </tr>
            </tbody>
        </table>
    </div>

```

- **Ver rutinas**

```
113④ @GetMapping("/verRutinas")
114 public String verRutinas(
115     HttpSession sesion,
116     @ModelAttribute("objRutina") Rutina objRutina,
117     Model model) {
118     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
119     if (objUsuario!=null) {
120         Tipousuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
121         boolean espaciente = vistaUsuario.getNomtipousu().equals("Paciente");
122         model.addAttribute("esPaciente", espaciente);
123     }
124     InfoPaciente pacienteActual = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
125     model.addAttribute("pacienteActual", pacienteActual);
126
127     List<Rutina> listaRutinas;
128     listaRutinas = rutinaRepository.findByEstado("Activo");
129     model.addAttribute("listaRutinas", listaRutinas);
130
131     /* VISTA PARA LOS PACIENTES*/
132     if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Deficit")) {
133         listaRutinas = rutinaRepository.findByTipo("Deficit");
134         model.addAttribute("listaRutinas", listaRutinas);
135
136         if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
137             listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Deficit");
138             model.addAttribute("listaRutinas", listaRutinas);
139         } else if(pacienteActual.getFrecEjercicios().equals("30 minutos a diario") || pacienteActual.getFrecEjercicios().equals("1 hora algunos dias")) {
140             List<String> niveles = Arrays.asList("Principiante", "Intermedio");
141             listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
142             model.addAttribute("listaRutinas", listaRutinas);
143         }
144     }
145
146     else if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Volumen")) {
147         listaRutinas = rutinaRepository.findByTipo("Volumen");
148         model.addAttribute("listaRutinas", listaRutinas);
149
150         else if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Volumen")) {
151             listaRutinas = rutinaRepository.findByTipo("Volumen");
152             model.addAttribute("listaRutinas", listaRutinas);
153
154             if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
155                 listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Volumen");
156                 model.addAttribute("listaRutinas", listaRutinas);
157             } else if(pacienteActual.getFrecEjercicios().equals("30 minutos diario") || pacienteActual.getFrecEjercicios().equals("1 hora algunos dias")) {
158                 List<String> niveles = Arrays.asList("Principiante", "Intermedio");
159                 listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Volumen");
160                 model.addAttribute("listaRutinas", listaRutinas);
161             }
162         }
163     }
164
165     else if (pacienteActual!=null){
166         if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
167             listaRutinas = rutinaRepository.findByNivel("Principiante");
168             model.addAttribute("listaRutinas", listaRutinas);
169         } else if(pacienteActual.getFrecEjercicios().equals("30 minutos a diario") || pacienteActual.getFrecEjercicios().equals("1 hora algunos dias")) {
170             List<String> niveles = Arrays.asList("Principiante", "Intermedio");
171             listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
172             model.addAttribute("listaRutinas", listaRutinas);
173         }
174     }
175
176     return "rutinas/menu_rutinas";
177 }
178
179
180
181
182
183
184
185
186 }
```

```
<div class="deslisable">
    <div class="pure-menu pure-menu-horizontal">
        <ul class="pure-menu-list">
            <li class="pure-menu-item pure-menu-selected">
                <a href="#" class="pure-menu-link">Rutinas</a>
            </li>
            <li class="pure-menu-item">
                <a th:href="@{/rutina/verEjercicios}" class="pure-menu-link">Ejercicios</a>
            </li>
        </ul>
    </div>

    <table class="pure-table">
        <thead>
            <tr>
                <th>Nombre de la Rutina</th>
                <th>Tipo</th>
                <th>Parte del cuerpo</th>
                <th>Nivel</th>
                <th>Descripción</th>
                <th>Acciones</th>
            </tr>
        </thead>

        <tbody class="pure-table-border">
            <tr th:each="rutina: ${ListaRutinas}">
                <td><span th:text="${rutina.nombre}"></span></td>
                <td><span th:text="${rutina.tipo}"></span></td>
                <td><span th:text="${rutina.partecuerpo}"></span></td>
                <td><span th:text="${rutina.nivel}"></span></td>
                <td><span th:text="${rutina.descripcion}"></span></td>
                <td><a th:href="@{/rutina/verDetalleRutina(idrutina=${rutina.idrutina})}">Ver detalles</a><br>
                </td>
            </tr>
        </tbody>
    </table>
</div>
<form th:if="${!esPaciente}" th:action="@{/rutina/crearRutina}" method="POST">
    <button type="submit" class="pure-button pure-button-primary">
        Crear Nueva rutina
    </button>
</form>
<a type="button" class="pure-button pure-button-secondary" th:href="@{/home/retroceder}">Retroceder</a>
</section>
</main>
<footer>
```

## - Ver Dietas

```
188@  
189    @GetMapping("/verDiетas")  
190    public String gestionarPartidos(  
191        HttpSession sesion,  
192        @ModelAttribute("objDieta") Dieta objDieta,  
193        Model model) {  
194        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");  
195        if (objUsuario!=null) {  
196            Tipousuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());  
197            boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");  
198            model.addAttribute("esPaciente", esPaciente);  
199        }  
200        InfoPaciente pacienteActual = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");  
201        model.addAttribute("pacienteActual", pacienteActual);  
202  
203        List<Dieta> listaDiетas;  
204        listaDiетas = dietaRepository.findByEstado("Activo");  
205        model.addAttribute("listaDiетas", listaDiетas);  
206  
207        /* VISTA PARA LOS PACIENTES*/  
208        if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Deficit")) {  
209            listaDiетas = dietaRepository.findByObjetivo("Deficit");  
210            model.addAttribute("listaDiетas", listaDiетas);  
211  
212            if(pacienteActual.getCondicion().equals("Lacteos")) {  
213                Condicion lacteos = condicionRepository.findByNombre("Lacteos");  
214                model.addAttribute("lacteos", lacteos);  
215  
216                listaDiетas = dietaRepository.findByCondicion(lacteos);  
217                model.addAttribute("listaDiетas", listaDiетas);  
218            }  
219  
220            else if(pacienteActual.getCondicion().equals("Gluten")) {  
221                Condicion gluten = condicionRepository.findByNombre("Gluten");  
222  
223                listaDiетas = dietaRepository.findByCondicionAndObjetivo(gluten, "Deficit");  
224                model.addAttribute("listaDiетas", listaDiетas);  
225            }  
226        }  
227  
228        Condicion lacteos = condicionRepository.findByIdByName("Lacteos");  
229  
230        listaDiетas = dietaRepository.findByCondicion(lacteos);  
231        model.addAttribute("listaDiетas", listaDiетas);  
232  
233        else if(pacienteActual.getCondicion().equals("Gluten")) {  
234            Condicion gluten = condicionRepository.findByNombre("Gluten");  
235  
236            listaDiетas = dietaRepository.findByCondicionAndObjetivo(gluten, "Volumen");  
237            model.addAttribute("listaDiетas", listaDiетas);  
238        }  
239  
240        else if (pacienteActual!=null){  
241            listaDiетas = dietaRepository.findByEstado("Activo");  
242            model.addAttribute("listaDiетas", listaDiетas);  
243  
244            if(pacienteActual.getCondicion().equals("Lacteos")) {  
245                Condicion lacteos = condicionRepository.findByNombre("Lacteos");  
246                model.addAttribute("lacteos", lacteos);  
247  
248                listaDiетas = dietaRepository.findByCondicionNot(lacteos);  
249                model.addAttribute("listaDiетas", listaDiетas);  
250            }  
251  
252            else if(pacienteActual.getCondicion().equals("Gluten")) {  
253                Condicion gluten = condicionRepository.findByNombre("Gluten");  
254                model.addAttribute("gluten", gluten);  
255  
256                listaDiетas = dietaRepository.findByCondicionNot(gluten);  
257                model.addAttribute("listaDiетas", listaDiетas);  
258            }  
259  
260            else if(pacienteActual.getCondicion().equals("Vegano")) {  
261                Condicion carne = condicionRepository.findByNombre("Carne");  
262                model.addAttribute("carne", carne);  
263  
264                listaDiетas = dietaRepository.findByCondicionNot(carne);  
265                model.addAttribute("listaDiетas", listaDiетas);  
266            }  
267  
268            else if(pacienteActual.getCondicion().equals("Carnes")) {  
269                Condicion carnes = condicionRepository.findByNombre("Carnes");  
270                model.addAttribute("carnes", carnes);  
271  
272                listaDiетas = dietaRepository.findByCondicionNot(carnes);  
273                model.addAttribute("listaDiетas", listaDiетas);  
274            }  
275  
276        }  
277    }  
278  
279    return "diетas/menu_diетas";  
280}
```

```

<div class="deslisable">

    <div class="pure-menu pure-menu-horizontal">
        <ul class="pure-menu-list">
            <li class="pure-menu-item pure-menu-selected">
                <a href="#" class="pure-menu-link">Dietas</a>
            </li>
            <li class="pure-menu-item">
                <a th:href="@{/dieta/verAlimentos}" class="pure-menu-link">Alimentos</a>
            </li>
        </ul>
    </div>

    <table class="pure-table">
        <thead>
            <tr>
                <th>Nombre</th>
                <th>Objetivo</th>
                <th>Descripción</th>
                <th>Acciones</th>
            </tr>
        </thead>

        <tbody class="pure-table-border">
            <tr th:each="dieta: ${listadietas}">
                <td><span th:text="${dieta.nombre}"></span></td>
                <td><span th:text="${dieta.objetivo}"></span></td>
                <td><span th:text="${dieta.descripcion}"></span></td>
                <td><a th:href="@{/dieta/verDetalleDieta(iddieta=${dieta.iddieta})}">Ver detalles</a>
                    <br>
                    <form th:if="${esPaciente}" th:action="@{/dieta/seguirDieta(iddieta=${dieta.iddieta})}" method="POST">
                        <button type="submit" class="pure-button pure-button-primary">
                            Seguir Dieta
                        </button>
                    </form>
                </td>
            </tr>
        </tbody>
    </table>
</div>

<form th:if="${!esPaciente}" th:action="@{/dieta/crearDieta}" method="POST">
    <button type="submit" class="pure-button pure-button-primary">
        Crear Nueva Dieta
    </button>
</form>

</div>

```

## - Ver mi horario (PACIENTE)

