

ConsultMe

Consulta el estado de pedidos y servicios

Proyecto Final de Curso

Autor: Alejandro Tejada Rodríguez

Grado Superior: Desarrollo de Aplicaciones Web

Curso académico 2021 - 2022

ÍNDICE

1. INTRODUCCIÓN	1
1.1 Resumen y justificación del proyecto.....	1
2. ANÁLISIS	3
2.1 Objetivos específicos	3
2.2 Requisitos Funcionales.....	4
2.3 Requisitos no Funcionales.....	7
2.4 Fases de realización	8
2.4.1 Planificación	9
2.5 Tecnologías y herramientas utilizadas.....	9
2.6 Casos de Uso	12
3. DISEÑO	14
3.1 Prototipo Frontend	14
3.2 Arquitectura	15
3.2.1 Frontend	15
3.2.2 Backend.....	17
3.3 Modelo de datos	18
3.4 Vistas	20
3.4.1 Inicio	21
3.4.2 Usuario registrado - Mis consultas, realizar y guardar.....	23
3.4.3 Empresa – Estadísticas, mis consultas, creación y modificación	25
3.4.4 Perfil	27
4. IMPLEMENTACIÓN	28
4.1 Configuración del entorno de desarrollo	28
4.2 Desarrollo	28

4.2.1 Frontend	29
4.2.2 Backend.....	35
4.3 Configuración servidor y despliegue	37
5. PRUEBAS	39
5.1 Pruebas en navegadores.....	39
5.2 Evaluación	41
6. CONCLUSIONES	42
7. REFERENCIAS.....	43
8. ANEXOS	44
8.1 Manual de usuario	44
8.1.1 Iniciar sesión / registrarse.....	44
8.1.2 Realizar consulta / guardar consulta.....	45
8.1.3 Borrar consulta guardada	46
8.2 Manual de usuario administrador.....	47
8.2.1 Crear consulta	47
8.2.2 Modificar consulta.....	47
8.2.3 Estadísticas	48

1. INTRODUCCIÓN

1.1 Resumen y justificación del proyecto

Las grandes empresas avanzan a pasos agigantados, pero el pequeño negocio no tiene los recursos ni los conocimientos suficientes para seguirles. Muchos ni si quiera tienen un ordenador para gestionar sus servicios.

Actualmente, resulta inconcebible un día a día sin internet, y más en concreto, sin *smartphone*. Debido al auge de estos dispositivos, cada vez más personas utilizan los múltiples servicios que internet ofrece. Estar en la nube cada vez es más esencial.

Los pequeños negocios están desapareciendo, pues las personas (sobre todo los jóvenes), prefieren utilizar comercios que les den las facilidades que internet proporciona. El pequeño comercio debe entrar en la era tecnológica, en específico, en la nube.

ConsultMe pretende dar una solución para facilitar el estado del pedido o servicio. Pretende así, ahorrar tiempo a la hora de comunicarse con el cliente y éste comprobar su pedido.

Aunque este proyecto sea una aplicación web, ConsultMe está diseñada con la idea de ser escalable y fácilmente implementable en una aplicación móvil para aprovechar las funcionalidades del *smartphone*. Además de aprovechar también las del ordenador de sobremesa.

A lo largo del documento se presenta la aplicación realizada, exponiendo el análisis, los requisitos, las herramientas utilizadas en su desarrollo, la estructura y las pruebas realizadas.

Abstract

Large companies are advancing by leaps and bounds, but small businesses do not have the resources or the know-how to keep up with them. Many do not even have a computer to manage their services.

Nowadays, a day-to-day life without the Internet, and more specifically, without a smartphone, is inconceivable. Due to the rise of these devices, more and more people use the multiple services that the Internet offers. Being in the cloud is becoming more and more essential.

Small businesses are disappearing, as people (especially young people), prefer to use businesses that give them the facilities that the internet provides. Small businesses must enter the technological era, specifically in the cloud.

ConsultMe aims to provide a solution to facilitate the status of the order or service. It aims to save time when communicating with the customer and the customer can check his order.

Although this project is a web application, ConsultMe is designed with the idea of being scalable and easily implementable in a mobile application to take advantage of the functionalities of the smartphone. In addition to also take advantage of those of the desktop computer.

Throughout the document the application is presented, exposing the analysis, the requirements, the tools used in its development, the structure and the tests carried out.

2. ANÁLISIS

El objetivo principal de la aplicación es servir como plataforma para poder llevar un control del estado de los pedidos y servicios de una manera rápida y eficiente. Se pretende así, sobre todo después del coronavirus, ahorrar tiempo, llamadas y visitas innecesarias a los comercios.

2.1 Objetivos específicos

En función del tiempo disponible los objetivos se van a dividir en dos categorías:

Realizados:

- Un sistema de registro de usuarios (creación e inicio y cierre de sesión).
- Creación de un sistema de autenticación por sesión mediante token.
- Realizar consultas rápidas.
- Guardar y eliminar consultas de la lista personal.
- Crear y modificar una consulta de empresa.
- Gráficos de estadísticas de empresa.

Deseables (futuras mejoras):

- Editar el perfil de usuario.
- Un sistema para almacenar las imágenes de las empresas.
- Mostrar un listado de comercios compatibles en la página principal.
- Back office.
- Crear un menú de *Ajustes* para guardar las preferencias del usuario.
- Inicio de sesión por RRSS.
- Poder modificar la ordenación de las listas.

2.2 Requisitos Funcionales

RF01	
NOMBRE:	Registro de usuario
PRIORIDAD:	Alta
DESCRIPCIÓN:	La aplicación permitirá al usuario registrarse con un correo electrónico y contraseña, o RRSS, y ésta se guardará en la base de datos. Se guardará el token de autenticación para comunicarse con el sistema <i>backend</i> .
MODIFICACIONES:	El registro por RRSS no está implementado.

RF02	
NOMBRE:	Inicio de sesión
PRIORIDAD:	Alta
DESCRIPCIÓN:	El usuario podrá iniciar sesión con el correo electrónico y contraseña, o RRSS. Se guardará el token de autenticación para comunicarse con el sistema <i>backend</i> .
MODIFICACIONES:	El inicio de sesión por RRSS no está implementado.

RF03	
NOMBRE:	Cierre de sesión
PRIORIDAD:	Alta
DESCRIPCIÓN:	Cerrar la sesión actual. Se borrará el token de autenticación.
MODIFICACIONES:	-

RF04	
NOMBRE:	Cuenta de usuario
PRIORIDAD:	Media
DESCRIPCIÓN:	Consultar y modificar los datos de usuario. Si son modificados, actualizar los datos en la base de datos.
MODIFICACIONES:	No se ha implementado la modificación de los datos de usuario.

