

Saborify

<https://github.com/RodrigoGB12/ProyectoFinalCurso>

Nome Alumno/a: Rodrigo Gajardo Blanco

Curso: 2º DAM

Módulo: Proxecto Final Ciclo

Contido

Contido	2
1. Introducción	3
2. Obxectivos	3
Objetivos alcanzados:	4
3. Situación previa	4
4. Tecnoloxías emplegadas	4
5. Solución proposta	4
6. Planificación do proxecto	4
7. Desenvolvemento e execución	5
8. Conclusións e reflexións	5
9. Bibliografía e Webgrafía	5
10. Anexo I – Manual técnico de instalación ou posta en marcha	5
11. Anexo II – Documentación de uso (manuais usuario)	5
12. Anexo III – Outra documentación	5

1. Introducción

El proyecto elegido fue “Saborify” una app de catering donde los usuarios podran consultar sus pedidos y crear nuevos viendo asi el catalogo completo y los administradores podran ver los datos de todos los clientes, platos, pedidos etc.

Análisis del proyecto

El proyecto surge de la necesidad de facilitar la gestión de los servicios de catering, especialmente en pequeñas y medianas empresas que no cuentan con soluciones tecnológicas específicas. Se identificó que muchas de estas empresas dependen todavía de métodos manuales para organizar pedidos, menús y comunicación con clientes, lo que conlleva errores, retrasos y una baja eficiencia operativa.

Necesidades identificadas:

- Un sistema centralizado para gestionar clientes, pedidos y menús.
- Una interfaz intuitiva para que el cliente pueda realizar pedidos fácilmente.
- Herramientas para mejorar la organización y reducir el trabajo manual.
- Una solución económica y escalable que pueda adaptarse a distintos tipos de empresas.

Solución planteada:

Desarrollar una aplicación web que cuente con un **backend en Java (Spring Boot)** y un **frontend en Angular**, comunicados mediante API REST. La solución incorpora una arquitectura basada en capas, separando lógica de negocio, persistencia de datos y presentación al usuario.

Casos de uso principales:

- Gestión de pedidos: crear, consultar, actualizar y eliminar.
- Administración de menús y platos.
- Visualización de pedidos por parte del cliente.

Este análisis ha servido como base para diseñar la arquitectura del sistema, elegir las tecnologías adecuadas y definir la estructura general del proyecto.

2. Obxectivos

- **Facilitar la gestión de pedidos y menús:** Permitir que los administradores de catering organicen los pedidos de manera eficiente sin depender de procesos manuales.
- **Reducir errores en la planificación:** Automatizar el registro y modificación de pedidos para evitar confusiones en la gestión de eventos.
- **Mejorar la comunicación con los clientes:** Proporcionar un sistema en el que los clientes puedan consultar menús, realizar pedidos y hacer modificaciones de manera sencilla.
- **Asegurar el acceso y la seguridad de la información:** Implementar un sistema de autenticación que garantice que solo los usuarios autorizados puedan gestionar la información del servicio de catering.
- **Optimizar el tiempo y los recursos:** Minimizar las tareas repetitivas y manuales, permitiendo a los administradores enfocarse en mejorar la experiencia del cliente.
- **Hacer accesible la digitalización del servicio de catering:** Ofrecer una plataforma económica y adaptable que pueda ser utilizada por pequeños y medianos negocios sin altos costos de inversión.

3. Situación previa

En la actualidad, la gestión eficiente de los servicios de catering es fundamental para garantizar la calidad y satisfacción de los clientes. Con el crecimiento del sector gastronómico y el aumento de la demanda de eventos personalizados, surge la necesidad de contar con herramientas digitales que faciliten la administración de pedidos, menús y clientes.

Tradicionalmente, la gestión de estos servicios se ha llevado a cabo mediante procesos manuales o con herramientas básicas como hojas de cálculo y sistemas no especializados. Esto genera ineficiencias en la organización, retrasos en los pedidos y una mayor probabilidad de errores en la planificación de eventos y control de inventarios. Además, la falta de una plataforma centralizada dificulta la comunicación entre clientes, proveedores y administradores, lo que puede afectar la calidad del servicio ofrecido.

Ante esta problemática, muchas empresas de catering buscan soluciones tecnológicas que permitan automatizar estos procesos, optimizar recursos y mejorar la experiencia tanto para los clientes como para los gestores del negocio. Sin embargo, las opciones comerciales disponibles suelen ser costosas o poco adaptables a las necesidades específicas de cada empresa.

4. Tecnologías empleadas

Backend (API REST - Java)

- **Java 17:** Lenguaje de programación moderno que ofrece mejoras en rendimiento, seguridad y funcionalidad.
- **Spring Boot:** Framework de Java que simplifica el desarrollo de aplicaciones web al proporcionar configuraciones predeterminadas y una integración sencilla con otros componentes del ecosistema Spring.
- **Spring Data JPA:** Biblioteca que facilita la interacción con bases de datos mediante la abstracción de consultas SQL y la integración con ORM como Hibernate.
- **Hibernate:** Framework ORM que permite mapear objetos de Java a tablas de bases de datos, reduciendo la complejidad en la gestión de datos.
- **H2:** Base de datos ligera y embebida utilizada para entornos de desarrollo, ideal para pruebas rápidas sin necesidad de configuración avanzada
- **Swagger:** Herramienta que facilita la documentación y prueba de APIs REST, proporcionando una interfaz interactiva para explorar y probar endpoints.

Frontend (Angular)

- **Angular 17:** Framework moderno para el desarrollo de aplicaciones web SPA (Single Page Applications) que permite crear interfaces dinámicas y modulares con una estructura mantenible.
- **TypeScript:** Lenguaje de programación basado en JavaScript que añade tipado estático, mejorando la robustez y mantenibilidad del código.
- **RxJS:** Biblioteca para la programación reactiva que permite manejar eventos y flujos de datos asíncronos de manera eficiente.
- **Bootstrap:** Framework de diseño frontend que proporciona una colección de componentes y estilos predefinidos para desarrollar interfaces modernas y responsivas con facilidad.

5. Solución propuesta

Este proyecto busca ofrecer una solución integral a través de una aplicación web compuesta por un backend desarrollado en Java con Spring Boot y un frontend en Angular. La API proporciona funcionalidades para la gestión de usuarios, autenticación, administración de menús y control de pedidos, asegurando un sistema robusto y escalable. Gracias al uso de tecnologías modernas como H2 para la persistencia de datos y Swagger para la documentación, la aplicación garantiza un alto rendimiento y facilidad de mantenimiento.

El frontend, construido con Angular y Angular Material, ofrece una interfaz intuitiva y atractiva para que los usuarios puedan interactuar de manera eficiente con la plataforma. Utilizando Bootstrap y RxJS, se logra una experiencia fluida y dinámica, adaptada a las necesidades de los clientes y administradores del servicio de catering.

Con este proyecto, se pretende simplificar la gestión de los servicios de catering, mejorando la organización y optimización de los procesos internos. Además, se busca proporcionar una herramienta adaptable y de fácil implementación para pequeñas y medianas empresas del sector.

6. Planificación do proxecto

- Planificación do proxecto cun diagrama de Gantt
- Descripción do modelo de desenvolvemento de software a implementar.
- Análise do proxecto:
 - Diagrama de casos de usos.
 - Universo de discurso da base de datos.
 - Diagrama de Entidad-Relación.
- Deseño do proxecto:
 - Modelo relacional da base de datos.
 - Diagrama de clases.
 - Diagramas de fluxo de cada caso de uso.
 - Mockups da interface.
- Presuposto completo (Hardware software e recursos humans)

Concepto	Especificaciones/Requerimientos	Costo Aproximado
Hardware		
Procesador	Intel Core i5 / AMD Ryzen 5 (o superior)	250€
Memoria RAM	8 GB (recomendado 16 GB)	90€
Almacenamiento	256 GB SSD (recomendado 512 GB SSD o superior)	90€
Placa Base	Compatible con procesador y memoria RAM	150€
Conectividad	Acceso a internet estable	20€/mes
Software		
Sistema Operativo	Windows 10/11, macOS, Linux (Ubuntu recomendado)	Gratis
IDE Backend	IntelliJ IDEA Community Edition o VS Code	Gratis
IDE Frontend	Visual Studio Code con extensiones de Angular	Gratis
Java 17	OpenJDK	Gratis
Spring Boot	Framework de código abierto	Gratis
Node.js y npm	Para gestión de dependencias de Angular	Gratis
Angular CLI	Para la creación y administración del frontend	Gratis

Universo de discurso de la base de datos

El universo de discurso de la base de datos del sistema de gestión de catering abarca todos los elementos, entidades y relaciones necesarias para representar de manera lógica y estructurada la operativa de un servicio de catering digitalizado.