```

321 @GetMapping("/verHorario")
322 public String verHorario(HttpServletRequest sesion, Model model) {
323     Usuario objusuario = (Usuario) sesion.getAttribute("usuario");
324     if (objusuario!=null) {
325         Tipousuario vistausuario = tipousuarioRepository.findByIdtipousu(objusuario.getTipousuario().getIdtipousu());
326         boolean esPaciente = vistausuario.getNomtipousu().equals("Paciente");
327         model.addAttribute("esPaciente", esPaciente);
328     }
329     List<Horario> miHorario = horarioRepository.findByPacienteAndEstado(objusuario, "Activo");
330     model.addAttribute("miHorario", miHorario);
331
332     List<Rutina> cbxRutinas = rutinaRepository.findAll();
333     model.addAttribute("listaRutinas", cbxRutinas);
334
335     Horario objHorario = new Horario();
336     model.addAttribute("objHorario", objHorario);
337
338     /* RECUPERAR LISTA DE RUTINAS DEBAJO DEL HORARIO */
339     InfoPaciente pacienteActual = infoPacienteRepository.findByPacienteAndEstado(objusuario, "Activo");
340     model.addAttribute("pacienteActual", pacienteActual);
341
342     List<Rutina> listaRutinas;
343     listaRutinas = rutinaRepository.findByEstado("Activo");
344     model.addAttribute("listaRutinas", listaRutinas);
345
346     if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Deficit")) {
347         listaRutinas = rutinaRepository.findByTipo("Deficit");
348         model.addAttribute("listaRutinas", listaRutinas);
349
350         if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
351
352             listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Deficit");
353             model.addAttribute("listaRutinas", listaRutinas);
354         }
355         else if(pacienteActual.getFrecEjercicios().equals("30 minutos a diario")) {
356
357             List<String> niveles = Arrays.asList("Principiante", "Intermedio");
358             listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
359             model.addAttribute("listaRutinas", listaRutinas);
360         }
361     }

```

```

363     }
364     else if(pacienteActual!=null && pacienteActual.getObjetivo().equals("Volumen")) {
365         listaRutinas = rutinaRepository.findByTipo("Volumen");
366         model.addAttribute("listaRutinas", listaRutinas);
367     }
368     if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
369         listaRutinas = rutinaRepository.findByNivelAndTipo("Principiante", "Volumen");
370         model.addAttribute("listaRutinas", listaRutinas);
371     }
372     else if(pacienteActual.getFrecEjercicios().equals("30 minutos diario")) {
373         List<String> niveles = Arrays.asList("Principiante", "Intermedio");
374         listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Volumen");
375         model.addAttribute("listaRutinas", listaRutinas);
376     }
377 }
378 }
379 else if (pacienteActual!=null){
380     if(pacienteActual.getFrecEjercicios().equals("A veces o nada") || pacienteActual.getFrecEjercicios().equals("Semanalmente")) {
381         listaRutinas = rutinaRepository.findByNivel("Principiante");
382         model.addAttribute("listaRutinas", listaRutinas);
383     }
384     else if(pacienteActual.getFrecEjercicios().equals("30 minutos a diario")) {
385         List<String> niveles = Arrays.asList("Principiante", "Intermedio");
386         listaRutinas = rutinaRepository.findByNivelesAndTipo(niveles, "Deficit");
387         model.addAttribute("listaRutinas", listaRutinas);
388     }
389 }
390 }
391 }
392 }
393 }
394 return "usuario/horario";
395 }
396 }

<h2>Horario de rutinas</h2>
<p>Programa tus tiempos para hacer rutinas a tu gusto.</p>
<br>

<span class="span" th:if="${eliminado}" th:text="${eliminado}"></span>

<div class="w3-row-padding" style="margin: 40px;">

    <div th:each="horario: ${miHorario}">
        <div class="w3-card w3-container">
            <h2 th:text="${horario.rutina.nombre}"></h2>
            <a th:href="@{/rutina/verDetalleRutina(idrutina=${horario.rutina.idrutina})}">Ver detalles</a>
            <br>
            <br>

            <div class="w3-col s6">

                <p>Descanso por serie: <span th:text="${horario.descaSerie}"></span></p>
                <p>Descanso por ejercicio: <span th:text="${horario.descaEjercicio}"></span></p>

                <h3><span th:text="${horario.dia}"></span></h3>
                <br>
                <a th:href="@{/horario/eliminarHora/{idhorario} (idhorario=${horario.idhorario})}"
                   class="btn btn-danger btn-circle btn-sm">Eliminar de tu horario</a>
            </div>
            <div class="w3-col s6">
                <h3>Periodo:</h3>
                <span th:text="${horario.periodo}"></span>
            </div>
        </div>
    </div>
</div>

```

- **Retornar al menú**

```

    @GetMapping("/retroceder")
    public String regresarMenu(HttpServletRequest sesion, Model model) {
        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
        if (objUsuario!=null) {
            TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
            boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
            model.addAttribute("esPaciente", esPaciente);
        }
        return "menu";
    }
}

```

## VISTA DEL MENÚ PARA LOS GET

```

<ul>
    <li><a th:href="@{/home/verPerfil}">
        Perfil<i class="fa-solid fa-user"></i></a></li>
    <li><a th:href="@{/home/ajustescuenta}">
        Ajustes<i class="fa-solid fa-cog"></i></a></li>
    <li><a th:href="@{/home/verNotis}">
        Notificaciones<i class="fa-solid fa-bell"></i></a></li>
    <li><a th:if="${esPaciente}" th:href="@{/home/verHorario}">
        Mi horario<i class="fa-regular fa-calendar-days"></i></a></li>
    </ul>

</header>
<main>
    <section id="intro">
        <h1>Bienvenido</h1>
        <b th:if="${esPaciente}">En esta aplicación podrás mejorar tu estilo de vida, con rutinas y dietas que puedes descubrir tu mismo o por reco
    </section>

    <section style="background-color: #d7ebee;" th:if="${esPaciente}">
        <h2>Nutriologos</h2>
        

        <div>
            <br><br>
            <p>Visita a nuestros nutriologos registrados y conócelos en medio de reuniones o citas según tus necesidades:</p>
            <span><a type="button" th:href="@{/home/verNutriologos}">Entrar</a></span>
            </p>
            <br>
        </div>
    </section>

    <section>
        <h2 th:if="${esPaciente}">Nuestras Recomendaciones</h2>
        <p th:if="${esPaciente}">Estas son listas creadas por usuarios nutriologos/instructores y están ajustada según tu información como paciente</p>
        <div class="container-01">
            <div>
                <a th:href="@{/home/verRutinas}" class="enlace-imagen">
                    
                </a>
                <h3>Rutinas</h3>
                <p>Hecha un vistazo a nuestras rutinas para organizarlas y ver sus detalles en cualquier momento</p>
            </div>
            <div>
                <a th:href="@{/home/verDietas}" class="enlace-imagen">
                    
                </a>
                <h3>Dietas</h3>
                <p>Hecha un vistazo a nuestras dietas para organizarlas y ver sus detalles en cualquier momento</p>
            </div>
        </div>
    </section>
</main>


<script type="text/javascript">
function myFunction() {
    var x = document.getElementById("myTopnav");
    if (x.className === "topnav") {
        x.className += " responsive";
    } else {
        x.className = "topnav";
    }
}
</script>

```

- **RutinaController**

```
1 package com.DietasYRutinasOnline.controller;
2
3④ import java.time.LocalDateTime;[]
4
5
6 @Controller
7 @RequestMapping("/rutina")
8 public class RutinaController {
9
10    @Autowired
11    EjercicioRepository ejercicioRepository;
12
13    @Autowired
14    RutinaRepository rutinaRepository;
15
16    @Autowired
17    InfoPacienteRepository infoPacienteRepository;
18
19    @Autowired
20    HorarioRepository horarioRepository;
21
22    @Autowired
23    TransaccionRepository transaccionRepository;
24
25    @Autowired
26    NotificacionRepository notificacionRepository;
27
28    @Autowired
29    TipoUsuarioRepository tipoUsuarioRepository;
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
```

- Ver y buscar ejercicios

```
62 // ---- VER EJERCICIOS -----
63④ @GetMapping("/verEjercicios")
64 public String verEjercicios(Model model) {
65     List<Ejercicio> listaEjercicios = ejercicioRepository.findAll();
66     model.addAttribute("listaEjercicios", listaEjercicios);
67
68     List<String> cbxEjercicios = ejercicioRepository.findDistinctGrupomuscular();
69     if (cbxEjercicios.size() > 5) {
70         cbxEjercicios = cbxEjercicios.subList(0, 5);
71     }
72     model.addAttribute("cbxEjercicios", cbxEjercicios);
73     return "rutinas/lista_ejercicios";
74 }
75
76④ @GetMapping("/buscarEjercicios")
77 public String buscarEjercicios(String grupomuscular, Model model) {
78
79     List<String> cbxEjercicios = ejercicioRepository.findDistinctGrupomuscular();
80     if (cbxEjercicios.size() > 5) {
81         cbxEjercicios = cbxEjercicios.subList(0, 5);
82     }
83
84     List<Ejercicio> listaEjercicios = ejercicioRepository.findByGrupomuscular(grupomuscular);
85     model.addAttribute("listaEjercicios", listaEjercicios);
86
87     model.addAttribute("cbxEjercicios", cbxEjercicios);
88     return "rutinas/lista_ejercicios";
89 }
90
```

```

<h2>Ejercicios</h2>

<div>
  <table class="pure-table">
    <div><ul class="pure-menu pure-menu-horizontal">
      <li class="pure-menu-item">
        <a th:href="@{/home/verRutinas}" class="pure-menu-link">Rutinas</a>
      </li>
      <li class="pure-menu-item pure-menu-selected">
        <a href="#" class="pure-menu-link">Ejercicios</a>
      </li>
    </ul>
  </div>
    <thead>
      <tr>
        <th>Nombre del Ejercicio</th>
        <th>Grupo Muscular</th>
        <th>Tipo</th>
        <th>Número de Series</th>
        <th>Número de Repeticiones</th>
        <th>Descripción</th>
      </tr>
    </thead>
    <tbody class="pure-table-border">
      <tr th:each="ejercicio: ${ListaEjercicios}">
        <td><span th:text="${ejercicio.nombre}"></span></td>
        <td><span th:text="${ejercicio.grupomuscular}"></span></td>
        <td><span th:text="${ejercicio.tipo}"></span></td>
        <td><span th:text="${ejercicio.series}"></span></td>
        <td><span th:text="${ejercicio.repeticiones}"></span></td>
        <td><span th:text="${ejercicio.descripcion}"></span></td>
      </tr>
    </tbody>
  </table>
</div>

```

- Crear y grabar rutina (NUTRÍÓLOGO)