RF05	
NOMBRE:	Realizar consultas
PRIORIDAD:	Alta
DESCRIPCIÓN:	El usuario podrá realizar una consulta sin previo registro. Tras meter el código de la consulta, se validará y se mostrarán sus datos.
MODIFICACIONES:	-

RF06	
NOMBRE:	Seguir/guardar consultas
PRIORIDAD:	Alta
DESCRIPCIÓN:	Tras realizar una consulta (usuario registrado), podrá guardar la consulta en su lista.
MODIFICACIONES:	-

RF07	
NOMBRE:	Eliminar consultas guardada
PRIORIDAD:	Alta
DESCRIPCIÓN:	El usuario podrá eliminar las consultas guardadas.
MODIFICACIONES:	-

RF08	
NOMBRE:	Crear y modificar consultas de la empresa
PRIORIDAD:	Alta
DESCRIPCIÓN:	<p>El administrador de una empresa podrá crear una consulta de su servicio o pedido.</p> <p>Desde el modo de administrador podrá gestionar las consultas para modificar su información.</p>
MODIFICACIONES:	-

RF09	
NOMBRE:	Estadísticas
PRIORIDAD:	Media
DESCRIPCIÓN:	<p>En la página de inicio del modo administrador se mostrarán diferentes estadísticas e información de utilidad.</p>
MODIFICACIONES:	-

2.3 Requisitos no Funcionales

RNF01	
NOMBRE:	Seguridad
PRIORIDAD:	Alta
DESCRIPCIÓN:	<p>FRONTEND:</p> <p>Para utilizar la aplicación (menos las consultas rápidas) es necesario autenticarse.</p> <p>La sesión se mantendrá abierta durante 30 días o hasta que el usuario cierre manualmente la sesión.</p> <p>BACKEND:</p> <p>La primera capa será validar la autenticación.</p>
MODIFICACIONES:	Este control se realiza con JWT [11], tras crear un usuario o iniciar sesión el sistema <i>backend</i> devuelve uno nuevo.

RNF02	
NOMBRE:	Interfaz y usabilidad
PRIORIDAD:	Alta
DESCRIPCIÓN:	La aplicación será sencilla e intuitiva para el usuario común, mantenido pocos elementos en pantalla.
MODIFICACIONES:	-

RNF03	
NOMBRE:	Rendimiento
PRIORIDAD:	Alta
DESCRIPCIÓN:	Al no guardar grandes cantidades de datos, los accesos a la base de datos deben ser rápidos, consiguiendo así un rendimiento óptimo.
MODIFICACIONES:	-

2.4 Fases de realización

A continuación, se describen las fases seguidas a lo largo de la realización del proyecto:

- Estudio: Investigación sobre las posibles tecnologías a utilizar. Indagación sobre aplicaciones relacionadas.
- Análisis: Definición de requisitos con el objetivo de definir exactamente qué es lo que se desea construir.
- Diseño: Tras finalizar las fases de estudio y análisis, y con el listado de requisitos, se procede a diseñar la arquitectura de la aplicación. Esto incluye diversos diagramas que definirán el diseño de pantallas, interfaz y la base de datos.
- Implementación: Programación del sistema frontend y backend.
- Despliegue: Creación y configuración del servidor.
- Pruebas: Una vez finalizadas todas las fases anteriores, se ha realizado una serie de pruebas generales.
- Documentación: La documentación es algo indispensable para el correcto mantenimiento de cualquier producto software.

2.4.1 Planificación

Tras la definición de las fases, se ha realizado una planificación para el proyecto:

	Estudio	Análisis	Diseño	Impl.	Despliegue	Pruebas	Documentación
Febrero	X	X					
Marzo		X	X				
Abril				X	X		
Mayo				X	X	X	X

2.5 Tecnologías y herramientas utilizadas

- Visual Studio Code [1]

Editor de código fuente que incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

- Node.js [2]

Entorno en tiempo de ejecución multiplataforma, para la capa del servidor basado en el lenguaje de programación JavaScript, asíncrono, con E/S de datos en una arquitectura orientada a eventos.

- Angular 13 [3]

Framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

- dialog (ngneat) [4]

Modales para aplicaciones Angular.

- ngx-charts (swimlane) [5]

Framework de gráficos para aplicaciones Angular

- Moment.js [6]

Biblioteca de fechas de JavaScript para analizar, validar, manipular y formatear fechas.

- NGX Cookie Service [7]

Servicio Angular para leer, configurar y eliminar las cookies del navegador.

- RxJS [8]

Librería JavaScript de programación reactiva que facilita la composición de código asíncrono basado en secuencias de observables.

- Express.js [9]

Entorno de trabajo para aplicaciones web de Node.js de código abierto. Se utiliza para desarrollar aplicaciones web y APIs

- dotenv [10]

Dependencia de node.js para cargar variables del entorno.

- jsonwebtoken [11]

Implementación de JWT.

- Lodash [12]

Librería de utilidades JavaScript.

- mysql2 [13]

Ciente MySQL para node.js.

- TypeORM [14]

ORM para node.js que proporciona funcionalidades para utilizar MySQL.

- Git [15]

Control de versiones compatible con Android Studio.

- Fluid UI [16]

Herramienta para diseñar un prototipo funcional de la app.

- Notion [17]

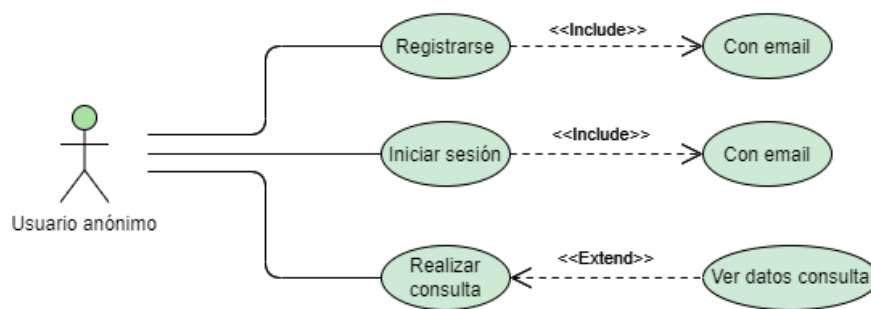
Gestor de proyectos.

