# Nutri Gemini

Manuel Sanabria Montoya, Randall Sánchez Rivera
Universidad CENFOTEC, Agosto 2025

# Nutri Gemini

- Asistente nutricional con Inteligencia Artificial
- Permite automatizar procesos repetitivos, mantener una gestión completa de pacientes y registro de valoraciones
- Ayuda a pacientes a acceder de manera sencilla y agradable a los registro de sus valoraciones, planes nutricionales y gráficas de progreso.


Nutri Gemini

## Funcionalidades

- Gestión de Pacientes
- Gestión de Valoraciones
- Generación de recomendaciones con AI
- Cáclulo de macronutrientes con AI
- Planes nutricionales con AI
- Tablas de equivalencia con AI
- Monitoreo de progreso

## Características de la aplicación

- **Accesibilidad:** Modo de alto contraste, compatible con lectores de pantalla y navegación por teclado.
- **Usabilidad:** Desarrollado conforme a las 10 heurísticas de usabilidad definidas por Jakob Nielsen.
- **Patrones arquitectónicos:** Django Model View Template (MVT), Single Responisbility Principle (SRP), Factory Patter, Template Method Pattern.

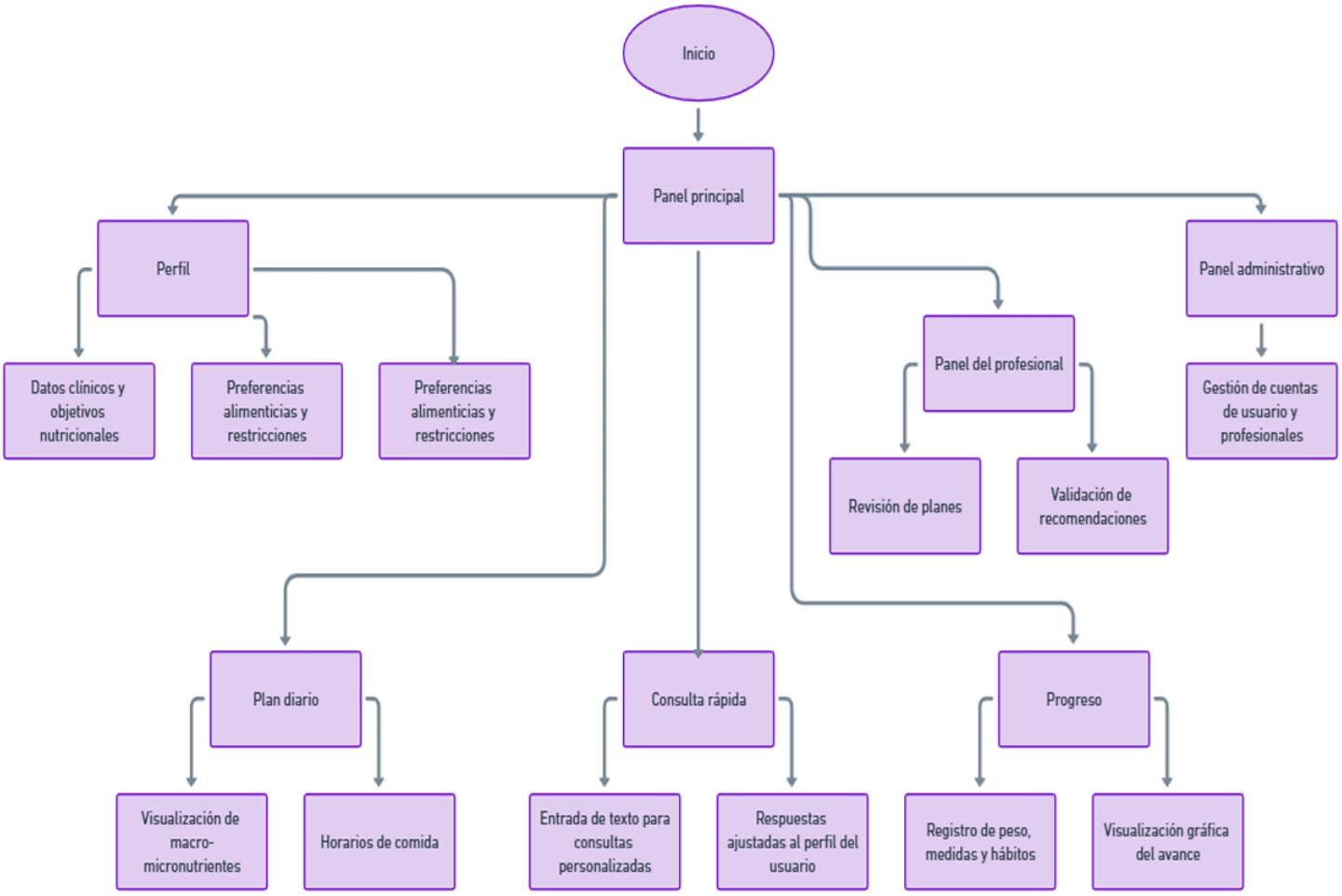
## Arquitectura de Información

Tabla de navegación según rol

| Pantalla | Acciones Permitidas | Usuarios |
|---|---|---|
| Bienvenida | Ingresar y ver información general de la aplicación | Todos |
| Inicio de sesión | Validar usuario. Ingresar según rol | Todos |
| Configuración de perfil | Ingresar datos clínicos y preferencias. Guardar | Nutricionista |
| Consulta textual | Escribir pregunta. Recibir respuesta IA | Paciente |
| Plan diario personalizado | Visualizar comidas. Ver macros por bloque | Paciente |
| Seguimiento | Registrar peso, síntomas, hábitos. Ver gráficos | Paciente |
| Panel profesional | Seleccionar paciente. Validar y ajustar planes | Nutricionista |
| Panel administrativo | Agregar/modificar usuarios. Consultar métricas | Administrador |

# Arquitectura de Información

## Diagrama de organizacion




## Diagrama

## Tecnologías utilizadas

- **Backend:** Django + Python
- **AI:** Gemini en la aplicación y Warp para desarrollo
- **Base de Datos:** Postgres en Railway
- **Frontend:** Bootstrap 5 + JavaScript
- **Contenedores:** Docker + Docker Compose
- **Versionamiento:** GitHub
- **Prototipado:** Figma
- **Documentación:** Quarto + HTML

## Hallazgos

- **Diseño:** Importancia de Arquitectura de Información y beneficios.
- **Accesibilidad:** Sensibilización en temas de accesibilidad.
- **Experiencia del usuario:** Importancia de enfoque en usabilidad.
- **Desarrollo:** Uso de patrones y AI para el desarrollo.

# Enlaces

- **Enlace al repositorio de Github del proyecto**
- **Enlace a la documentación del proyecto**

Speaker notes