```
// ---- VENTANA CREAR RUTINA -----
@PostMapping("/crearRutina")
public String crearRutina(Model model) {
    List<Ejercicio> listaEjercicios = ejercicioRepository.findAll();
    model.addAttribute("listaejercicios", listaEjercicios);
    Rutina objRutina = new Rutina();
    model.addAttribute("objRutina", objRutina);
    return "rutas/nueva_rutina";
}

@PostMapping("/grabarRutina")
public String grabarRutina(
    HttpSession sesion,
    @ModelAttribute("objRutina") Rutina objRutina,
    Model model) {

    Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
    if (objUsuario!=null) {
        objRutina.setNutriologo(objUsuario);
        objRutina.setEstado("Activo");
        rutinaRepository.save(objRutina);

        Transaccion objTransaccion = new Transaccion();
        objTransaccion.setFecha(LocalDateTime.now());
        objTransaccion.setTipo("Creación");
        objTransaccion.setUsuario(objUsuario);
        objTransaccion.setRutina(objRutina);
        transaccionRepository.save(objTransaccion);

        TipoUsuario vistaUsuario = objUsuario.getTipousuario();
        boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
        model.addAttribute("esPaciente", esPaciente);
    }
    List<Rutina> listaRutinas = rutinaRepository.findByEstado("Activo");
    model.addAttribute("finalizado", "Bien hecho, ahora los pacientes podrán ver tu rutina");
    model.addAttribute("listarutinas", listaRutinas);
    return "rutas/menu_rutinas";
}

<section>
    <h2>Insertar nueva rutina</h2>
    <form class="pure-form pure-form-stacked"
        th:action="@{/rutina/grabarRutina}"
        th:object="${objRutina}"
        method="POST" >

        <label for="nombres_apellidos">Nombre de rutina</label>
        <input
            type="text"
            id="multi-first-name"
            th:field="*{nombre}"
            required="">

        <label for="percentage">Tipo de rutina</label>
        <select id="percentage" name="tipo">
            <option th:value="Deficit" selected="selected">Rutina de Deficit</option>
            <option th:value="Volumen">Rutina de Volumen</option>
        </select>

        <label for="percentage">Parte del cuerpo</label>
        <select id="percentage" name="partecuerpo">
            <option th:value="Tren Superior" selected="selected">Tren Superior</option>
            <option th:value="Tren Inferior">Tren Inferior</option>
        </select>

        <label for="percentage">Nivel</label>
        <select id="percentage" name="nivel">
            <option th:value="Principiante" selected="selected">Principiante</option>
            <option th:value="Intermedio">Intermedio</option>
            <option th:value="Avanzado">Avanzado</option>
        </select>

        <label for="mensaje">Descripción</label>
        <textarea id="mensaje" th:field="*{descripcion}"></textarea>
    
```

```

<label>Ejercicios</label>

<div class="deslisable">
<table class="pure-table">

    <thead>
        <tr>
            <th>Nombre del Ejercicio</th>
            <th>Grupo Muscular</th>
            <th>Tipo</th>
            <th>Número de Series</th>
            <th>Número de Repeticiones</th>
            <th>Descripción</th>
            <th>Seleccionar</th>
        </tr>
    </thead>

    <tbody class="pure-table-border">
        <tr th:each="ejercicio: ${listaEjercicios}">
            <td><span th:text="${ejercicio.nombre}"></span></td>
            <td><span th:text="${ejercicio.grupomuscular}"></span></td>
            <td><span th:text="${ejercicio.tipo}"></span></td>
            <td><span th:text="${ejercicio.series}"></span></td>
            <td><span th:text="${ejercicio.repeticiones}"></span></td>
            <td><span th:text="${ejercicio.descripcion}"></span></td>
            <td><input
                    type="checkbox"
                    th:field="*{ejercicio}"
                    th:value="${ejercicio.idejercicio}"></input></td>
        </tr>
    </tbody>
</table>

</div>
<span class="span" th:if="${error}" th:text="${error}"></span>

<br>
<input type="submit" class="pure-button pure-button-primary" value="Guardar Rutina">

</form>
</section>
</main>

```

- Ver detalles de una rutina

```

130 // ---- VER DETALLE RUTINA -----
131 @GetMapping("/verDetalleRutina")
132 public String verDetalleRutina(
133     HttpSession sesion,
134     @RequestParam("idrutina") int idrutina,
135     @ModelAttribute("objRutina") Rutina objRutina,
136     Model model) {
137
138     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
139     if (objUsuario!=null) {
140         TipoUsuario vistausuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
141         boolean esPaciente = vistausuario.getNomtipousu().equals("Paciente");
142         model.addAttribute("esPaciente", esPaciente);
143     }
144
145     Rutina detalleRutina = rutinaRepository.findByIdrutina(idrutina);
146     model.addAttribute("detalleRutina", detalleRutina);
147
148     Usuario nutriActual = (Usuario) sesion.getAttribute("usuario");
149     model.addAttribute("esCreador", nutriActual.getIdusuario() == detalleRutina.getNutriologo().getIdusuario());
150     return "rutas/detalle_rutina";
151 }

```

```

<section>
    <div class="mb-3" th:value="${detalleRutina}">
        <h2 th:text="${detalleRutina.nombre}"></h2>
        <br>
        <label class="form-label">Tipo de Rutina:</label>
        <span th:text="${detalleRutina.tipo}"></span>
        <br>
        <label class="form-label">Descripción:</label>
        <span th:text="${detalleRutina.descripcion}"></span>
        <br>
        <label class="form-label">Nivel:</label>
        <span th:text="${detalleRutina.nivel}"></span>
        <br>
        <label class="form-label">Creado por:</label>
        <a th:href="@{/perfil/verSuPerfil(idusuario=${detalleRutina.nutriologo.idusuario})}">
            <span th:text="${detalleRutina.nutriologo.nombres}"></span>
            <span th:text="${detalleRutina.nutriologo.apellidos}"></span>
        </a>
        <!--<span th:text="${detalleRutina.nutriologo.nombres}"></span>
        <span th:text="${detalleRutina.nutriologo.apellidos}"></span>-->
    </div>
    <div>
        <br>
        <h3>Ejercicios</h3>
    </div>
    <table class="pure-table">
        <thead>
            <tr>
                <th>Nombre del Ejercicio</th>
                <th>Grupo Muscular</th>
                <th>Tipo</th>
                <th>Número de Series</th>
                <th>Número de Repeticiones</th>
                <th>Descripción</th>
            </tr>
        </thead>
        <tbody class="pure-table-border">
            <tr th:each="ejercicio: ${detalleRutina.ejercicio}">
                <td><span th:text="${ejercicio.nombre}"></span></td>
                <td><span th:text="${ejercicio.grupomuscular}"></span></td>
                <td><span th:text="${ejercicio.tipo}"></span></td>
                <td><span th:text="${ejercicio.series}"></span></td>
                <td><span th:text="${ejercicio.repeticiones}"></span></td>
                <td><span th:text="${ejercicio.descripcion}"></span></td>
            </tr>
        </tbody>
    </table>

```

```

<form th:if="${esCreador}" th:action="@{/rutina/editarRutina}" th:method="POST">
    <input type="hidden" name="idrutina" th:value="${detalleRutina.idrutina}" />
    <button type="submit" class="pure-button pure-button-primary">
        Editar Rutina
    </button>
</form>

```

- **Editar y actualizar rutina (NUTRIÓLOGO)**

```

// ----- EDITAR RUTINA -----
@PostMapping("/editarRutina")
public String editarRutina(
    @RequestParam("idrutina")int idrutina,
    Model model){
    Rutina objRutina = rutinaRepository.findByIdrutina(idrutina);
    model.addAttribute("objRutina", objRutina);

    List<Ejercicio> listaEjercicios = ejercicioRepository.findAll();
    model.addAttribute("listaEjercicios", listaEjercicios);
    return "rutas/editar_rutina";
}

@PostMapping("/actualizarRutina")
public String actualizarRutina(
    HttpSession sesion,
    @ModelAttribute("objRutina") Rutina objRutina,
    Model model) {

    Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
    if (objUsuario!=null) {
        Rutina rutinaActual = rutinaRepository.findByIdrutina(objRutina.getIdrutina());
        rutinaActual.setNombre(objRutina.getNombre());
        rutinaActual.setTipo(objRutina.getTipo());
        rutinaActual.setNivel(objRutina.getNivel());
        rutinaActual.setDescripcion(objRutina.getDescripcion());
        rutinaActual.setEjercicio(objRutina.getEjercicio());
        rutinaRepository.save(rutinaActual);

        TipoUsuario vistaUsuario = objUsuario.getTipousuario();
        boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
        model.addAttribute("esPaciente", esPaciente);
    }
    List<Rutina> listaRutinas = rutinaRepository.findByEstado("Activo");
    model.addAttribute("listaRutinas", listaRutinas);
    return "rutas/menu_rutinas";
}

```

- **DietaController**

```

1 package com.DietasYRutinasOnline.controller;
2
3+ import java.time.LocalDateTime;[]
36
37
38 @Controller
39 @RequestMapping("/dieta")
40 public class DietaController {
41
42@  @Autowired
43 AlimentoRepository alimentoRepository;
44
45@  @Autowired
46 CondicionRepository condicionRepository;
47
48@  @Autowired
49 DietaRepository dietaRepository;
50
51@  @Autowired
52 InfoPacienteRepository infoPacienteRepository;
53
54@  @Autowired
55 TransaccionRepository transaccionRepository;
56
57@  @Autowired
58 NotificacionRepository notificacionRepository;
59
60@  @Autowired
61 UsuarioRepository usuarioRepository;
62
63@  @Autowired
64 TipoUsuarioRepository tipoUsuarioRepository;
65

```

- Ver y buscar alimentos