- Visual Paradigm Online [18]

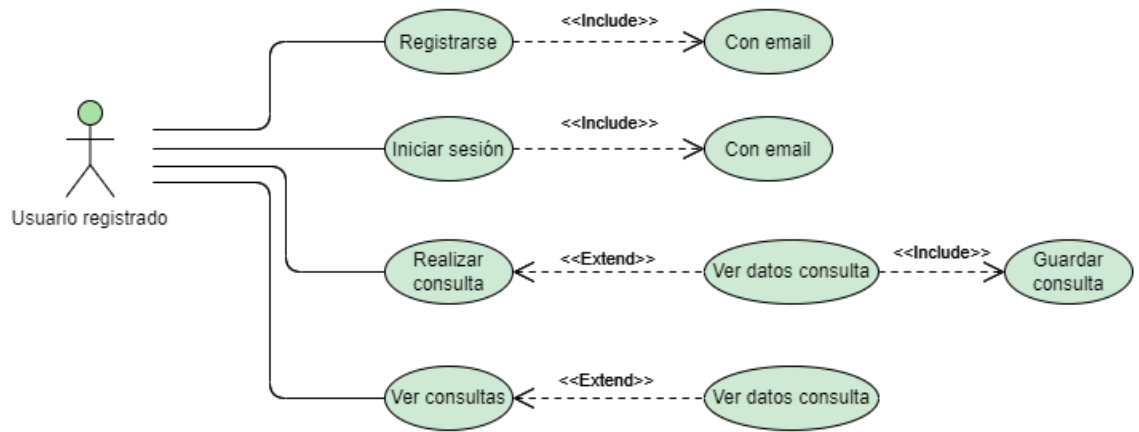
Herramienta online para la creación de diagramas.

2.6 Casos de Uso

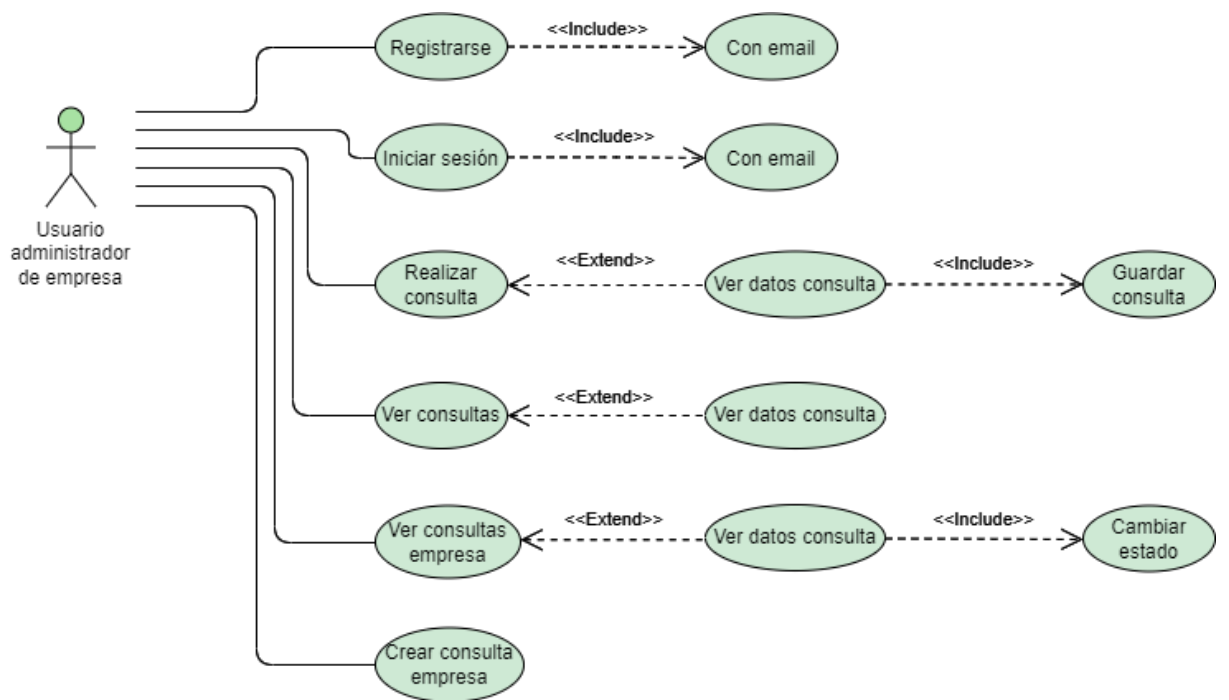
Los casos de uso permiten identificar el comportamiento de la aplicación y sus funcionalidades. Se han identificado tres actores: Usuario anónimo, Usuario registrado y Usuario administrador de una empresa.



Caso de uso de usuario anónimo



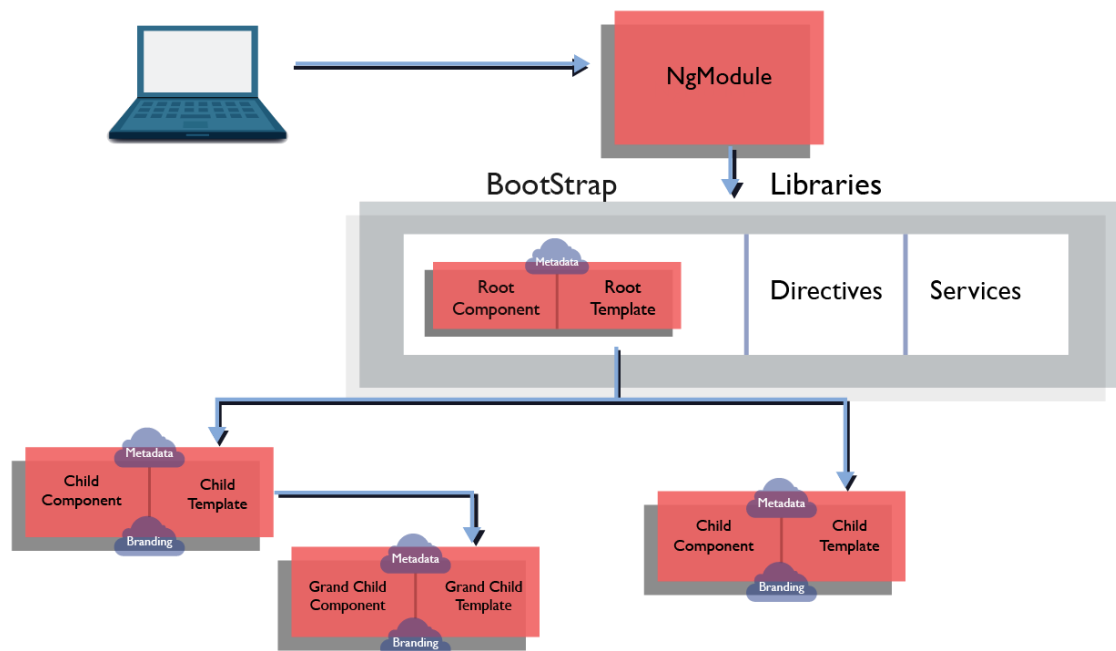
Caso de uso de usuario registrado



Caso de uso de usuario administrador

3.2 Arquitectura

3.2.1 Frontend



Arquitectura de Angular

 La arquitectura de Angular se basa principalmente en:

- **Módulos**

Un módulo declara un contexto de compilación para un conjunto de componentes para formar unidades funcionales.