Dentro de este universo se encuentran tanto los **usuarios** (clientes y administradores) como los **menús** disponibles, los **pedidos realizados**, y los elementos auxiliares que permiten gestionar la autenticación, la trazabilidad de las operaciones y la organización del catálogo gastronómico.

Este universo está limitado al contexto de una empresa de catering que necesita:

- Gestionar sus productos gastronómicos (platos, menús, categorías).
- Administrar pedidos realizados por los clientes.
- Registrar y autenticar a los usuarios del sistema.
- Realizar un seguimiento de los pedidos según su estado (pendiente, confirmado, entregado).
- Mantener un histórico de pedidos y configuraciones asociadas.

En resumen, el universo de discurso incluye todos los elementos que intervienen en la prestación del servicio de catering desde la plataforma digital, y que son necesarios para garantizar su correcto funcionamiento mediante un sistema de información relacional.

Diagrama de Gantt:



1.1 Fase 1: Análisis de Requisitos (Semana 1)

- **Identificación de Requisitos HW y SW:** Evaluación de las necesidades técnicas para el desarrollo del proyecto, incluyendo hardware (PC, conexión, memoria) y software (Java, Angular, IDEs, gestor de versiones).
 - **Análisis de Necesidades del Usuario:** Definición de las funcionalidades principales desde el punto de vista de los usuarios finales: clientes y administradores.
 - **Primeros Diagramas de Casos de Uso:** Esbozo de las principales funcionalidades y actores del sistema.
-

1.2 Fase 2: Diseño (Semanas 2-3)

- **Diseño de Base de Datos:** Creación del modelo entidad-relación para definir las tablas necesarias (usuarios, pedidos, menús, platos, etc.).
 - **Diagramas de Clases:** Representación UML de las clases que conformarán la lógica de negocio del sistema backend.
 - **Arquitectura del Sistema:** Definición de la separación lógica entre backend y frontend, patrones de diseño, uso de controladores REST, autenticación JWT, y estructura de servicios.
-

1.3 Fase 3: Implementación (Semanas 4-8)

- **Desarrollo del Backend (Java + Spring Boot):** Creación de la API REST con endpoints protegidos por JWT, persistencia con JPA/Hibernate, y configuración de la base de datos H2.
 - **Desarrollo del Frontend (Angular + Bootstrap):** Construcción de interfaces para login, gestión de pedidos, visualización de menús y administración. Uso de RxJS para manejar datos reactivos.
 - **Documentación de la API (Swagger):** Generación automática de documentación interactiva para facilitar pruebas y mantenimiento.
-

1.4 Fase 4: Pruebas (Semanas 9-10)

- **Pruebas Unitarias:** Validación de métodos y servicios tanto en backend como en frontend.
- **Pruebas de Integración:** Comprobación de flujos completos de usuario, como autenticación, creación de pedidos o administración de menús.

1.5 Fase 5: Despliegue (Semanas 11-12)

- **Preparación de Entorno de Producción:** Simulación de despliegue en servidor local o entorno cloud.
- **Presentación del Proyecto y Demo:** Demostración funcional del sistema frente a usuarios y/o tribunal evaluador.

1.6 Fase 6: Mantenimiento y Documentación Final (Semana 13 en adelante)

- **Corrección de Erros Detectados:** Solución de bugs encontrados durante la demo o el uso posterior.
- **Redacción de Documentación Técnica:** Entrega de memoria final con detalles del desarrollo, decisiones técnicas y funcionalidades implementadas.

Modelo de desarrollo

Para el desarrollo de este proyecto se ha optado por implementar un modelo incremental e iterativo, basado en los principios de las metodologías ágiles. Este enfoque permite una mayor flexibilidad durante el desarrollo, facilitando la incorporación de mejoras progresivas y adaptaciones según las necesidades que surjan a lo largo del proceso.

Justificación del modelo

El modelo incremental es especialmente adecuado para proyectos académicos y de tamaño medio como este, donde se desea obtener una primera versión funcional del sistema en poco tiempo y posteriormente ir ampliando sus funcionalidades. Esto permite validar partes del sistema de forma temprana y recoger feedback útil antes de avanzar a fases más complejas.

Asimismo, la combinación con prácticas de desarrollo ágil (como la planificación en fases cortas y la validación constante del software) permite:

- Mejor organización del trabajo y priorización de tareas.
- Entrega frecuente de versiones funcionales del sistema.
- Mejora continua mediante revisiones iterativas.
- Mayor control sobre errores y desviaciones del plan inicial.

Etapas principales del modelo aplicado

Análisis de Requisitos:

- Recogida y documentación de las necesidades del usuario.
- Establecimiento de funcionalidades clave.
- Identificación de requerimientos técnicos.

Diseño:

- Diseño del modelo de datos y de la arquitectura de la aplicación.
- Diagramas de clases, casos de uso y estructura del sistema.
- Planificación de la comunicación entre frontend y backend.

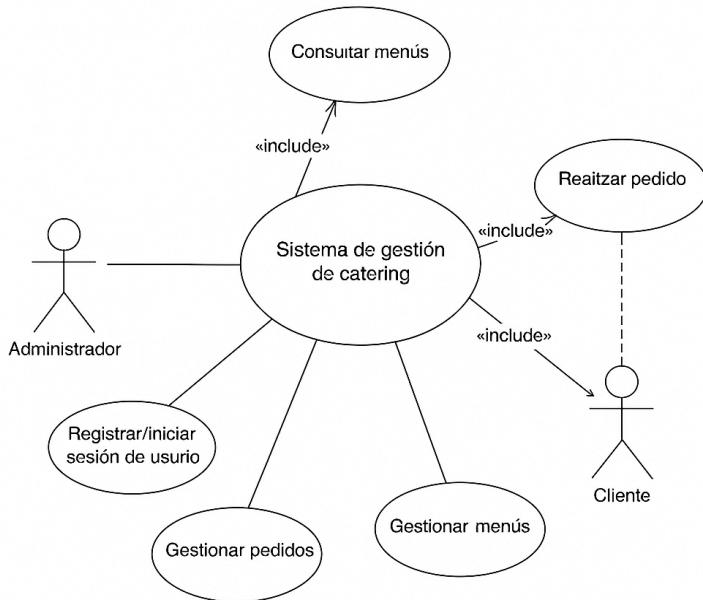
Implementación incremental:

- Desarrollo inicial de una versión mínima viable (MVP).
- Implementación por módulos: primero backend básico, luego autenticación, después frontend y finalmente la integración.
- Pruebas continuas durante la implementación.

Despliegue y Mantenimiento:

- Puesta en marcha del sistema.
- Documentación final del proyecto.
- Planificación de futuras mejoras y resolución de errores tras el uso.

Diagrama de casos de usos:



Universo de Discurso de la Base de Datos

El propósito general de este proyecto es gestionar la información relacionada con un servicio de pedidos de catering, incluyendo la administración de clientes, personal (cocineros y conductores), pedidos, artículos del menú y alérgenos alimentarios.

El sistema permitirá registrar pedidos personalizados de comida, asignar personal para su preparación y entrega, controlar los ingredientes con posibles alérgenos, y llevar un registro detallado del historial de pedidos y artículos seleccionados.

Requisitos de Información

La análisis de requisitos incluye tanto aspectos funcionales como no funcionales:

- Requisitos funcionales:
- Registro y gestión de clientes y personal.
- Creación y seguimiento de pedidos.
- Gestión de los artículos del menú y sus precios.
- Control de los alérgenos asociados a los productos.
- Asignación de cocineros o conductores a los pedidos.

Requisitos no funcionales:

- Seguridad y protección de datos personales (correo, DNI, etc.).
- Escalabilidad para soportar múltiples pedidos y registros simultáneos.
- Disponibilidad y fiabilidad en el acceso al sistema.

Usuarios del Sistema

Los usuarios finales del sistema incluyen:

- Administradores: gestionan la información del sistema (clientes, personal, menú, alérgenos, etc.).
- Clientes: realizan pedidos de los artículos disponibles en el menú.