```

    ...
67@  @GetMapping("/verAlimentos")
68  public String verAlimentos(Model model) {
69      List<Alimento> listaAlimentos = alimentoRepository.findAll();
70      model.addAttribute("listaAlimentos", listaAlimentos);
71
72      List<String> cbxAlimentos = alimentoRepository.findDistinctTipos();
73      if (cbxAlimentos.size() > 4) {
74          cbxAlimentos = cbxAlimentos.subList(0, 4);
75      }
76      model.addAttribute("cbxAlimentos", cbxAlimentos);
77      return "dietas/lista_alimentos";
78  }
79
80@  @GetMapping("/buscarAlimentos")
81  public String buscarEjercicios(String tipo, Model model) {
82      List<String> cbxAlimentos = alimentoRepository.findDistinctTipos();
83      if (cbxAlimentos.size() > 4) {
84          cbxAlimentos = cbxAlimentos.subList(0, 4);
85      }
86
87      List<Alimento> listaAlimentos = alimentoRepository.findByTipo(tipo);
88      model.addAttribute("listaAlimentos", listaAlimentos);
89
90      model.addAttribute("cbxAlimentos", cbxAlimentos);
91      return "dietas/lista_alimentos";
92  }
93
94  <table class="pure-table">
95
96      <div class="pure-menu pure-menu-horizontal">
97          <ul class="pure-menu-list">
98              <li class="pure-menu-item">
99                  <a th:href="@{/home/verDietas}" class="pure-menu-link">Dieta</a>
100             </li>
101             <li class="pure-menu-item pure-menu-selected">
102                 <a href="#" class="pure-menu-link">Alimentos</a>
103             </li>
104         </ul>
105
106         <form th:action="@{/dieta/buscarAlimentos}" method="GET">
107             <select class="form-select" name="tipo">
108                 <option th:each="alimento : ${cbxAlimentos}"
109                         th:value="${alimento}"
110                         th:text="${alimento}">
111             </option>
112         </select>
113         <button type="submit" class="pure-button pure-button-secondary">Buscar</button>
114     </form>
115
116  </div>
117
118  <thead>
119      <tr>
120          <th>Alimento</th>
121          <th>Nutrientes</th>
122          <th>Tipo</th>
123          <th>Descripción</th>
124      </tr>
125  </thead>
126
127  <tbody class="pure-table-border">
128      <tr th:each="alimento: ${listaAlimentos}">
129          <td><span th:text="${alimento.nombre}"></span></td>
130          <td><span th:text="${alimento.nutrientes}"></span></td>
131          <td><span th:text="${alimento.tipo}"></span></td>
132          <td><span th:text="${alimento.descripcion}"></span></td>
133      </tr>
134  </tbody>
135</table>
136

```

- Seguir dieta (PACIENTE)

```
--  
94@  
95 @PostMapping("/seguirDieta")  
96 public String seguirDieta(  
97     HttpSession sesion,  
98     @ModelAttribute("objDieta") Dieta objDieta,  
99     Model model) {  
100  
101     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");  
102     if (objUsuario!=null) {  
103         TipoUsuario vistaUsuario = objUsuario.getTipousuario();  
104         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");  
105         model.addAttribute("esPaciente", esPaciente);  
106  
107         InfoPaciente pacienteActual = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");  
108         model.addAttribute("pacienteActual", pacienteActual);  
109  
110         if (pacienteActual!=null) {  
111             Dieta dietaPaciente = dietaRepository.findById(objDieta.getIddieta()).orElse(null);  
112             if (dietaPaciente!=null && !pacienteActual.getDieta().contains(dietaPaciente)) {  
113                 pacienteActual.getDieta().add(dietaPaciente);  
114                 infoPacienteRepository.save(pacienteActual);  
115                 model.addAttribute("exito", "Se guardó a tu perfil con éxito");  
116             }  
117             else {  
118                 model.addAttribute("excepcion", "Ya estás haciendo esta dieta");  
119             }  
120  
121             Transaccion objTransaccion = new Transaccion();  
122             objTransaccion.setFecha(LocalDateTime.now());  
123             objTransaccion.setTipo("Seguir Dieta");  
124             objTransaccion.setInfopaciente(pacienteActual);  
125             objTransaccion.setUsuario(objUsuario);  
126             objTransaccion.setDieta(dietaPaciente);  
127             transaccionRepository.save(objTransaccion);  
128  
129  
130             /* RETORNAR */  
131             List<Dieta> listaDietas;  
132             listaDietas = dietaRepository.findByEstado("Activo");  
133             model.addAttribute("listaDietas", listaDietas);  
134  
135             if (pacienteActual != null && "Deficit".equals(pacienteActual.getObjetivo())) {  
136                 listaDietas = dietaRepository.findByObjetivo("Deficit");  
137                 model.addAttribute("listaDietas", listaDietas);  
138             }  
139             else if (pacienteActual != null && "Volumen".equals(pacienteActual.getObjetivo())) {  
140                 listaDietas = dietaRepository.findByObjetivo("Volumen");  
141                 model.addAttribute("listaDietas", listaDietas);  
142             }  
143             return "dietas/menu_dietas";  
144         }  
145     }  
146     return "index";  
147 }  
148 }
```

- Dejar de seguir dieta (PACIENTE)

```
150 @PostMapping("/dejarSeguirDieta")
151 public String dejarSeguirDieta(
152     HttpSession sesion,
153     @ModelAttribute("objDieta") Dieta objDieta,
154     Model model) {
155
156     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
157     if (objUsuario!=null) {
158         TipoUsuario vistaUsuario = objUsuario.getTipousuario();
159         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
160         model.addAttribute("esPaciente", esPaciente);
161
162         InfoPaciente miInfo = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
163         model.addAttribute("miInfo", miInfo);
164
165         if(miInfo!=null) {
166             Dieta dietaPaciente = dietaRepository.findById(objDieta.getIddieta()).orElse(null);
167
168             miInfo.getDieta().remove(dietaPaciente);
169             infoPacienteRepository.save(miInfo);
170             model.addAttribute("mensaje", "Dieta eliminada de tu perfil.");
171         }
172
173         Transaccion objTransaccion = new Transaccion();
174         objTransaccion.setFecha(LocalDateTime.now());
175         objTransaccion.setTipo("Dejar seguir Dieta");
176         objTransaccion.setInfopaciente(miInfo);
177         objTransaccion.setUsuario(objUsuario);
178         objTransaccion.setDieta(objDieta);
179         transaccionRepository.save(objTransaccion);
180
181         model.addAttribute("objUsuario", objUsuario);
182
183         InfoPaciente miInfo1 = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
184         model.addAttribute("miInfo", miInfo1);
185         return "perfil";
186     }
187     return "index";
188 }
```

#### - Crear y grabar dieta (NUTRIOLOGO)

```
192 @PostMapping("/crearDieta")
193 public String crearDieta(Model model) {
194     List<Alimento> listaAlimentos = alimentoRepository.findAll();
195     model.addAttribute("listaAlimentos", listaAlimentos);
196     List<Condicion> listaCondiciones = condicionRepository.findAll();
197     model.addAttribute("listaCondiciones", listaCondiciones);
198
199     List<String> cbxAlimentos = alimentoRepository.findDistinctTipos();
200     if (cbxAlimentos.size() > 4) {
201         cbxAlimentos = cbxAlimentos.subList(0, 4);
202     }
203     model.addAttribute("cbxAlimentos", cbxAlimentos);
204
205     Dieta objDieta = new Dieta();
206     model.addAttribute("objDieta", objDieta);
207     return "dietas/nueva_dieta";
208 }
```

```
210@PostMapping("/grabarDieta")
211public String grabarDieta(
212    HttpSession sesion,
213    @ModelAttribute("objDieta") Dieta objDieta,
214    Model model) {
215
216    try{
217        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
218        if (objUsuario!=null) {
219            objDieta.setNutriologo(objUsuario);
220            objDieta.setEstado("Activo");
221            dietaRepository.save(objDieta);
222
223            Transaccion objTransaccion = new Transaccion();
224            objTransaccion.setFecha(LocalDateTime.now());
225            objTransaccion.setTipo("Creación");
226            objTransaccion.setUsuario(objUsuario);
227            objTransaccion.setDieta(objDieta);
228            transaccionRepository.save(objTransaccion);
229
230            TipoUsuario vistaUsuario = objUsuario.getTipousuario();
231            boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
232            model.addAttribute("esPaciente", esPaciente);
233        }
234        List<Dieta> listaDietas = dietaRepository.findByEstado("Activo");
235        model.addAttribute("listaDietas", listaDietas);
236        return "dietas/menu_dietas";
237    }
238    catch(Exception ex) {
239        return "dietas/nueva_dieta";
240    }
241
242}
```

```
<section>
<h2>Insertar nueva dieta</h2>
<form class="pure-form pure-form-stacked"
      th:action="@{/dieta/grabarDieta}"
      th:object="${objDieta}"
      method="POST" >

    <label for="nombres_apellidos">Nombre</label>
    <input
        type="text"
        id="multi-first-name"
        th:field="#{nombre}"
        required=""
    />

    <label for="percentage">Objetivo</label>
    <select id="percentage" name="objetivo">
        <option th:value="'Deficit'" selected="selected">Deficit</option>
        <option th:value="'Volumen'">Volumen</option>
    </select>

    <label for="mensaje">Descripción</label>
    <textarea id="mensaje" th:field="*{descripcion}"></textarea>

    <label>¿Qué alimento incluye?</label>
    <div class="deslisable">
        <table class="pure-table tabla-check">
            <thead>
                <tr>
                    <th>Alimento</th>
                    <th>Nutrientes</th>
                    <th>Tipo</th>
                    <th>Descripción</th>
                    <th>Seleccionar</th>
                </tr>
            </thead>
```

```

<tbody class="pure-table-border">
    <tr th:each="alimento: ${ListaAlimentos}">
        <td><span th:text="${alimento.nombre}"></span></td>
        <td><span th:text="${alimento.nutrientes}"></span></td>
        <td><span th:text="${alimento.tipo}"></span></td>
        <td><span th:text="${alimento.descripcion}"></span></td>
        <td><input
            type="checkbox"
            th:field="*{alimento}"
            th:value="${alimento.idalimento}"></input></td>
    </tr>
</tbody>
</table>
</div>

<label>Condición alimenticia</label>
<div th:each="condicion : ${ListaCondiciones}">
    <label th:text="${condicion.nombre}"></label>
    <input
        type="checkbox"
        th:field="*{condicion}"
        th:value="${condicion.idcondicion}"
    />
</div>
<span class="span" th:if="${error}" th:text="${error}"></span>
<input type="submit" class="pure-button pure-button-primary" value="Guardar Dieta">
</form>
</section>
</main>

```

- Ver detalle de una dieta