Cada aplicación generada con Angular cuenta con un *root module*, el cual provee el mecanismo de arranque que inicia nuestra aplicación. Una aplicación generalmente contiene varios módulos funcionales.

Como en Javascript (y en muchos lenguajes con programación funcional), un módulo puede importar funcionalidades de otros módulos, y exportar sus propias funcionalidades.

- **Componentes**

Al igual que el *root module*, existe el *root component* que conecta una jerarquía de componentes con el DOM. Cada componente define una clase que contiene la lógica y está vinculado a un *template* (HTML).

- **Templates, directivas y data binding**

Un template es una mezcla de HTML con *Angular markup* (tags personalizados de Angular). Las directivas de un *template* proveen lógica de programación, y hacen un *data binding* (enlazan lógica) con las vistas.

Hay dos tipos de *data binding*:

- *Event binding* o enlace de eventos, que responden a la interacción del usuario al modificar algún input en la aplicación, actualizando los datos.
- *Property binding* o enlace de propiedades, que permite agregar valores modificados desde nuestro componente, a nuestro HTML.

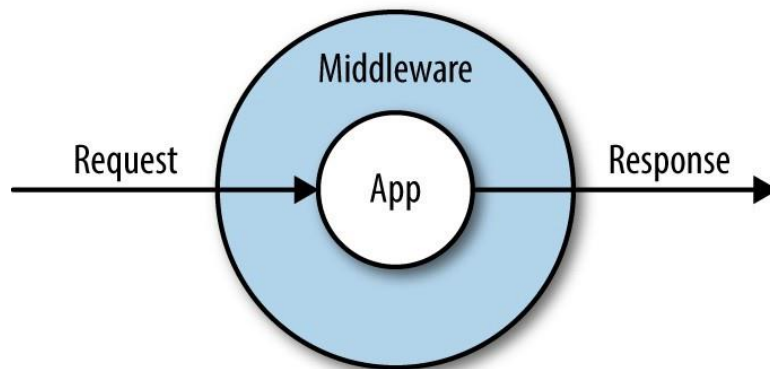
Antes de que se muestre una vista, Angular evalúa las directivas y resuelve la sintaxis del *data binding* en el *template* para modificar los elementos HTML y el DOM, según los datos y la lógica de nuestra aplicación. Angular cuenta con *two-way data binding*, que significa que los cambios en el DOM también se reflejan en nuestro componente.

- **Servicios e Inyección de Dependencias**

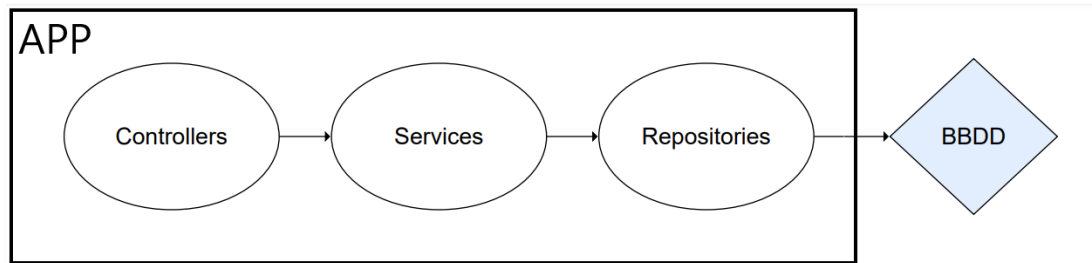
Toda la lógica que no está asociada directamente a una vista y que se quiere utilizar en diferentes partes de tu aplicación y entre diferentes componentes, puede ser escrita en un servicio. Tal como un componente, los servicios son exportados como clases. Los servicios cuentan con el decorator *@Injectable()*, el cual permite que los servicios sean inyectados en componentes como dependencias o a nivel de aplicación.

Dependency *injection* o Inyección de Dependencias permite manejar las clases de tus componentes de forma ligera y eficiente.

3.2.2 Backend



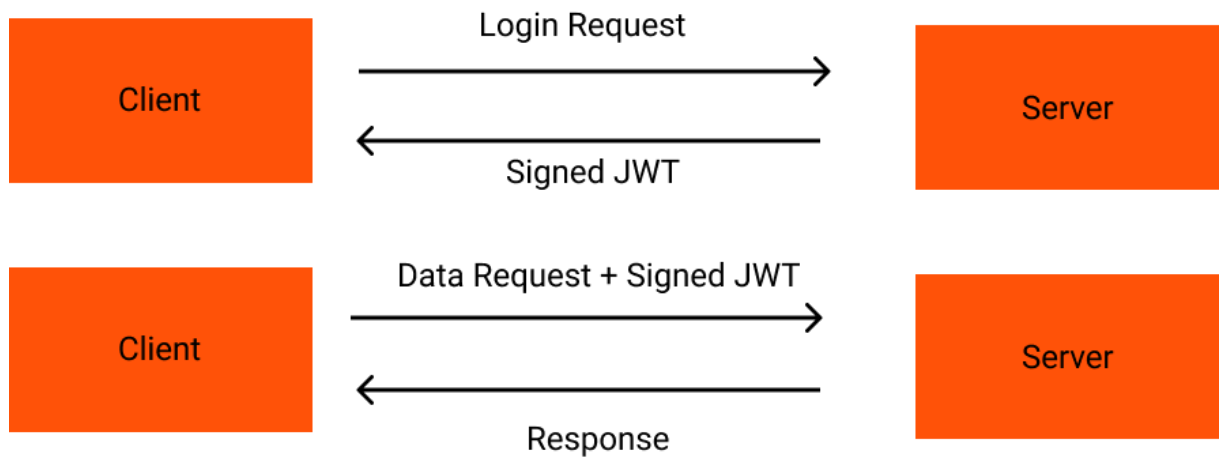
Arquitectura aplicación Node Express.JS



Arquitectura de capas App Express.JS

Para el backend se ha seguido una arquitectura de capas. La idea es utilizar el principio *de separación de conceptos*.

Los middlewares se pueden utilizar en cualquier punto de la aplicación, por ejemplo, tenemos de autenticar al usuario y logs.



Ejemplo flujo con middleware de autenticación con token JWT

En el controlador tenemos la definición e implementación de las rutas de Express.js. En el servicio está toda la lógica de negocio. Y, por último, el repositorio, que es el enlace con la base de datos.