Entidades Principales del Sistema

A continuación se describen las principales entidades y su información asociada:

- Clientes: Información de las personas que realizan pedidos.
Atributos: ID, nombre, correo electrónico, teléfono, dirección, DNI.
- Personal: Datos del personal disponible para preparar o entregar pedidos.
Atributos: ID, nombre, rol (Cocinero/Conductor), correo electrónico, teléfono, DNI.
- Artículos del Menú: Productos disponibles para pedidos.
Atributos: ID, nombre, descripción, imagen, precio.
- Alérgenos: Información de ingredientes con potencial alergénico presentes en los artículos.
Atributos: ID, nombre, descripción.
- Pedidos: Registro de cada pedido realizado por los clientes.
Atributos: ID, cliente_id, fecha del pedido, estado, monto total, personal asignado.
- Artículos por Pedido: Relación entre pedidos y artículos solicitados, incluyendo cantidades.
Atributos: ID, pedido_id, articulo_menu_id, cantidad, precio, personal asignado.
- Artículos-Menú-Alérgenos: Relación muchos a muchos entre los artículos del menú y sus alérgenos.
Atributos: ID, articulo_menu_id, alergeno_id.

Relaciones Clave del Sistema

- Un cliente puede tener múltiples pedidos.
- Cada pedido puede contener múltiples artículos del menú.
- Un artículo del menú puede estar asociado con varios alérgenos y viceversa.
- El personal puede estar vinculado a un pedido como responsable de su preparación o entrega.
- Cada alérgeno está relacionado a los productos que lo contienen, permitiendo advertencias al cliente.

Definición de alcance

El presente proyecto tiene como objetivo el desarrollo de una aplicación web destinada a optimizar la gestión integral de un servicio de catering. El sistema permitirá a las empresas del sector gestionar de forma eficiente pedidos, menús, alérgenos, clientes y personal, a través de una interfaz intuitiva y una arquitectura robusta basada en tecnologías modernas.

Alcance funcional:

- Gestión de clientes: alta, baja, modificación y consulta de datos personales.
- Gestión de personal (cocineros y conductores): administración de sus roles y datos.
- Administración de artículos del menú: creación y modificación de platos, descripción, precio e imagen.
- Gestión de alérgenos asociados a los artículos del menú.
- Registro y seguimiento de pedidos realizados por los clientes.
- Asignación de personal a los pedidos.
- Relación de artículos con los pedidos y cálculo de montos totales.
- Sistema de autenticación seguro basado en JWT.
- Interfaz web responsive para clientes y administradores, desarrollada con Angular.

Alcance tecnológico:

Backend desarrollado en Java 17 con Spring Boot, usando Spring Data JPA e Hibernate.

Base de datos relacional (H2 en desarrollo) con entidades normalizadas y relaciones bien definidas.

API documentada con Swagger para facilitar su comprensión y uso por terceros.

Frontend implementado con Angular 17, TypeScript, Bootstrap y RxJS para una experiencia fluida.

Exclusiones del alcance:

No se incluye la gestión de pagos en línea ni facturación automática.

No se prevé la integración con servicios externos como correo electrónico o mensajería.

El sistema se enfoca en el entorno de desarrollo y no contempla un entorno de producción en la nube.

Diagrama de base de datos/Entidad-Relacion:

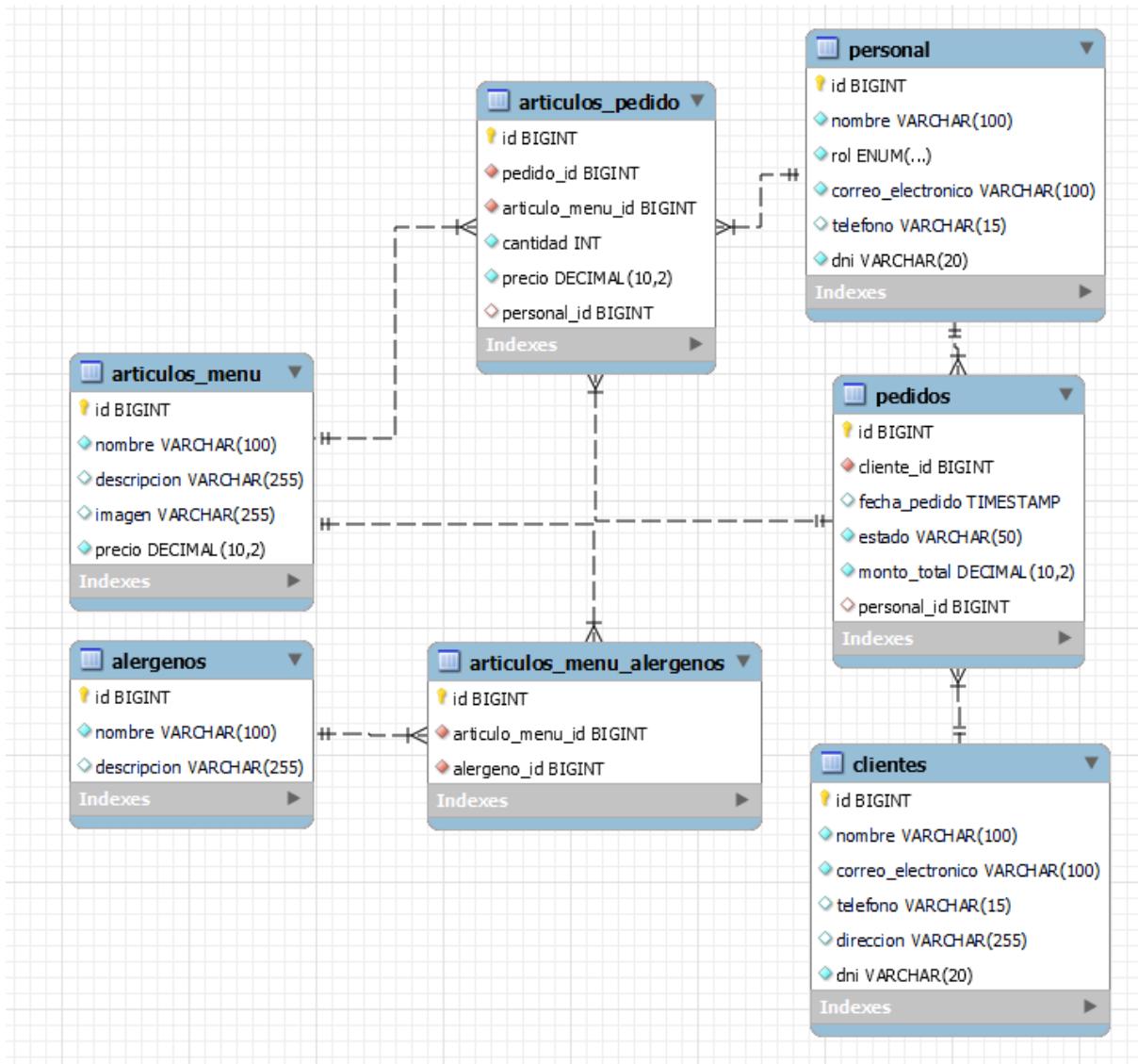
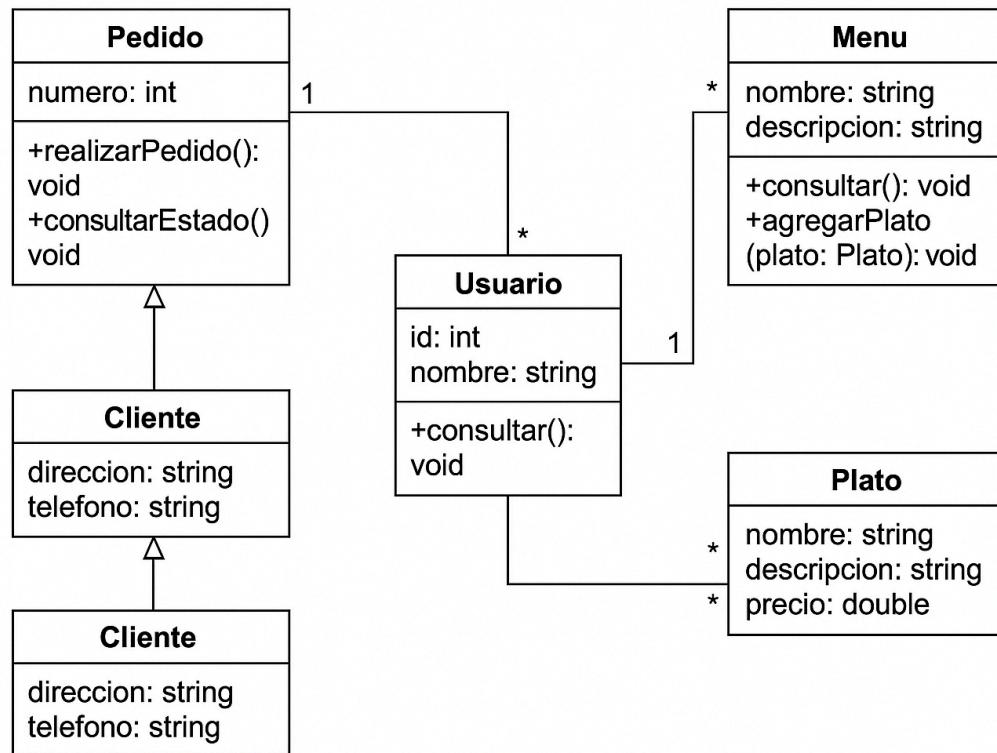
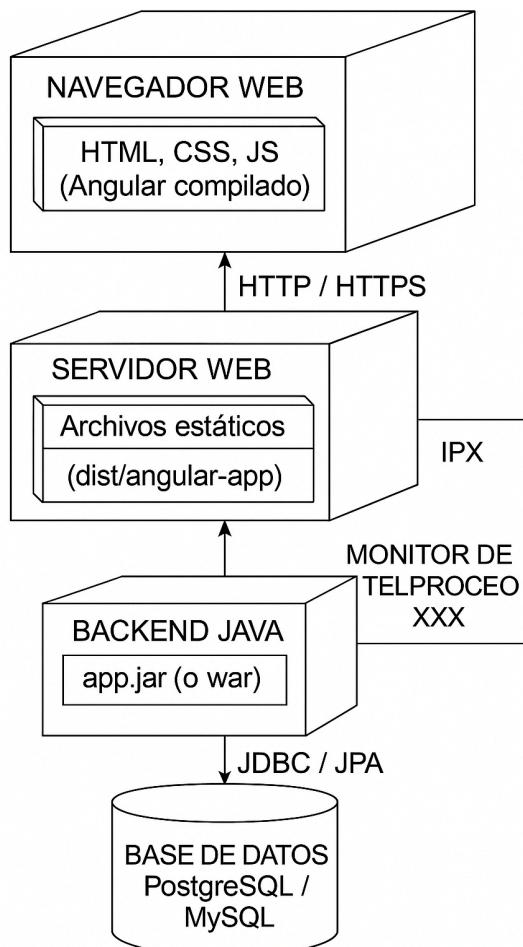