```

245④ @GetMapping("/verDetalleDieta")
246 public String verDetalleDieta(
247     HttpSession sesion,
248     @RequestParam("iddieta") int iddieta,
249     @ModelAttribute("objDieta") Dieta objDieta,
250     Model model) {
251
252     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
253     if (objUsuario!=null) {
254         TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
255         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
256         model.addAttribute("esPaciente", esPaciente);
257     }
258
259     Dieta detalleDieta = dietaRepository.findByIddieta(iddieta);
260     model.addAttribute("detalleDieta", detalleDieta);
261
262     Usuario nutriActual = (Usuario) sesion.getAttribute("usuario");
263     model.addAttribute("esCreador", nutriActual.getIdusuario() == detalleDieta.getNutriologo().getIdusuario());
264
265     InfoPaciente pacienteActual = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
266     model.addAttribute("pacienteActual", pacienteActual);
267
268     return "dietas/detalle_dieta";
269 }

```

```
<div class="mb-3" th:value="${detalleDieta}">
    <h2 th:text="${detalleDieta.nombre}"></h2>
    <br>

    <label class="form-label">Objetivo:</label>
    <span th:text="${detalleDieta.objetivo}"></span>
    <br>

    <label class="form-label">Descripción:</label>
    <span th:text="${detalleDieta.descripcion}"></span>
    <br>

    <label class="form-label">Creado por:</label>
    <a th:href="@{/perfil/versuPerfil(idusuario=${detalleDieta.nutriologo.idusuario})}">
        <i class="bx bx-edit-alt me-2"></i>
        <span th:text="${detalleDieta.nutriologo.nombres}"></span>
        <span th:text="${detalleDieta.nutriologo.apellidos}"></span>
    </a>
</div>

<div><h3>Condición</h3>
    <div th:each="condicion : ${detalleDieta.condicion}">
        <p th:text="${condicion.nombre}"></p>
    </div>
</div>

<div><h3>Alimentos incluidos</h3></div>
<table class="pure-table">
    <thead>
        <tr>
            <th>Alimento</th>
            <th>Nutrientes</th>
            <th>Tipo</th>
            <th>Descripción</th>
            <th>Descripción</th>
        </tr>
    </thead>
    <tbody class="pure-table-border">
        <tr th:each="alimento: ${detalleDieta.alimento}">
            <td><span th:text="${alimento.nombre}"></span></td>
            <td><span th:text="${alimento.nutrientes}"></span></td>
            <td><span th:text="${alimento.tipo}"></span></td>
            <td><span th:text="${alimento.descripcion}"></span></td>
            <td><span th:text="${alimento.descripcion}"></span></td>
        </tr>
    </tbody>
</table>

<form th:if="${esCreador}" th:action="@{/dieta/editarDieta}" th:method="POST">
    <input type="hidden" name="iddieta" th:value="${detalleDieta.iddieta}" />
    <button type="submit" class="pure-button pure-button-primary">
        Editar Dieta
    </button>
</form>
```

- **Editar y actualizar dieta**

```
324 @PostMapping("/editarDieta")
325 public String editarDieta(
326     @RequestParam("iddieta")int iddieta,
327     Model model){
328     Dieta objDieta = dietaRepository.findByIdieta(iddieta);
329     model.addAttribute("objDieta", objDieta);
330
331     List<Alimento> listaAlimentos = alimentoRepository.findAll();
332     model.addAttribute("listaAlimentos", listaAlimentos);
333     List<Condicion> listaCondiciones = condicionRepository.findByEstado("Activo");
334     model.addAttribute("listaCondiciones", listaCondiciones);
335
336     return "dietas/editar_dieta";
337 }
338
339 @PostMapping("/actualizarDieta")
340 public String actualizarDieta(
341     HttpSession sesion,
342     @ModelAttribute("objDieta") Dieta objDieta,
343     Model model) {
344
345     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
346     if (objUsuario!=null) {
347         Dieta dietaActual = dietaRepository.findByIdieta(objDieta.getIdieta());
348         dietaActual.setNombre(objDieta.getNombre());
349         dietaActual.setObjetivo(objDieta.getObjetivo());
350         dietaActual.setDescripcion(objDieta.getDescripcion());
351         dietaActual.setAlimento(objDieta.getAlimento());
352         dietaActual.setCondicion(objDieta.getCondicion());
353         dietaRepository.save(dietaActual);
354
355         TipoUsuario vistaUsuario = objUsuario.getTipousuario();
356         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
357         model.addAttribute("esPaciente", esPaciente);
358     }
359     List<Dieta> listaDietas = dietaRepository.findByEstado("Activo");
360     model.addAttribute("listaDietas", listaDietas);
361     return "dietas/menu_dietas";
362 }
363
```

- **HorarioController (PACIENTE)**

```
44 import com.DietasYRutinasOnline.entity.Usuario;
45 import com.DietasYRutinasOnline.repository.DietaRepository;
46 import com.DietasYRutinasOnline.repository.HorarioRepository;
47 import com.DietasYRutinasOnline.repository.ReunionRepository;
48 import com.DietasYRutinasOnline.repository.RutinaRepository;
49 import com.DietasYRutinasOnline.repository.TipoUsuarioRepository;
50 import com.DietasYRutinasOnline.repository.TransaccionRepository;
51 import com.DietasYRutinasOnline.repository.UsuarioRepository;
52
53 @Controller
54 @RequestMapping("/horario")
55 public class HorarioController {
56
57     @Autowired
58     TipoUsuarioRepository tipoUsuarioRepository;
59
60     @Autowired
61     RutinaRepository rutinaRepository;
62
63     @Autowired
64     DietaRepository dietaRepository;
65
66     @Autowired
67     ReunionRepository reunionRepository;
68
69     @Autowired
70     HorarioRepository horarioRepository;
71
72     @Autowired
73     UsuarioRepository usuarioRepository;
74
75     @Autowired
76     TransaccionRepository transaccionRepository;
```

## - Programar nuevo horario de ejercicios

```
58@PostMapping("/grabarHorario")
59 public String grabarHorario(
60     HttpSession sesion,
61     @ModelAttribute("objHorario") Horario objHorario,
62     @RequestParam("dia") String dia,
63     @RequestParam("periodo") string periodo,
64     Model model) {
65
66     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
67     if (objUsuario!=null) {
68         objHorario.setPaciente(objUsuario);
69         objHorario.setEstado("Activo");
70
71         Horario conflictoHorario = horarioRepository.findByPacienteAndDiaAndPeriodoAndEstado(objUsuario, dia, periodo, "Activo");
72         Horario conflictoDia = horarioRepository.findByPacienteAndDiaAndEstado(objUsuario, dia, "Activo");
73
74         if(conflictoHorario!=null) {
75             model.addAttribute("error", "El horario está en conflicto con otro existente");
76         }
77         else if(conflictoDia!=null){
78             model.addAttribute("error", "Te sugerimos que no elijas más de una rutina para un mismo día");
79         }
80         else {
81             //objHorario.setEstado("Activo");
82             horarioRepository.save(objHorario);
83             model.addAttribute("exito", "se añadió con éxito");
84
85             Transaccion objTransaccion = new Transaccion();
86             objTransaccion.setFecha(LocalDateTime.now());
87             objTransaccion.setTipo("CREACIÓN");
88             objTransaccion.setUsuario(objUsuario);
89             objTransaccion.setHorario(objHorario);
90             transaccionRepository.save(objTransaccion);
91         }
92     }
93     List<Rutina> listaRutinas = rutinaRepository.findAll();
94     model.addAttribute("listaRutinas", listaRutinas);
95
96     List<Horario> miHorario = horarioRepository.findByPacienteAndEstado(objUsuario, "Activo");
97     model.addAttribute("miHorario", miHorario);
98
99     model.addAttribute("objHorario", new Horario());
100    return "usuario/horario";
101 }
```

```
<form class="pure-form pure-form-stacked"
      style="margin: 20px;">
    <th:action="@{/horario/grabarHorario}"
      th:object="${objHorario}"
      method="POST">

      <h3>Inserte un nuevo horario</h3>
      <span class="Alert" thi:if="${error}" th:text="${error}"></span>
      <span class="span" th:if="${exito}" th:text="${exito}"></span>
      <br>

      <label>Rutina:</label>
      <select th:field="*{rutina.idrutina}">
        <option th:each="rutinas: ${listaRutinas}"
               th:value="${rutinas.idrutina}"
               th:text="${rutinas.nombre}">
        </option>
      </select>

      <label for="dia">Día:</label>
      <select name="dia">
        <option th:value="'LUNES'">Lunes</option>
        <option th:value="'MARTES'">Martes</option>
        <option th:value="'MIERCOLES'">Miércoles</option>
        <option th:value="'JUEVES'">Jueves</option>
        <option th:value="'VIERNES'">Viernes</option>
        <option th:value="'SABADO'">Sábado</option>
        <option th:value="'DOMINGO'">Domingo</option>
      </select>

      <label for="periodo">Periodo de día:</label>
      <select name="periodo">
        <option th:value="'Mañana'">Mañana</option>
        <option th:value="'Medio dia'">Medio día</option>
        <option th:value="'Tarde'">Tarde</option>
        <option th:value="'Noche'">Noche</option>
      </select>
```

```

<label for="descansoEjercicio">Descanso por ejercicio:</label>
<select name="descansoEjercicio">
    <option th:value="'no especificado'">No especificado</option>
    <option th:value="'5 minutos'">5 minutos</option>
    <option th:value="'10 minutos'">10 minutos</option>
    <option th:value="'15 minutos o más'">15 minutos o más</option>
</select>

<label for="descansoSerie">Descanso por serie:</label>
<select name="descansoSerie">
    <option th:value="'no especificado'">No especificado</option>
    <option th:value="'15 segundos'">15 segundos</option>
    <option th:value="'30 segundos'">30 segundos o más</option>
    <option th:value="'1 minuto'">1 minuto o más</option>
</select>

<!-- &lt;label for="horaInicio"&gt;Descanso por serie:&lt;/label&gt;
&lt;input type="time" min="00:30" max="01:30" th:field="*{horaInicio}" /&gt;