Es decir, a la hora de recibir una petición desde la nube, ésta pasa por N número de middlewares, después, se recibe en el controlador correspondiente y éste pasa los parámetros recibidos al servicio, el cual realiza distintas comprobaciones y ejecuta la lógica de negocio. Por último, el servicio utiliza los repositorios para conectar con la base de datos.

3.3 Modelo de datos

Se ha implementado la base de datos con MySQL porque es rápida, ampliamente utilizada, con un mucho soporte técnico y alta seguridad.

En el backend, se ha utilizado TypeORM [14] como librería para explotar la base de datos.

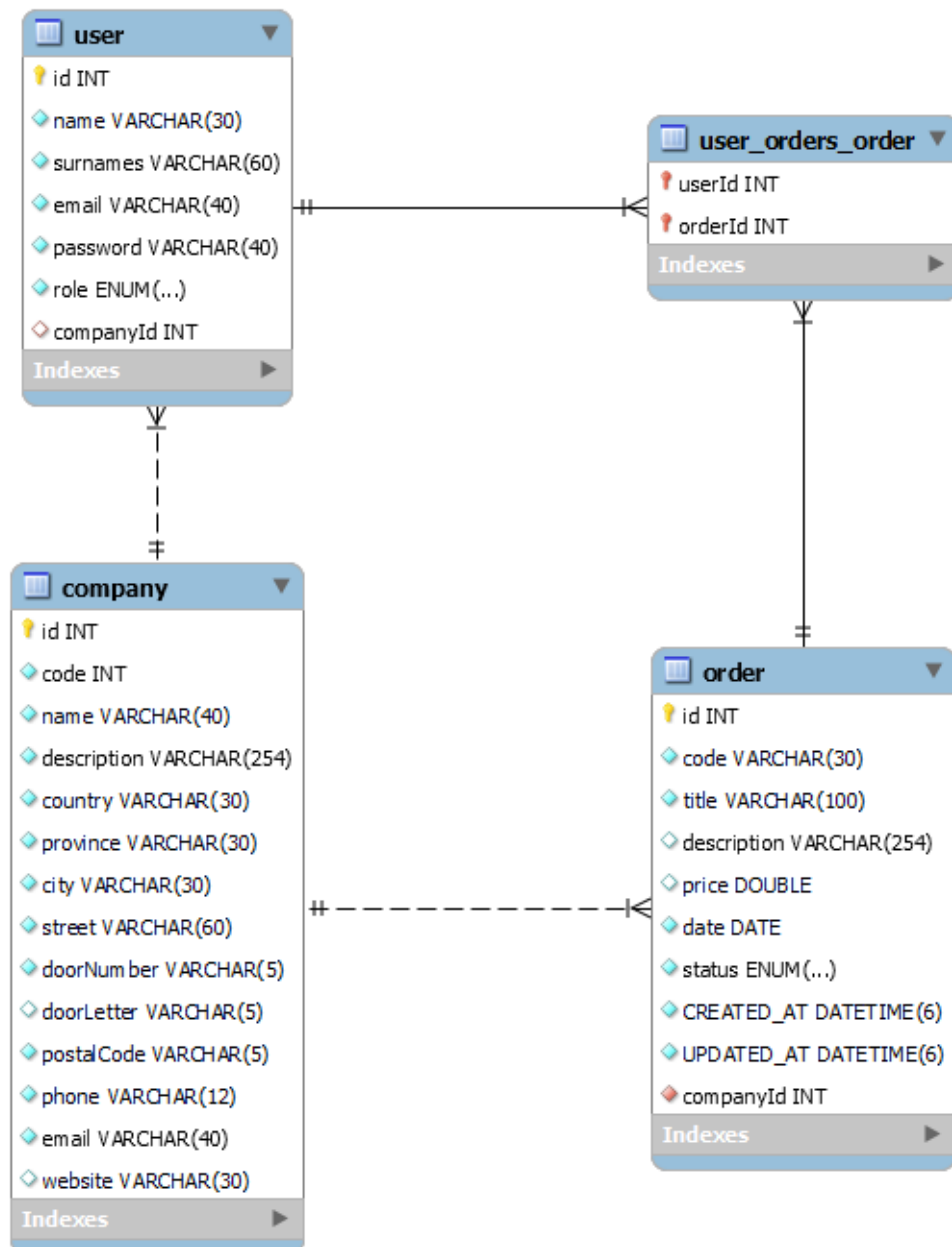


Diagrama E/R

Tenemos tres tablas principales y una auxiliar:

- **Company:**

Contiene toda la información de la empresa.

- **User:**

Contiene toda la información del usuario, y si es administrador de una empresa, su relación con ésta.

- **Order:**

Contiene toda la información del pedido, además de la empresa a la que pertenece.