Diagrama de clases:



7. Desenvolvemento e execución

Diagrama de despliegue:



8. Conclusións e reflexións

La realización de este proyecto ha supuesto una experiencia muy enriquecedora tanto a nivel técnico como organizativo. A lo largo de su desarrollo se ha podido aplicar una gran variedad de conocimientos adquiridos previamente, desde el diseño de bases de datos relacionales hasta la implementación de una arquitectura cliente-servidor utilizando tecnologías modernas como Spring Boot en el backend y Angular en el frontend.

Uno de los aprendizajes más valiosos ha sido comprender cómo interactúan los distintos componentes de una aplicación web completa: base de datos, servidor, API REST, interfaz de usuario, seguridad y despliegue. También se ha reforzado la importancia de una planificación detallada y de un desarrollo iterativo para detectar y resolver errores a tiempo.

En cuanto a las dificultades encontradas, destacan principalmente dos:

Subida de imágenes: integrar correctamente la funcionalidad de carga y visualización de imágenes dentro del flujo del backend y frontend presentó ciertos retos, especialmente en la gestión del almacenamiento y en los formatos de envío de imágenes, se intentaron guardar en la carpeta del proyecto angular manualmente para luego descubrir que esto no era posible.

Despliegue en producción: poner el proyecto en marcha en un entorno productivo fue uno de los pasos más complejos. Hubo que cancelar esto al final por falta de tiempo y problemas en las maquinas de ubuntu

A pesar de estos retos, el resultado final es un sistema funcional que cumple con la mayoría de objetivos planteados al inicio del proyecto. La experiencia ha contribuido significativamente a mejorar la capacidad para afrontar proyectos reales y resolver problemas técnicos de forma autónoma.

9. Bibliografía e Webgrafía

[Formato APA](#)

10. Anexo I – Manual técnico de instalación ou posta en marcha

Este manual detalla paso a paso cómo preparar tu entorno local para ejecutar y desarrollar el proyecto, tanto en la parte del backend (Java + Spring Boot) como en el frontend (Angular). Se incluye la instalación de herramientas clave como IntelliJ IDEA, Visual Studio Code, Node.js, Angular CLI y configuración de base de datos MySQL.

1. Instalación de Java JDK (Java Development Kit)

Versión recomendada: JDK 17

- Ve al sitio oficial: <https://jdk.java.net/17/>
- Descarga la versión adecuada para tu sistema operativo.
- Instala siguiendo las instrucciones del instalador.
- Verifica la instalación desde la terminal/cmd:

2. Instalación de IntelliJ IDEA (Community Edition)

- Accede a: <https://www.jetbrains.com/idea/download/>
- Descarga la versión gratuita Community Edition.
- Ejecuta el instalador y sigue los pasos por defecto.
- Al abrir por primera vez, instala los siguientes plugins si se solicitan:
 - Spring Boot
 - Maven
- Abre tu proyecto desde “Open” y selecciona la carpeta del backend.

3. Instalación de Node.js y npm

Versión recomendada: Node.js v16 o superior

- Ve a: <https://nodejs.org/>
- Descarga la versión LTS recomendada.
- Ejecuta el instalador
- Verifica que se haya instalado correctamente:
`node -v | npm -v`

4. Instalación de Angular CLI

- Una vez tengas Node.js, instala Angular CLI con el siguiente comando:

```
npm install -g @angular/cli
```

- Verifica la instalación:

```
ng version
```

5. Instalación de Visual Studio Code (VS Code)

- Descarga desde: <https://code.visualstudio.com/>
- Instala siguiendo los pasos por defecto.
- Abre la carpeta del frontend con “Open Folder”.

6. Clonación del proyecto

Abre una terminal y clona los repositorios:

7. Ejecución del Backend (Spring Boot)

- Abre IntelliJ IDEA y abre la carpeta del backend.
- Configura el archivo application.properties con los datos de H2.
- Ejecuta la aplicación desde el botón "Run" o usando:
 - mvn spring-boot:run
- La API estará disponible en: <http://localhost:8080>

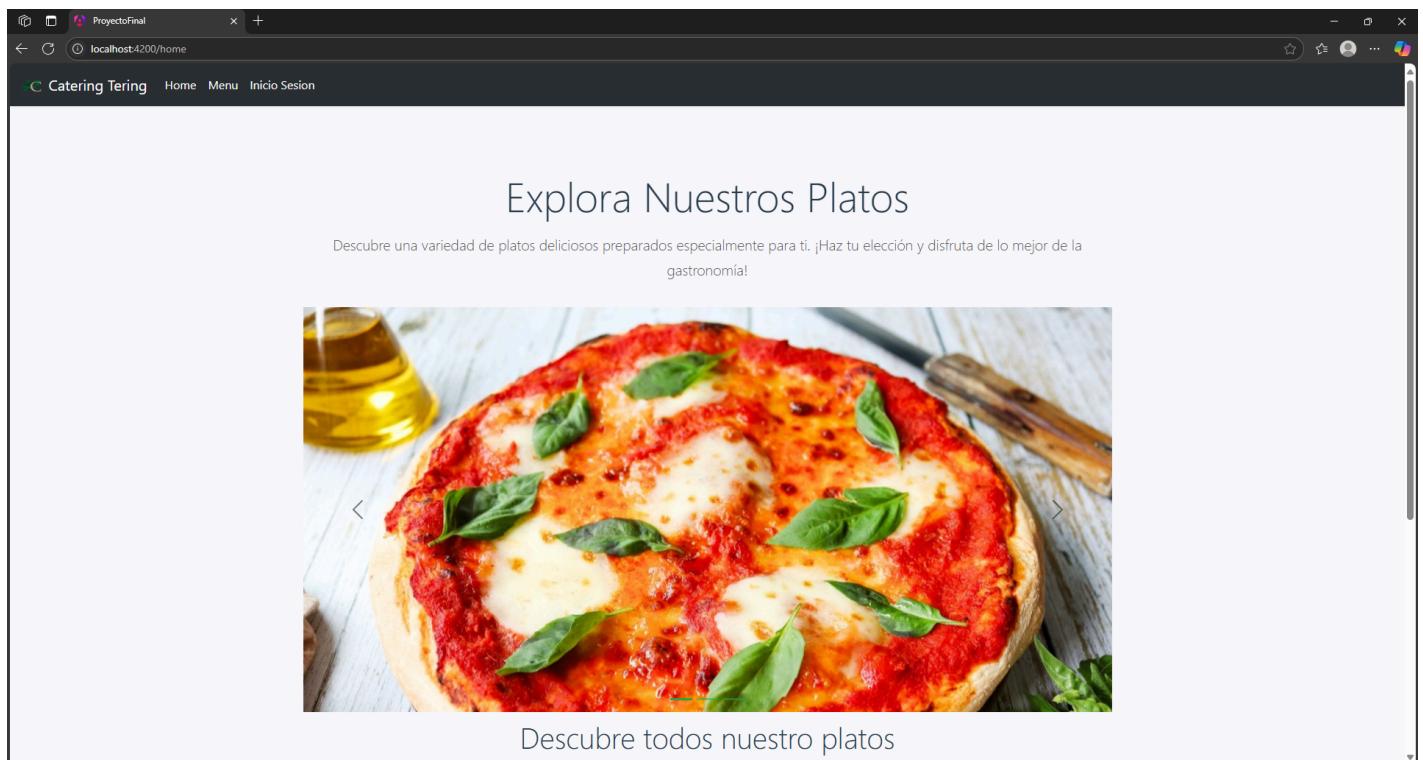
8. Ejecución del Frontend (Angular)