&lt;label for="horaFin"&gt;Descanso por ejercicio:&lt;/label&gt;
&lt;input type="time" min="01:00" max="07:00" th:field="*{horaFin}" /&gt;--&gt;

&lt;br&gt;
&lt;button type="submit" class="pure-button pure-button-secondary"&gt;Guardar nuevo horario&lt;/button&gt;
&lt;/form&gt;
</pre>

```

- **Eliminar horario**

```

103 @GetMapping("/eliminarHora/{idhorario}")
104 public String eliminarHora(
105     HttpSession sesion,
106     @ModelAttribute ("idhorario") int idhorario,
107     Model model) {
108
109     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
110     if (objUsuario!=null) {
111         Horario horaEliminada = horarioRepository.findByIdhorario(idhorario);
112         horaEliminada.setEstado("Inactivo");
113         horarioRepository.save(horaEliminada);
114         model.addAttribute("eliminado", "El horario ha eliminado con éxito");
115
116         Tipousuario vistaUsuario = objUsuario.getTipousuario();
117         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
118         model.addAttribute("esPaciente", esPaciente);
119     }
120     List<Rutina> listaRutinas = rutinaRepository.findAll();
121     model.addAttribute("listaRutinas", listaRutinas);
122
123     List<Horario> miHorario = horarioRepository.findByPacienteAndEstado(objUsuario, "Activo");
124     model.addAttribute("miHorario", miHorario);
125
126     Horario nuevoHorario = new Horario();
127     model.addAttribute("objHorario", nuevoHorario);
128     return "usuario/horario";
129 }
130
131 }
132

```

- **PerfilController**

```
1 package com.DietasYRutinasOnline.controller;
2
3④ import java.time.LocalDateTime;⑤
41
42 @Controller
43 @RequestMapping("/perfil")
44 public class PerfilController {
45
46④     @Autowired
47     HorarioRepository horarioRepository;
48
49④     @Autowired
50     RutinaRepository rutinaRepository;
51
52④     @Autowired
53     DietaRepository dietaRepository;
54
55④     @Autowired
56     UsuarioRepository usuarioRepository;
57
58④     @Autowired
59     TipoUsuarioRepository tipoUsuarioRepository;
60
61④     @Autowired
62     InfoPacienteRepository infoPacienteRepository;
63
64④     @Autowired
65     TransaccionRepository transaccionRepository;
66 }
```

- **Editar perfil**

```
68④     @PostMapping("/editarPerfil")
69     public String editarPerfil(
70         HttpSession sesion,
71         Model model) {
72         Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
73         if (objUsuario!=null) {
74             model.addAttribute("objUsuario", objUsuario);
75             return "usuario/editar_perfil";
76         }
77         else {
78             return "index";
79         }
80     }
```

- **Actualizar información y medidas de paciente después de 7 días (PACIENTE)**

```
82④     public boolean puedeEditar(InfoPaciente infoPaciente) {
83         LocalDateTime fechaHoy = LocalDateTime.now();
84         LocalDateTime fechaModificacion = infoPaciente.getFecha();
85         return ChronoUnit.DAYS.between(fechaModificacion, fechaHoy) >= 7;
86     }
87 }
```

```

88@GetMapping("/editarInfo")
89    public String editarInfo(HttpServletRequest sesion, Model model) {
90        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
91        if (objUsuario != null) {
92            Tipousuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
93            boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
94            model.addAttribute("esPaciente", esPaciente);
95
96            InfoPaciente infoActiva = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
97
98            if (infoActiva!=null && puedeEditar(infoActiva)) {
99                infoActiva.setEstado("Inactivo");
100               infoPacienteRepository.save(infoActiva);
101
102               InfoPaciente nuevaInfo = new InfoPaciente();
103               nuevaInfo.setFrecEjercicios(infoActiva.getFrecEjercicios());
104               nuevaInfo.setCondicion(infoActiva.getCondicion());
105               nuevaInfo.setObjetivo(infoActiva.getObjetivo());
106               nuevaInfo.setPesoCorporal(infoActiva.getPesoCorporal());
107               nuevaInfo.setEstatura(infoActiva.getEstatura());
108               nuevaInfo.setPerimCintura(infoActiva.getPerimCintura());
109               nuevaInfo.setPerimCadera(infoActiva.getPerimCadera());
110               nuevaInfo.setPerimMuslo(infoActiva.getPerimMuslo());
111               nuevaInfo.setPerimBrazo(infoActiva.getPerimBrazo());
112
113               model.addAttribute("infoActiva", infoActiva);
114               model.addAttribute("nuevaInfo", nuevaInfo);
115               return "usuario/editar_infopaciente";
116           }
117           else {
118               model.addAttribute("objUsuario", objUsuario);
119
120               InfoPaciente miInfo = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
121               model.addAttribute("miInfo", miInfo);
122
123               model.addAttribute("inactivo", "Debe esperar al menos 7 días antes de poder realizar otra modificación.");
124               return "perfil";
125           }
126       }
127       return "redirect:/index";
128   }
129
130 @PostMapping("/actualizarInfo")
131     public String actualizarInfo(
132         HttpServletRequest sesion,
133         @ModelAttribute("nuevaInfo") InfoPaciente nuevaInfo,
134         Model model) {
135
136         Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
137         if (objUsuario != null) {
138             Usuario usuRegistrado = usuarioRepository.findById(objUsuario.getIdusuario()).orElse(null);
139             nuevaInfo.setPaciente(usuRegistrado);
140             nuevaInfo.setEstado("Activo");
141             nuevaInfo.setFecha(LocalDateTime.now());
142
143             infoPacienteRepository.save(nuevaInfo);
144
145             Tipousuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
146             boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
147             model.addAttribute("esPaciente", esPaciente);
148         }
149
150         model.addAttribute("exito", "Su información se actualizó con éxito");
151         model.addAttribute("objUsuario", objUsuario);
152
153         InfoPaciente miInfo = infoPacienteRepository.findByPacienteAndEstado(objUsuario, "Activo");
154         model.addAttribute("miInfo", miInfo);
155
156         return "menu";
157     }

```

- **Ver historial de seguimiento**

```

159@GetMapping("/verSeguimiento")
160    public String verSeguimiento(HttpServletRequest sesion, Model model) {
161        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
162
163        List<InfoPaciente> listaSeguimiento = infoPacienteRepository.findByPaciente(objUsuario);
164        model.addAttribute("listaSeguimiento", listaSeguimiento);
165        Collections.reverse(listaSeguimiento);
166
167        return "usuario/seguimiento";
168    }

```

```

<div class="deslisable">
    <table class="pure-table">
        <thead>
            <tr>
                <th>Fecha de modificación</th>
                <th>Frecuencia de ejercicios</th>
                <th>Peso corporal</th>
                <th>Estatura</th>
                <th>Perímetro de cintura</th>
                <th>Perímetro de cadera</th>
                <th>Perímetro de muslo</th>
                <th>Perímetro de brazo</th>
            </tr>
        </thead>
        <tbody class="pure-table-border">
            <tr th:each="infopaciente: ${listaseguimiento}">
                <td><span th:text="${infopaciente.fecha}"></span></td>
                <td><span th:text="${infopaciente.frecEjercicios}"></span></td>
                <td><span th:text="${infopaciente.pesoCorporal}"></span></td>
                <td><span th:text="${infopaciente.estatura}"></span></td>
                <td><span th:text="${infopaciente.perimCintura}"></span></td>
                <td><span th:text="${infopaciente.perimCadera}"></span></td>
                <td><span th:text="${infopaciente.perimMuslo}"></span></td>
                <td><span th:text="${infopaciente.perimBrazo}"></span></td>
            </tr>
        </tbody>
    </table>
</div>

```

- **ReunionController**

```

1 package com.DietasYRutinasOnline.controller;
2
3+ import java.time.LocalDateTime;[]
41
42 @Controller
43 @RequestMapping("/reunion")
44 public class ReunionController {
45
46@    @Autowired
47    ReunionRepository reunionRepository;
48
49@    @Autowired
50    TipoUsuarioRepository tipoUsuarioRepository;
51
52@    @Autowired
53    AsistenciaRepository asistenciaRepository;
54
55@    @Autowired
56    UsuarioRepository usuarioRepository;
57
58@    @Autowired
59    InfoPacienteRepository infoPacienteRepository;
60
61@    @Autowired
62    HorarioRepository horarioRepository;
63
64@    @Autowired
65    RutinaRepository rutinaRepository;
66
67@    @Autowired
68    DietaRepository dietaRepository;
69
70@    @Autowired
71    TransaccionRepository transaccionRepository;
72
73@    @Autowired
74    NotificacionRepository notificacionRepository;
75
76

```

- Programar una nueva reunión (NUTRIOLOGO)

```
215@GETMapping("/crearReunion")
216 public String crearReunion(HttpServletRequest sesion, Model model) {
217     Reunion objReunion = new Reunion();
218     model.addAttribute("objReunion", objReunion);
219     return "usuario/programar_reunion";
220 }
221
222@POSTMapping("/grabarReunion")
223 public String grabarReunion(
224     HttpServletRequest sesion,
225     @ModelAttribute("objReunion") Reunion objReunion,
226     @RequestParam("dia") String dia,
227     Model model) {
228
229     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
230     if (objUsuario!=null) {
231         Tipousuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
232         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
233         model.addAttribute("esPaciente", esPaciente);
234
235         Reunion reunionUsuario = reunionRepository.findByNutriologoAndDiaAndEstado(objUsuario, dia, "Activo");
236
237         if(reunionUsuario==null) {
238             objReunion.setNutriologo(objUsuario);
239             objReunion.setEstado("Activo");
240             reunionRepository.save(objReunion);
241
242             Transaccion objTransaccion = new Transaccion();
243             objTransaccion.setFecha(LocalDateTime.now());
244             objTransaccion.setTipo("Creación");
245             objTransaccion.setUsuario(objUsuario);
246             objTransaccion.setReunion(objReunion);
247             transaccionRepository.save(objTransaccion);
248
249             Notificacion objNotificacion = new Notificacion();
250             objNotificacion.setTransaccion(objTransaccion);
251             objNotificacion.setRol("Paciente");
252             objNotificacion.setMensaje("Un nutriólogo a programado una nueva reunión.");
253             objNotificacion.setTimestamp(LocalDateTime.now());
254             objNotificacion.setEstado("Activo");
255             notificacionRepository.save(objNotificacion);
256         }
257     else {
258         model.addAttribute("error", "No se pudo programar la reunión.");
259         return "usuario/programar_reunion";
260     }
261 }
262 model.addAttribute("objUsuario", objUsuario);
263
264 List<Rutina> misRutinas = rutinaRepository.findByNutriologo(objUsuario);
265 model.addAttribute("misRutinas", misRutinas);
266
267 List<Dieta> misDietas = dietaRepository.findByNutriologo(objUsuario);
268 model.addAttribute("misDietas", misDietas);
269
270 return "perfil";
271 }
```