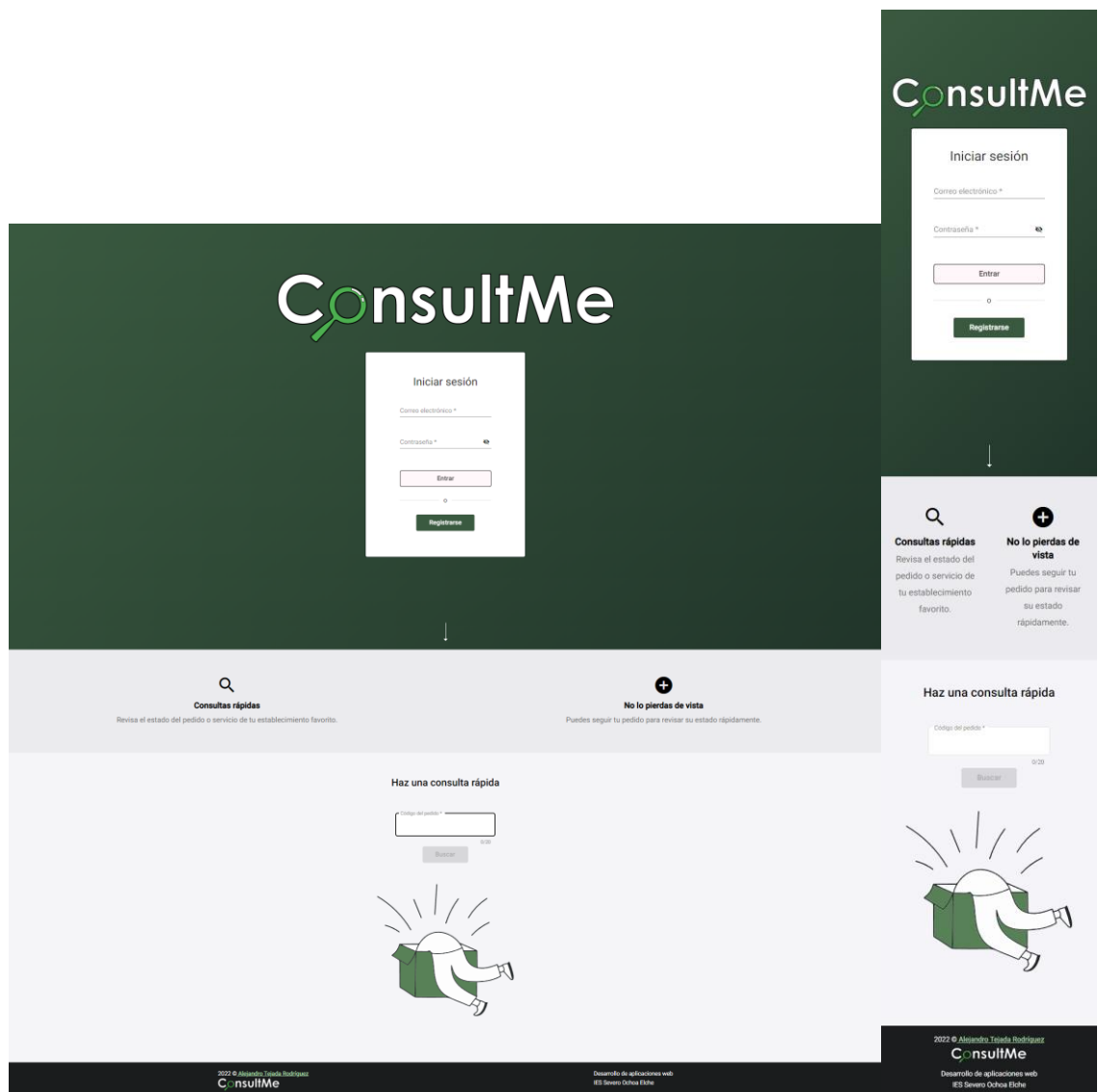
- **User_orders_order:**

Contiene la relación de los usuarios con los pedidos.

3.4 Vistas

A continuación, se mostrarán unas capturas finales de la aplicación, de escritorio y móvil.

3.4.1 Inicio



Página de inicio usuario anónimo



Página de inicio usuario existente

3.4.2 Usuario registrado - Mis consultas, realizar y guardar

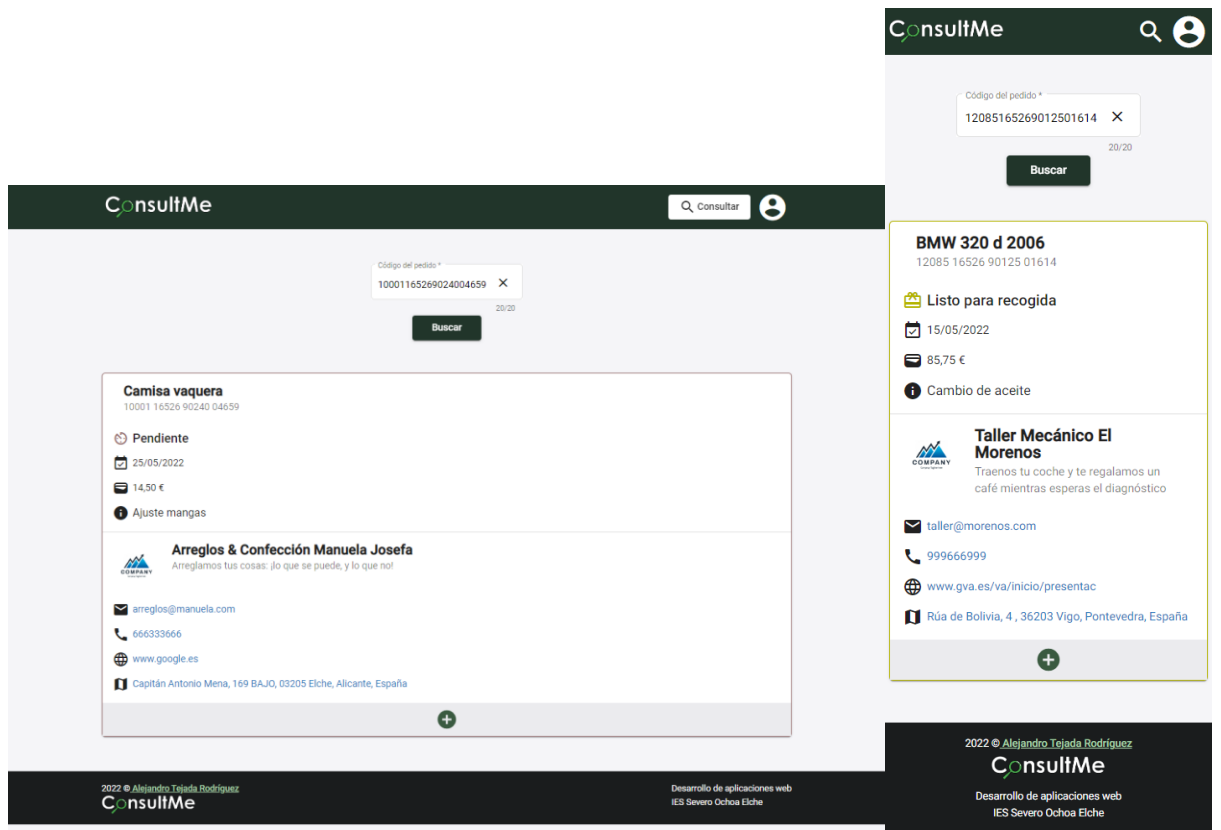
The screenshot displays the 'ConsultMe' web application interface. The main area shows a list of orders with details such as product name, code, status, date, and price. The sidebar on the right provides filters for 'Código', 'Estado', and 'Fecha', along with an 'Ordenar por' dropdown set to 'Fecha prevista de entrega'. The footer includes copyright information for 2022 by Alejandro Tejada Rodríguez and development credits to IES Severo Ochoa Elche.