- Abre VS Code y navega a la carpeta del frontend.
- Instala las dependencias:
 - npm install
- Ejecuta el servidor de desarrollo:
 - ng serve
- Abre el navegador en: <http://localhost:4200>

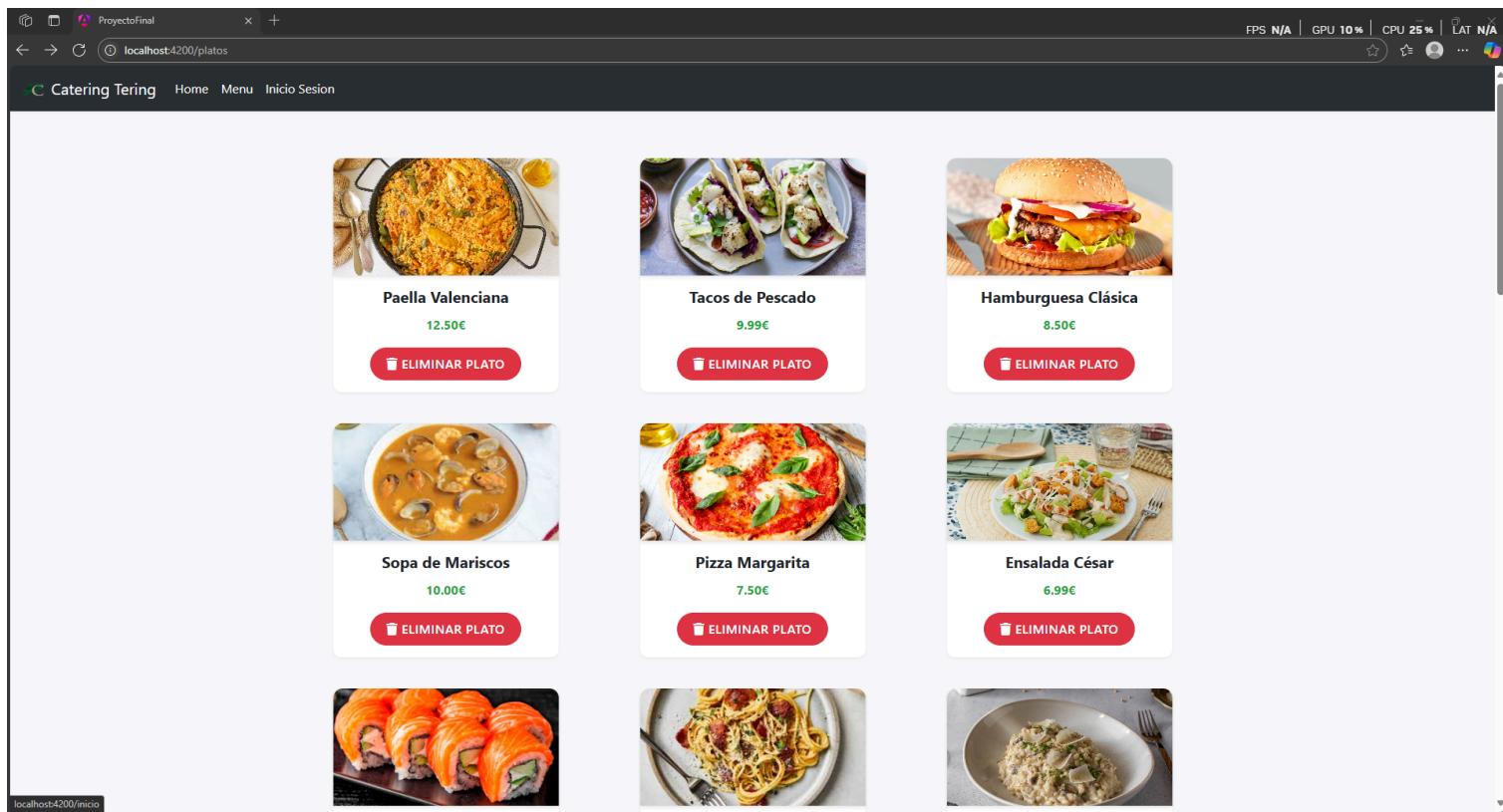
11. Anexo II – Documentación de uso (manuais usuario)

Manual de usuario:

Esta es la primera pagina la de inicio donde vemos unos pocos platos, en la barra de arriba podremos ir al apartado Menu e Inicio Sesion



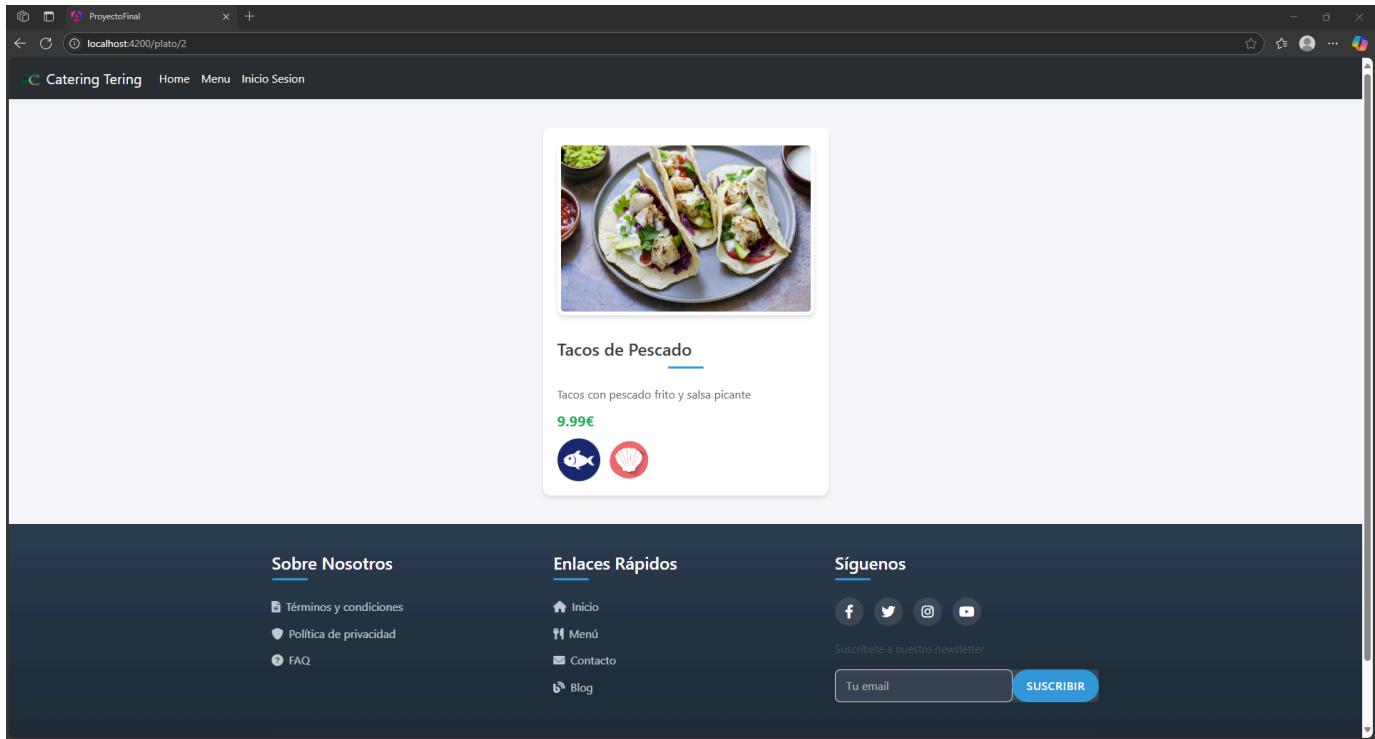
Esta es la Pagina de menu donde vemos los platos y al clickar encima de ellos podremos entrar para ver sus detalles y sus alergenos:



The screenshot shows a web-based catering menu system. The interface includes a header with the Xunta de Galicia logo, the IES San Mamede logo, and a European Union flag. Below the header, there's a navigation bar with links like 'Catering Tering', 'Home', 'Menu', and 'Inicio Sesión'. The main content area displays a grid of nine food items, each with a small image, the name of the dish, its price, and a red 'ELIMINAR PLATO' (Delete Dish) button.