```

<section style="background-color: #white;">
    <h2>Programar una nueva reunión</h2>
    <form class="pure-form pure-form-stacked"
        th:action="@{/reunion/grabarReunion}"
        th:object="${objReunion}"
        method="POST" >

        <label for="motivo">Motivo</label>
        <input
            type="text"
            id="multi-first-name"
            th:field="*{motivo}"
            required="" />

        <label for="dia">Día</label>
        <select name="dia">
            <option th:value="LUNES">Lunes</option>
            <option th:value="MARTES">Martes</option>
            <option th:value="MIERCOLES">Miércoles</option>
            <option th:value="JUEVES">Jueves</option>
            <option th:value="VIERNES">Viernes</option>
            <option th:value="SABADO">Sábado</option>
            <option th:value="DOMINGO">Domingo</option>
        </select>

        <label for="hora">Hora</label>
        <input type="time" min="08:00" max="23:00" th:field="*{hora}" />

        <label for="enlace">En qué plataforma se va a hacer la reunión?</label>
        <a type="button" class="pure-button pure-button-primary" href="https://meet.google.com/Landing" target="_blank">Zoom</a>
        <br>
        <a type="button" class="pure-button pure-button-secondary" href="https://developers.zoom.us/docs/api/" target="_blank">Meet</a>
        <br>

        <input type="text" id="multi-enlace" th:field="*{enlace}" required="" />

        <br>
        <span class="alert" th:if="${error}" th:text="${error}"></span>

        <input type="submit" value="Guardar Reunión">
    </form>

```

- Desactivar reunión (NUTRÍÓLOGO)

```

273 @GetMapping("/desactivarReunion/{idreunion}")
274     public String eliminarHora(
275         HttpSession sesion,
276         @ModelAttribute ("idreunion") int idreunion,
277         Model model) {
278
279     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
280     if (objUsuario!=null) {
281         Reunion reunionDesac = reunionRepository.findByIdreunion(idreunion);
282         reunionDesac.setEstado("Inactivo");
283         reunionRepository.save(reunionDesac);
284         model.addAttribute("eliminado", "El horario ha eliminado con éxito");
285
286         TipoUsuario vistaUsuario = objUsuario.getTipousuario();
287         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
288         model.addAttribute("esPaciente", esPaciente);
289     }
290     model.addAttribute("objUsuario", objUsuario);
291
292     List<Reunion> objReunion = reunionRepository.findByNutriologoAndEstado(objUsuario, "Activo");
293     model.addAttribute("objReunion", objReunion);
294
295     List<Rutina> misRutinas = rutinaRepository.findByNutriologo(objUsuario);
296     model.addAttribute("misRutinas", misRutinas);
297
298     List<Dieta> misDietas = dietaRepository.findByNutriologo(objUsuario);
299     model.addAttribute("misDietas", misDietas);
300
301     return "perfil";
302 }

```

- Ver las reuniones programadas por un nutriólogo (PACIENTE)

```

    @GetMapping("/verReuniones")
    public String verReuniones(
        HttpSession sesion,
        @RequestParam("idusuario") int idusuario,
        Model model) {

        Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
        if (objUsuario!=null) {
            TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
            boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
            model.addAttribute("esPaciente", esPaciente);
        }

        Usuario nutriologoReunion = usuarioRepository.findByIdusuario(idusuario);
        model.addAttribute("nutriologoReunion", nutriologoReunion);

        List<Reunion> listaReuniones = reunionRepository.findByNutriologoAndEstado(nutriologoReunion, "Activo");
        model.addAttribute("listaReuniones", listaReuniones);

        Map<Integer, String> estadoAsistencia = listaReuniones.stream()
            .collect(Collectors.toMap(
                Reunion::getIdreunion,
                objReunion -> {
                    Asistencia asistencia = asistenciaRepository.findByPacienteAndReunionAndEstado(objUsuario, objReunion, "Activo");
                    return asistencia!=null ? asistencia.getEstado() : "Inactivo";
                }
            ));
        model.addAttribute("estadoAsistencia", estadoAsistencia);

        return "usuario/reuniones";
    }
}

```

- **Confirmar asistencia a una reunión**

```

109@ 109@     @GetMapping("/confirmarAsistencia")
110@ 110@     public String confirmarAsistencia(
111@ 111@         HttpSession sesion,
112@ 112@         @RequestParam("idreunion") int idreunion,
113@ 113@         Model model) {
114@ 114@ 
115@ 115@         Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
116@ 116@         if (objUsuario==null) {
117@ 117@             return "redirect:/index";
118@ 118@         }
119@ 119@ 
120@ 120@         TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
121@ 121@         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
122@ 122@         model.addAttribute("esPaciente", esPaciente);
123@ 123@ 
124@ 124@         Usuario paciente = userRepository.findById(objUsuario.getIdusuario())
125@ 125@             .orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
126@ 126@ 
127@ 127@         Reunion objReunion = reunionRepository.findById(idreunion)
128@ 128@             .orElseThrow(() -> new IllegalArgumentException("Reunión no encontrada"));
129@ 129@ 
130@ 130@         Asistencia asistencia = asistenciaRepository.findByPacienteAndReunionAndEstado(paciente, objReunion, "Activo");
131@ 131@         if (asistencia==null) {
132@ 132@ 
133@ 133@             asistencia = new Asistencia();
134@ 134@             asistencia.setPaciente(paciente);
135@ 135@             asistencia.setReunion(objReunion);
136@ 136@             asistencia.setEstado("Activo");
137@ 137@             asistencia.setFecha(LocalDateTime.now());
138@ 138@             asistenciaRepository.save(asistencia);
139@ 139@ 
140@ 140@             Transaccion objTransaccion = new Transaccion();
141@ 141@             objTransaccion.setFecha(LocalDateTime.now());
142@ 142@             objTransaccion.setTipo("Acceso a reunión");
143@ 143@             objTransaccion.setReunion(objReunion);
144@ 144@             objTransaccion.setUsuario(objUsuario);
145@ 145@             objTransaccion.setAsistencia(asistencia);
146@ 146@             transaccionRepository.save(objTransaccion);
147@ 147@ 
148@ 148@             Notificacion objNotificacion = new Notificacion();
149@ 149@             objNotificacion.setTransaccion(objTransaccion);
150@ 150@             objNotificacion.setRol("Nutriólogo");
151@ 151@             objNotificacion.setMensaje("El paciente " + paciente.getNombres() + " confirmó unirse a la reunión " + objReunion.getMotivo());
152@ 152@             objNotificacion.setTimestamp(LocalDateTime.now());
153@ 153@             objNotificacion.setEstado("Activo");
154@ 154@             notificacionRepository.save(objNotificacion);
155@ 155@ 
156@ 156@         TipoUsuario rolNutriologo = tipoUsuarioRepository.findByNomtipousu("Nutriólogo");
157@ 157@ 
158@ 158@         List<Usuario> listaNutriologos = usuarioRepository.findByTipousuarioAndEstado(rolNutriologo, "Activo");
159@ 159@         model.addAttribute("listaNutriologos", listaNutriologos);
160@ 160@ 
161@ 161@         return "usuario/lista_nutriologos";
162@ 162@     }
163@ 163@ }

```

- **Cancelar asistencia**

```
165 @GetMapping("/cancelarAsistencia")
166     public String cancelarAsistencia(
167         @RequestParam("idreunion") int idreunion,
168         HttpSession sesion) {
169
170     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
171     Reunion objReunion = reunionRepository.findByIdreunion(idreunion);
172
173     Asistencia objAsistencia = asistenciaRepository.findByPacienteAndReunionAndEstado(objUsuario, objReunion, "Activo");
174     if (objAsistencia!=null) {
175         objAsistencia.setEstado("Inactivo");
176         asistenciaRepository.save(objAsistencia);
177     }
178
179     return "menu";
180 }
```

-----

```
<section style="background-color: #white;">
    <h2>Reuniones disponibles</h2>

    <div th:each="reunion : ${ListaReuniones}">
        <p><h3 th:text="${reunion.motivo}"></h3><br>
        <p th:text="${reunion.dia}"></p>
        <p th:text="${reunion.hora}"></p>

        <!-- <span th:if="${estadoAsistencia[reunion.idreunion] == null}">Asistirás a esta reunión</span>-->
        <p><a class="pure-button pure-button-secondary"
            th:if="${estadoAsistencia[reunion.idreunion] == 'Inactivo'}"
            th:href="@{/reunion/confirmarAsistencia(idreunion=${reunion.idreunion})}">
            Confirmar Asistencia
        </a></p>
        <br><br>

        <p><a class="pure-button pure-button-primary"
            th:if="${estadoAsistencia[reunion.idreunion] == 'Activo'}"
            th:href="${reunion.enlace}"
            target="_blank">
            Entrar a la reunión
        </a></p>
        <br><br>

        <p><a class="pure-button pure-button-secondary"
            th:if="${estadoAsistencia[reunion.idreunion] == 'Activo'}"
            th:href="@{/reunion/cancelarAsistencia(idreunion=${reunion.idreunion})}">
            Cancelar Asistencia
        </a></p>
        <br><br>

        <hr>
    </div>
```

- Ver el perfil de un nutriólogo (PACIENTE)

```
182④ @GetMapping("/perfilNutriologo")
183 public String perfilNutriologo(
184     HttpSession session,
185     @RequestParam("idusuario") int idusuario,
186     Model model) {
187
188     Usuario objUsuario = (Usuario) session.getAttribute("usuario");
189     if (objUsuario != null) {
190         TipoUsuario vistaUsuario = objUsuario.getTipousuario();
191         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
192         model.addAttribute("esPaciente", esPaciente);
193     }
194
195     Usuario suPerfil = usuarioRepository.findByIdusuario(idusuario);
196     model.addAttribute("suPerfil", suPerfil);
197
198     List<Reunion> objReunion = reunionRepository.findByNutriologoAndEstado(suPerfil, "Activo");
199     model.addAttribute("objReunion", objReunion);
200
201     List<Rutina> susRutinas = rutinaRepository.findByNutriologo(suPerfil);
202     model.addAttribute("susRutinas", susRutinas);
203
204     List<Dieta> susDietas = dietaRepository.findByNutriologo(suPerfil);
205     model.addAttribute("susDietas", susDietas);
206
207     return "usuario/nutriologo";
208 }
209
```

#### - Ver listado de pacientes asistidos a las reuniones (NUTRÍÓLOGO)

```
302④ @GetMapping("/pacientesAsistidos")
303 public String pacientesAsistidos(
304     HttpSession session,
305     Model model) {
306
307     Usuario objUsuario = (Usuario) session.getAttribute("usuario");
308     if (objUsuario!=null) {
309         TipoUsuario vistaUsuario = tipoUsuarioRepository.findByIdtipousu(objUsuario.getTipousuario().getIdtipousu());
310         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
311         model.addAttribute("esPaciente", esPaciente);
312     }
313
314     List<Reunion> listaReuniones = reunionRepository.findByNutriologo(objUsuario);
315     model.addAttribute("listaReuniones", listaReuniones);
316
317     List<Asistencia> listaPacientes = listaReuniones.stream()
318         .flatMap(reunion -> asistenciaRepository.findByReunionAndEstado(reunion, "Activo").stream())
319         .collect(Collectors.toList());
320     Collections.reverse(listaPacientes);
321     model.addAttribute("listaPacientes", listaPacientes);
322
323     return "usuario/lista_pacientes2";
324 }
```

```

<!-->
<main>
    <section style="background-color: #white;">
        <h2>Pacientes asistidos a la reunión</h2>
        <div class="deslisable">
            <table class="pure-table">
                <thead>
                    <tr>
                        <th>Reunión</th>
                        <th>Hora y día</th>
                        <th>Paciente</th>
                        <th>Se unió en:</th>
                        <th>Estado</th>
                        <th>Historial de medidas</th>
                        <th>Perfil</th>
                    </tr>
                    <!-- <th>Para ti?</th>-->
                </thead>
                <tbody class="pure-table-border">
                    <tr th:each="asistencia: ${listaPacientes}">
                        <td><span th:text="${asistencia.reunion.motivo}"></span></td>
                        <td><span th:text="${asistencia.reunion.dia}"></span></td>
                        <td><span th:text="${asistencia.reunion.hora}"></span></td>
                        <td><span th:text="${asistencia.paciente.nombres}"></span>
                            <span th:text="${asistencia.paciente.apellidos}"></span></td>
                        <td><span th:text="${asistencia.fecha}"></span></td>
                        <td><span th:text="${asistencia.estado}"></span></td>
                        <td><a type="button" class="pure-button pure-button-secondary"
                            th:href="@{/reunion/versusSeguimiento(idusuario=${asistencia.paciente.idusuario})}">Ver</a></td>
                        <td><a th:href="@{/reunion/perfilPaciente(idusuario=${asistencia.paciente.idusuario})}">Ver</a></td>
                    </tr>
                </tbody>
            </table>
        </div>
    </section>

```

- Ver el perfil, horario e historial de medidas de un paciente (NUTRÍÓLOGO)

```

326 @GetMapping("/perfilPaciente")
327 public String perfilPaciente(
328     HttpSession sesion,
329     @RequestParam("idusuario") int idusuario,
330     Model model) {
331
332     Usuario objUsuario = (Usuario) sesion.getAttribute("usuario");
333     if (objUsuario != null) {
334         Tipousuario vistaUsuario = objUsuario.getTipousuario();
335         boolean esPaciente = vistaUsuario.getNomtipousu().equals("Paciente");
336         model.addAttribute("esPaciente", esPaciente);
337     }
338
339     Usuario suPerfil = usuarioRepository.findByIdusuario(idusuario);
340     model.addAttribute("suPerfil", suPerfil);
341
342     InfoPaciente suInfo = infoPacienteRepository.findByPacienteAndEstado(suPerfil, "Activo");
343     model.addAttribute("suInfo", suInfo);
344
345     Long asistencias = asistenciaRepository.countByPacienteAndEstado(suPerfil, "Activo");
346     model.addAttribute("asistencias", asistencias);
347
348     return "usuario/paciente";
349 }

```

```
351@GetMapping("/versuSeguimiento")
352 public String versuSeguimiento(
353     HttpSession sesion,
354     @RequestParam("idusuario") int idusuario,
355     Model model) {
356
357     Usuario suPerfil = usuarioRepository.findByIdusuario(idusuario);
358     if(suPerfil!=null) {
359         List<InfoPaciente> listaSeguimiento = infoPacienteRepository.findByPaciente(suPerfil);
360         model.addAttribute("listaseguimiento", listaSeguimiento);
361         model.addAttribute("suPerfil", suPerfil);
362
363         return "usuario/seguimiento";
364     }
365     model.addAttribute("error", "El usuario no existe.");
366     return "error";
367 }
368
369@GetMapping("/versuHorario")
370 public String versuHorario(
371     HttpSession sesion,
372     @RequestParam("idusuario") int idusuario,
373     Model model) {
374
375     Usuario suPerfil = usuarioRepository.findByIdusuario(idusuario);
376     if(suPerfil!=null) {
377         List<Horario> suHorario = horarioRepository.findByPacienteAndEstado(suPerfil, "Activo");
378         model.addAttribute("suHorario", suHorario);
379
380         return "usuario/vista_horario";
381     }
382     model.addAttribute("error", "El usuario no existe.");
383     return "error";
384 }
385
```

## RESULTADOS DE LAS PRUEBAS DE ACEPTACIÓN O CONFORMIDAD

Antes de pasar a producción, se realizaron pruebas exhaustivas en toda la aplicación para asegurar que todas las funcionalidades cumplieran con los requisitos establecidos. El programador revisó cada aspecto del sistema, identificando y solucionando posibles errores. Las pruebas incluyeron verificación de la interfaz de usuario, funcionalidad de las características clave (como el registro de usuarios, la gestión de rutinas y dietas, y la programación de reuniones).

## PLAN DE SOPORTE Y MANTENIMIENTO DEL PRODUCTO

Una vez que el MVP fue completado, se subió al repositorio en GitHub para permitir el acceso al código fuente y su mantenimiento continuo. Aunque no fue desplegado directamente en un entorno de producción, el acceso a través del repositorio permite que cualquier error o inconveniente detectado, ya sea por los usuarios finales o por el equipo de desarrollo, sea corregido de manera eficiente. El equipo técnico está disponible para realizar ajustes, mejoras y actualizaciones en el código según sea necesario. Este proceso garantiza que la plataforma siga siendo funcional, actualizada y en línea con las expectativas de los usuarios a largo plazo.

## 11. LECCIONES APRENDIDAS

### LO QUE FUNCIONÓ BIEN Y POR QUÉ

- **Distribución clara de tareas:** La asignación de roles y responsabilidades, permitió que las tareas se distribuyeran de manera eficiente. Esto facilitó el desarrollo de la plataforma y permitió que todos los miembros del equipo cumplieran con sus objetivos dentro de los plazos establecidos.
- **Interfaz intuitiva y usabilidad:** La plataforma resultó ser fluida y fácil de usar, lo que se destacó como un punto clave en el feedback tanto de los nutriólogos como de los usuarios. Esto fue posible gracias a la constante validación y pruebas de usabilidad durante el desarrollo, lo que permitió mejorar la interfaz en tiempo real.
- **Colaboración con el nutriólogo:** La estrecha colaboración con el nutriólogo fue crucial para asegurar que las rutinas y dietas fueran funcionales, realistas y adaptadas a las necesidades de los pacientes. Esto permitió que la plataforma fuera muy bien recibida, pues se alineó con las expectativas del nutriólogo y el objetivo del proyecto.

### DESAFÍOS ENFRENTADOS Y CÓMO SE SUPERARON:

- **Falta de tiempo para implementar funcionalidades adicionales:** Un desafío importante fue la eliminación de algunas funcionalidades deseadas, como las recomendaciones automáticas basadas en datos alterados. Este reto se superó priorizando las funcionalidades más críticas para el MVP y dejando las mejoras para futuras versiones del proyecto.
- **Integración con servicios externos:** Un desafío técnico fue la integración completa con servicios externos como Google para vincular cuentas. El equipo logró avanzar con el sistema de inicio de sesión básico, pero la integración más profunda se dejó para una fase posterior debido a la complejidad técnica y el tiempo limitado.

### CONSEJOS PARA FUTUROS PROYECTOS:

- **Planificación del tiempo:** Es fundamental planificar con suficiente antelación para prever posibles complicaciones y evitar la eliminación de funcionalidades importantes debido a restricciones de tiempo. Tener un cronograma flexible y priorizar las tareas ayudará a gestionar los tiempos de manera más efectiva.
- **Comunicación constante con los usuarios clave:** La colaboración cercana con los usuarios finales (en este caso, el nutriólogo) fue vital para el éxito del proyecto. Asegúrate de mantener una comunicación constante durante todo el proceso para validar ideas y funcionalidades, lo que facilita ajustes rápidos y evita desviaciones de los objetivos del proyecto.
- **Desarrollo iterativo y modular:** Adoptar un enfoque ágil, con iteraciones constantes, permite ajustar la plataforma según los comentarios y las necesidades que surgen en el camino. En proyectos complejos como este, la flexibilidad es clave, y la capacidad de ajustar el enfoque según el feedback mejora el producto final.
- **Pruebas y validación continua:** No subestimar la importancia de realizar pruebas de usabilidad y funcionalidad con los usuarios finales. Los test continuos aseguran que la plataforma sea

realmente efectiva y que el diseño sea intuitivo, lo que evita errores costosos en etapas posteriores del proyecto.

## 12. REFERENCIAS

- <https://www.who.int/es/news-room/fact-sheets/detail/physical-activity>
- <https://www.who.int/es/news-room/questions-and-answers/item/malnutrition>
- <https://elperuano.pe/noticia/206989-minsa-solo-el-26-de-peruanos-adultos-realiza-actividad-física>
- <https://www.gob.pe/institucion/minsa/noticias/630337-malnutricion-le-cuesta-al-peru-10-500-millones-de-dolares-equivalente-al-4-6-del-pbi>