Plato	Precio	Opciones
Paella Valenciana	12.50€	ELIMINAR PLATO
Tacos de Pescado	9.99€	ELIMINAR PLATO
Hamburguesa Clásica	8.50€	ELIMINAR PLATO
Sopa de Mariscos	10.00€	ELIMINAR PLATO
Pizza Margarita	7.50€	ELIMINAR PLATO
Ensalada César	6.99€	ELIMINAR PLATO
(Plato 7)		
(Plato 8)		

Aquí vemos los detalles del pedido con sus alergenos:



The screenshot shows a product detail page for 'Tacos de Pescado'. The product image shows three fish tacos on a plate with salsa and guacamole. Below the image, the product name 'Tacos de Pescado' is displayed, followed by a short description: 'Tacos con pescado frito y salsa picante'. The price is listed as '9.99€'. Two small icons representing fish and a shell are shown below the price.

Sobre Nosotros

- Términos y condiciones
- Política de privacidad
- FAQ

Enlaces Rápidos

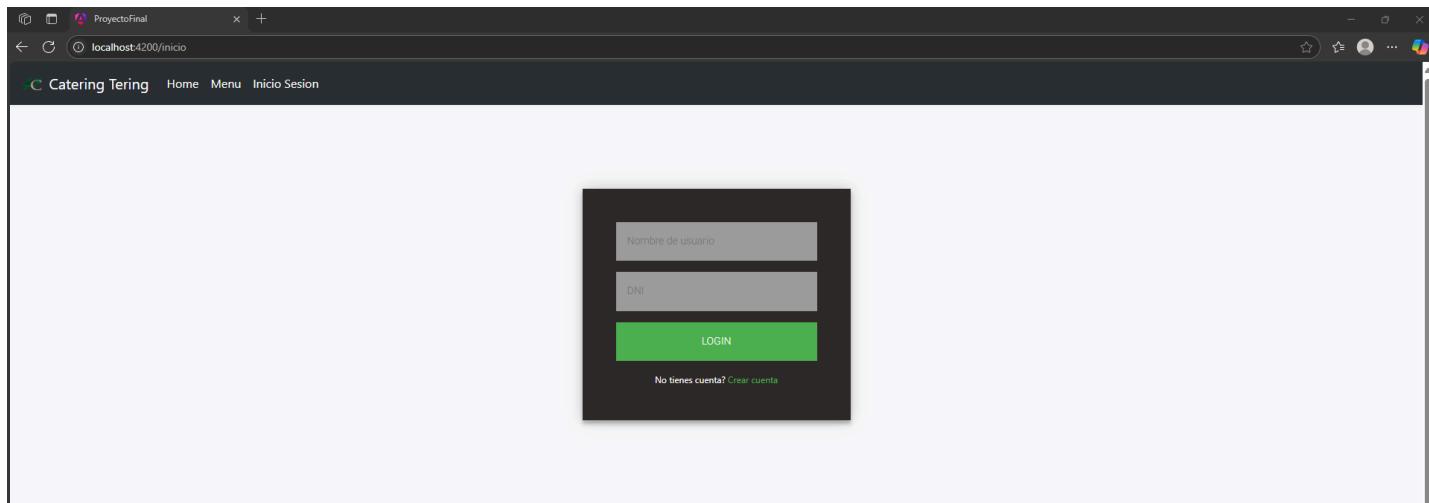
- Inicio
- Menú
- Contacto
- Blog

Síguenos

Suscríbete a nuestro newsletter

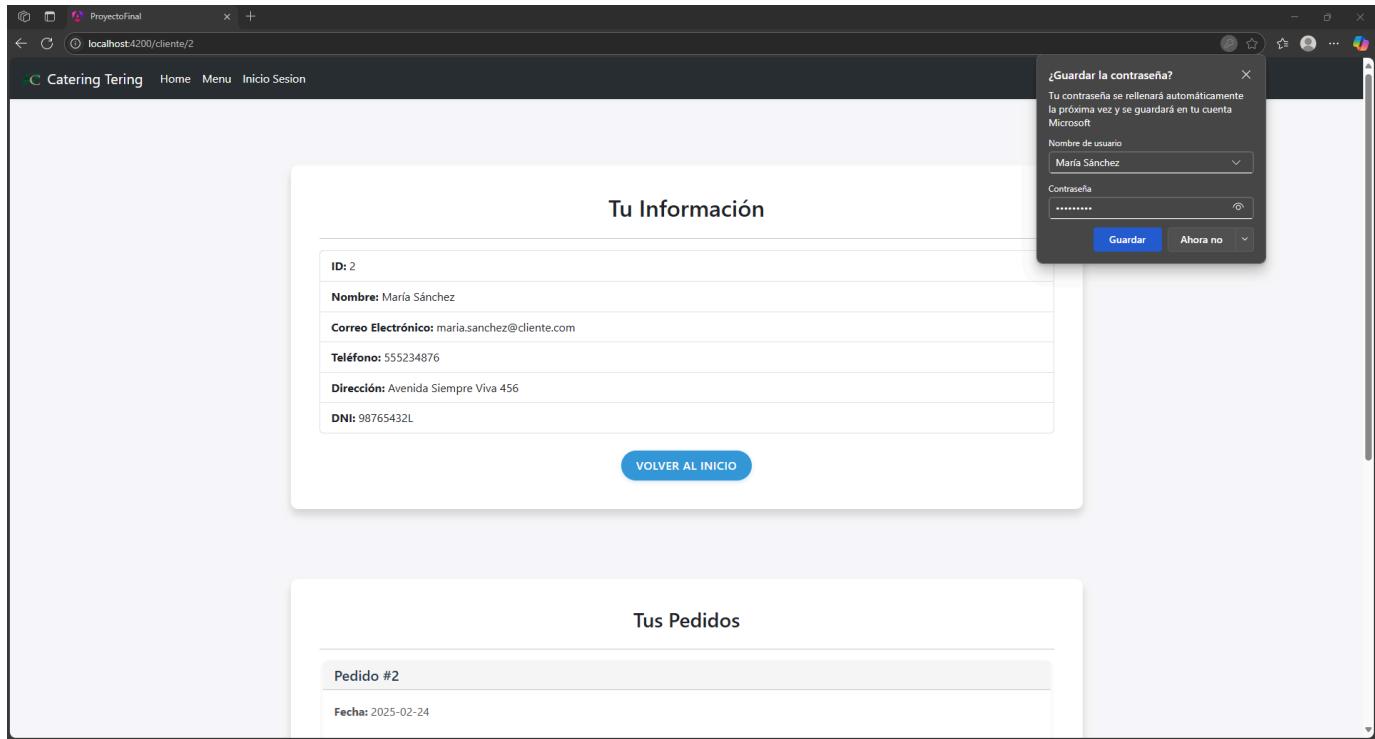
Tu email SUSCRIBIR

Este es el inicio de sesión, al poner el nombre del cliente y su dni podremos ver sus pedidos y información, al poner admin/admin accederemos al panel de administración



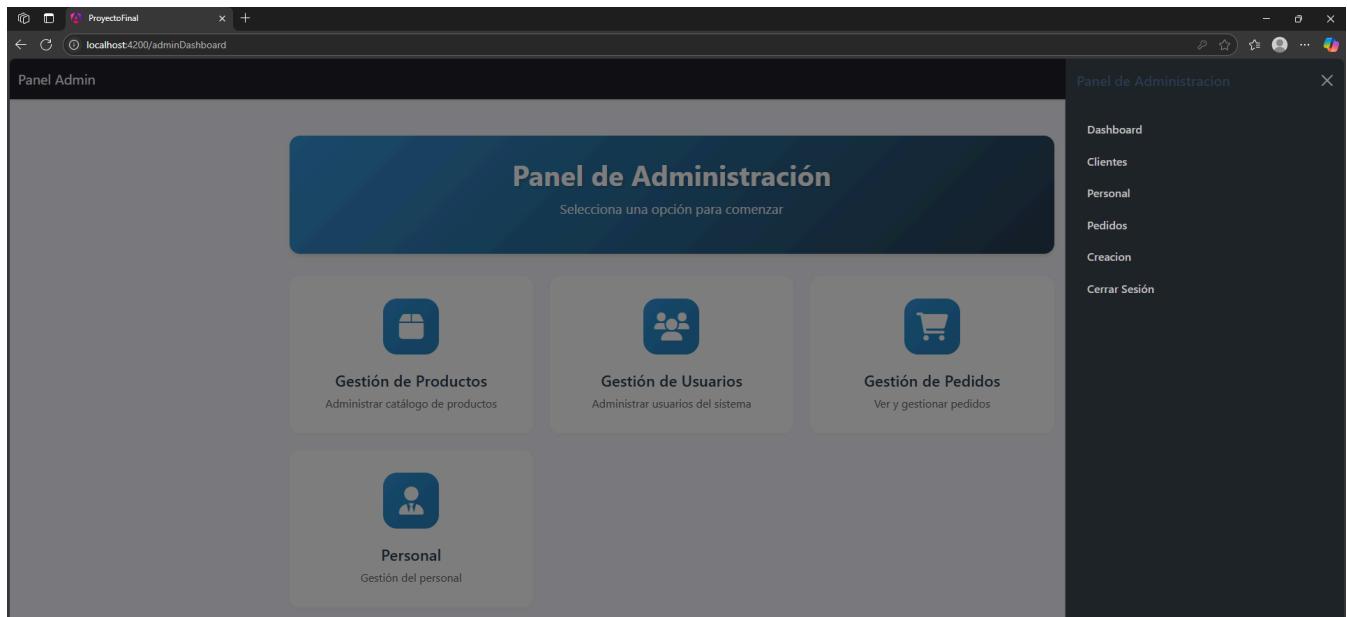
The screenshot shows a login form with a dark background. It has three input fields: 'Nombre de usuario', 'DNI', and a green 'LOGIN' button. Below the 'LOGIN' button, there is a link 'No tienes cuenta? [Crear cuenta](#)'.

Aquí vemos lo que se ve al acceder a la casa cliente:



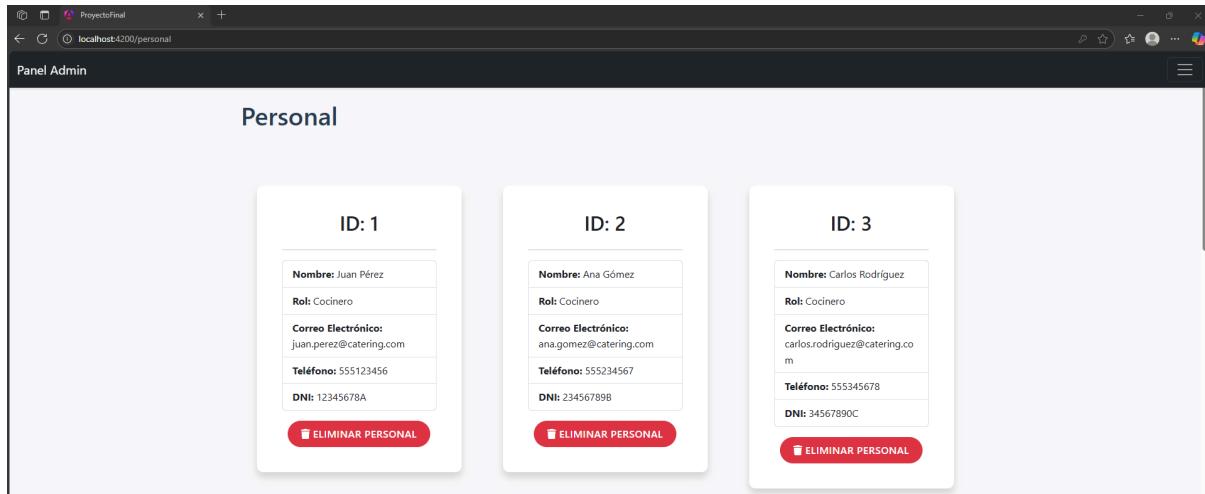
The screenshot shows a web browser window titled "ProyectoFinal" at "localhost:4200/cliente/2". The main content area displays a form titled "Tu Información" with fields for ID (2), Name (María Sánchez), Electronic Mail (maria.sanchez@cliente.com), Phone (555234876), Address (Avenida Siempre Viva 456), and DNI (98765432L). Below the form is a blue button labeled "VOLVER AL INICIO". A modal dialog titled "¿Guardar la contraseña?" is open on the right, asking if the password should be saved automatically for future logins. It contains fields for "Nombre de usuario" (set to María Sánchez) and "Contraseña" (set to a masked password). Buttons for "Guardar" (Save) and "Ahora no" (Not Now) are at the bottom.

Este es el panel de administración, a la derecha veremos todas las opciones que hay:



The screenshot shows a web browser window titled "ProyectoFinal" at "localhost:4200/adminDashboard". The main area is a dark-themed "Panel de Administración" with a central button "Selecciona una opción para comenzar". Below it are four cards: "Gestión de Productos" (Administrar catálogo de productos), "Gestión de Usuarios" (Administrar usuarios del sistema), "Gestión de Pedidos" (Ver y gestionar pedidos), and "Personal" (Gestión del personal). To the right is a sidebar titled "Panel de Administración" containing a vertical list of options: Dashboard, Clientes, Personal, Pedidos, Creacion, and Cerrar Sesión.

Las opciones de clientes, personal y pedidos mostraran un menu similar ha este con cada uno de sus datos y un boton de borrar:



The screenshot shows a web browser window titled 'Panel Admin' with the URL 'localhost:4200/personal'. The page is titled 'Personal' and displays three entries, each in a card-like box:

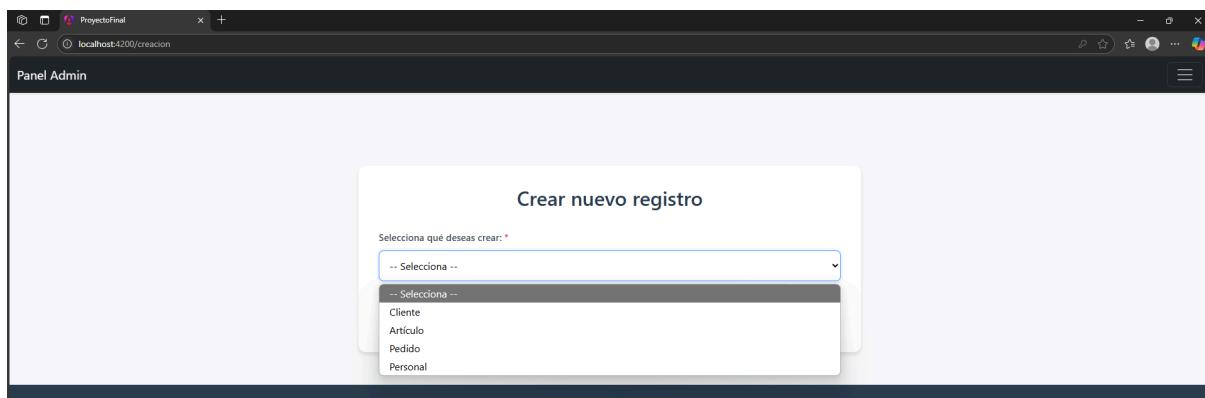
- ID: 1**
 - Nombre:** Juan Pérez
 - Rol:** Cocinero
 - Correo Electrónico:** juan.perez@catering.com
 - Teléfono:** 555123456
 - DNI:** 12345678A

ELIMINAR PERSONAL
- ID: 2**
 - Nombre:** Ana Gómez
 - Rol:** Cocinero
 - Correo Electrónico:** ana.gomez@catering.com
 - Teléfono:** 555234567
 - DNI:** 23456789B

ELIMINAR PERSONAL
- ID: 3**
 - Nombre:** Carlos Rodríguez
 - Rol:** Cocinero
 - Correo Electrónico:** carlos.rodriguez@catering.co.m
 - Teléfono:** 555345678
 - DNI:** 34567890C

ELIMINAR PERSONAL

Esta es la pestaña de creacion donde podremos seleccionar que crear y meter en los campos sus datos:

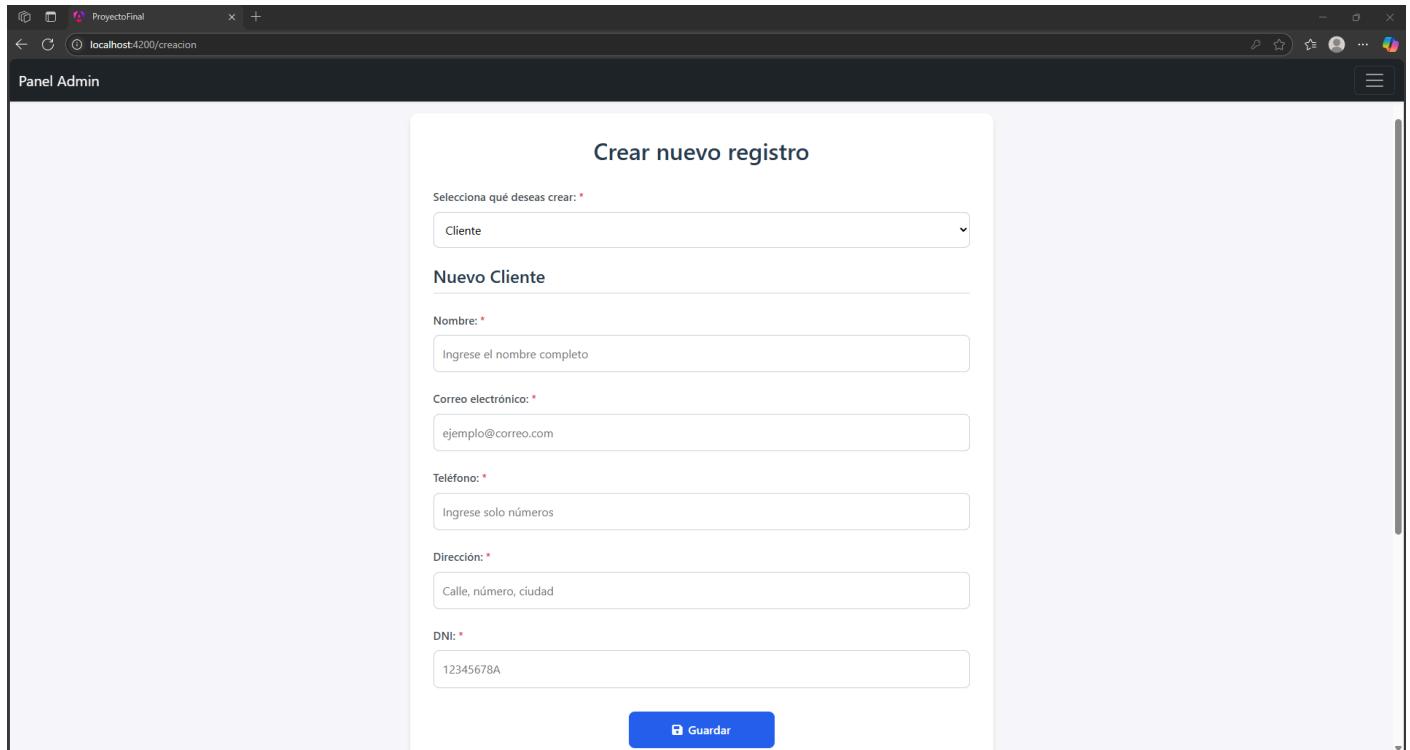


The screenshot shows a web browser window titled 'Panel Admin' with the URL 'localhost:4200/creacion'. The page is titled 'Crear nuevo registro' and contains a dropdown menu:

Selecciona qué deseas crear: *

- Selecciona --
- Selecciona --
- Cliente
- Artículo
- Pedido
- Personal

En algunos campos como por ejemplo si creamos un pedido deberemos meter el nombre completo del cliente:

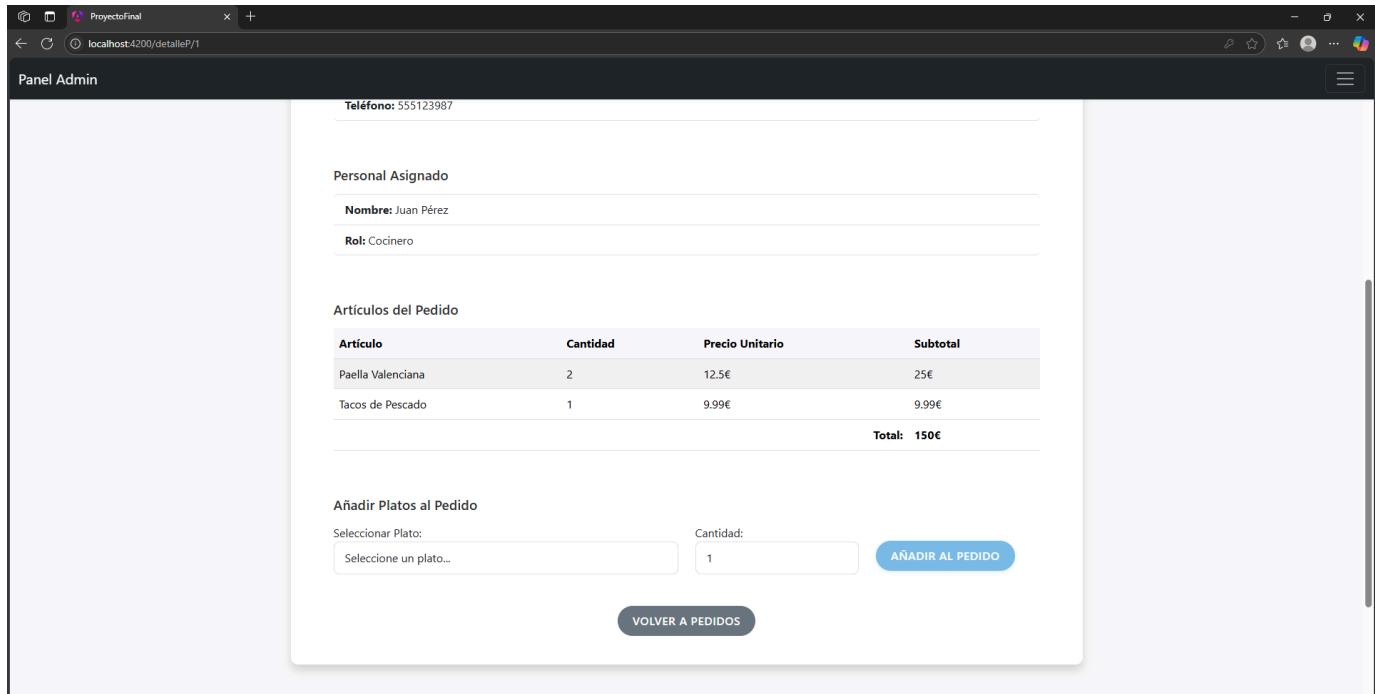


The screenshot shows a web browser window titled "ProyectoFinal" with the URL "localhost:4200/creacion". The page is titled "Panel Admin" and contains a form titled "Crear nuevo registro". A dropdown menu labeled "Cliente" is open. The form fields are as follows:

- Nuevo Cliente**
- Nombre:** * (Input field placeholder: "Ingrese el nombre completo")
- Correo electrónico:** * (Input field placeholder: "ejemplo@correo.com")
- Teléfono:** * (Input field placeholder: "Ingrese solo números")
- Dirección:** * (Input field placeholder: "Calle, número, ciudad")
- DNI:** * (Input field placeholder: "12345678A")

A blue "Guardar" button is located at the bottom right of the form.

Al ver los pedidos podremos ver esto donde podremos agregar los platos a este:



The screenshot shows a web application interface titled "Panel Admin". At the top, there's a header with the phone number "Teléfono: 555123987". Below it, under "Personal Asignado", there are fields for "Nombre: Juan Pérez" and "Rol: Cocinero". A table titled "Artículos del Pedido" lists items with their quantities and prices: Paella Valenciana (2 units at 12.5€ each), Tacos de Pescado (1 unit at 9.99€ each), and a total of 150€. Below this, a section titled "Añadir Platos al Pedido" allows adding new dishes. It has a dropdown for "Seleccionar Plato" with "Seleccione un plato..." and a quantity input field set to "1". A blue button labeled "AÑADIR AL PEDIDO" is next to the input field. At the bottom, a dark button says "VOLVER A PEDIDOS". The browser address bar shows "localhost:4200/detalleP/1".

Esto seria todo el manual de usuario como ultimo detalle en los platos deberemos poner el nombre de la imagen que queremos meter y luego meterla en la carpeta public/images del proyecto y lo volvemos a lanzar.