Código	Estado	Fecha	Producto	Estado	Fecha	Precio
12085 16526 90125 01614	Todos (excepto Entrega...)	Todas	BMW 320 d 2006	Listo para recogida	15/05/2022	85,75 €
10001 16526 90240 00202			Bajo pantalón vaquero 2	Pendiente	26/05/2022	9,99 €
50133 10056 22565 11071			Samsung S9 Pantalla	Pendiente	15/06/2022	183,00 €

Consultas (pedidos) guardados por el usuario

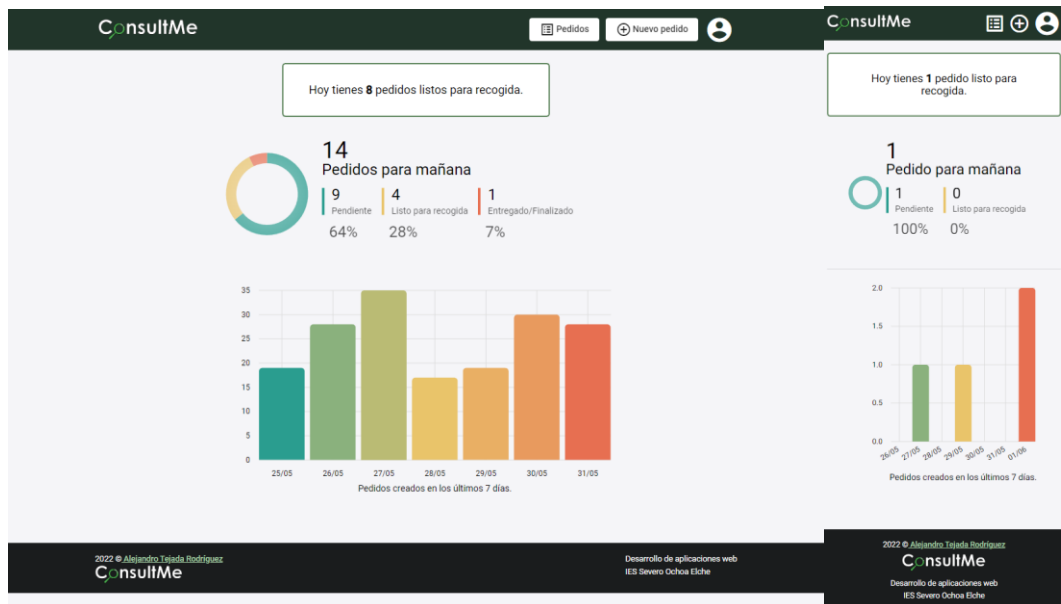
This screenshot shows the search functionality of the 'ConsultMe' application. It features a search bar labeled 'Código del pedido *' with a 'Buscar' button. Below the search bar is a cartoon illustration of a person emerging from a box. The footer contains the same copyright and development information as the previous screenshot.

Realizar consulta (buscar pedido)

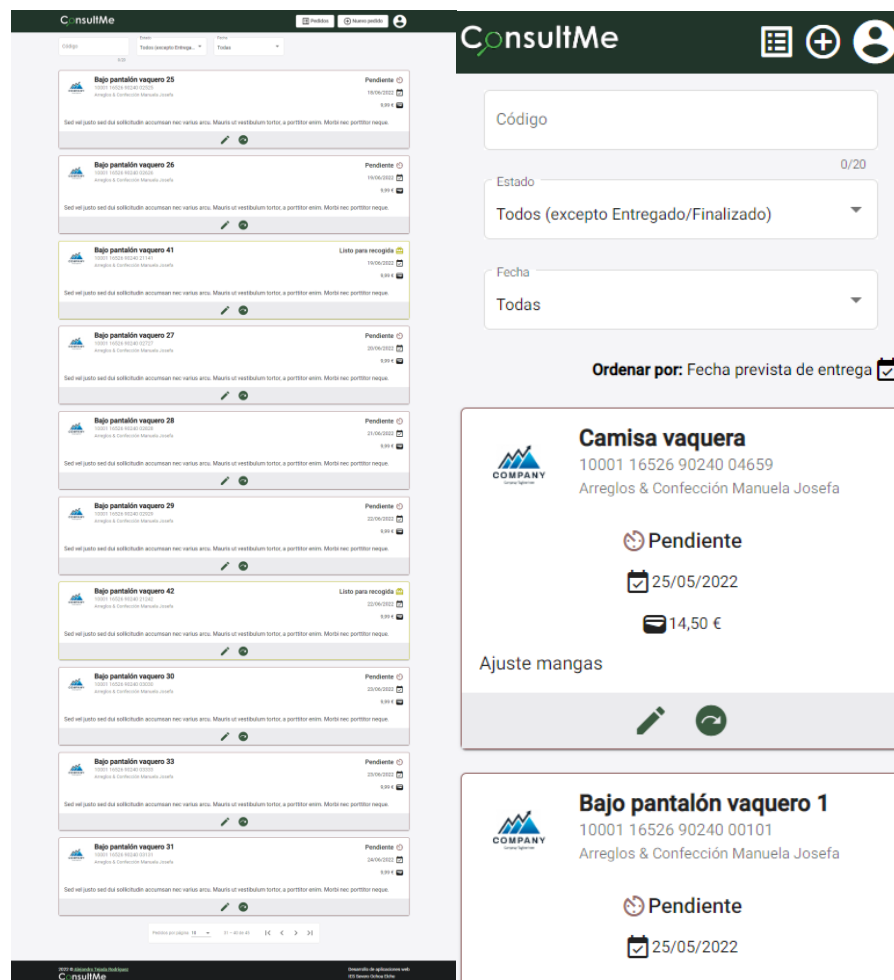


Realizar consulta (buscar pedido) – resultado satisfactorio

3.4.3 Empresa – Estadísticas, mis consultas, creación y modificación



Estadísticas



Consultas (pedidos) empresa

ConsultMe

+

Código del pedido

10001 16540 95328 26866

Título *

0/30

Descripción

0/254

Estado *

Pendiente

Fecha prevista de entrega *

DD/MM/YYYY

Cantidad a abonar

€

Crear

2022 © Alejandro Tejada Rodríguez

ConsultMe

Desarrollo de aplicaciones web

IES Severo Ochoa Elche

Crear nuevo pedido

ConsultMe

+

Código del pedido

10001 16526 90240 04659

Título *

Camisa vaquera

14/30

Descripción

Ajuste mangas

13/254

Estado *

Pendiente

Fecha prevista de entrega *

25/5/2022

DD/MM/YYYY

Cantidad a abonar

14.5 €

Actualizar

2022 © Alejandro Tejada Rodríguez

ConsultMe

Desarrollo de aplicaciones web

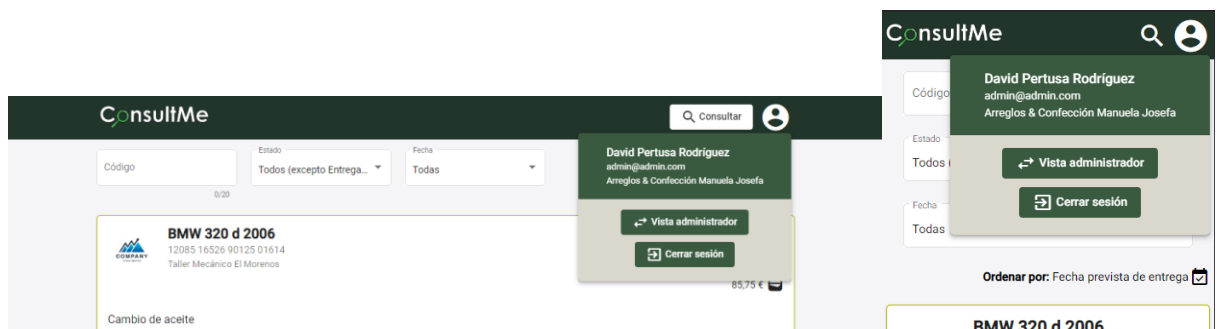
IES Severo Ochoa Elche

Modificar pedido

3.4.4 Perfil



Perfil modo administrador



Perfil modo usuario registrado

4. IMPLEMENTACIÓN

4.1 Configuración del entorno de desarrollo

Para el desarrollo se ha empleado un equipo con las siguientes características:

- Sobremesa AMD Ryzen 5 con 16GB de RAM.

Lo primero fue instalar Node y el entorno de MySQL, junto a Git para el control de versiones (*Adobe Illustrator y Adobe Photoshop para el diseño de los iconos*).

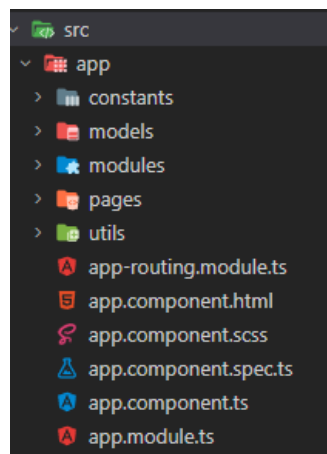
- Node 16.5.0
- MySQL 8.0.29.0
- Git 2.36.0

4.2 Desarrollo

Una vez configurado el entorno de desarrollo se creó la aplicación Angular + SASS (SCSS) con el comando `ng new My_New_Project --style=scss`, y la aplicación node (Express.js), con `npm init`.

4.2.1 Frontend

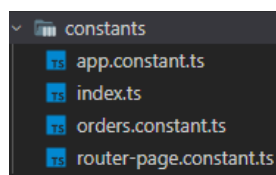
La aplicación Angular ha sido desarrollada siguiendo las recomendaciones del propio *framework* y las buenas prácticas. Además, se ha utilizado SASS como *framework* de CSS y se ha realizado un diseño responsivo.



src/app

La aplicación Angular se divide principalmente en:

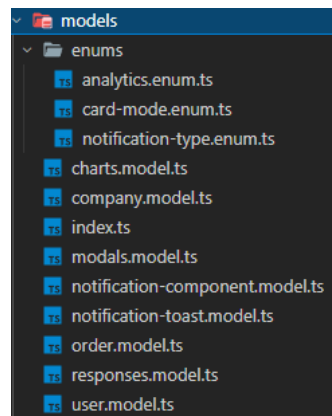
4.2.1.1 Constants



src/app/constants

Contiene las constantes de la aplicación.

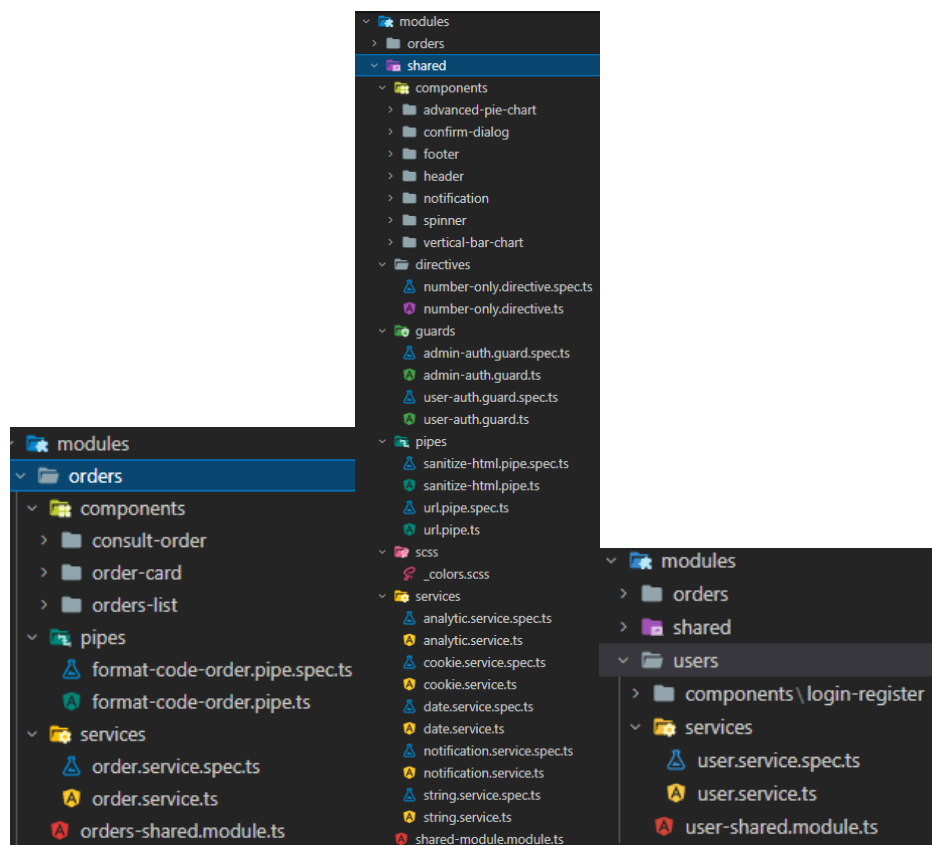
4.2.1.2 Models



src/app/models

Incluye los diferentes modelos de la aplicación.

4.2.1.3 Modules



src/app/modules/orders

src/app/modules/shared

src/app/modules/users

- **orders:** módulo funcional de los pedidos:

Components	Descripción
consult-order	Realizar una consulta.
order-card	Tarjeta de visualización de un pedido.
orders-list	Lista de pedidos.

Pipes	Descripción
format-code	Dar formato al código del pedido. (XXXXXX-XXXXXX-XXXXXX-XXXXXX-XXXXXX)

Services	Descripción
consult-order	Realizar las peticiones al backend relacionadas con los pedidos.

- **shared:** módulo funcional de elementos comunes a toda la aplicación.

Components	Descripción
advanced-pie-chart	Gráfico circular.
confirm-dialog	Pop-up de confirmación.
footer	Pie de página.
header	Cabecera de la página.
notification	Notificación informativa.
spinner	Indica que algo se está procesando, cargando.
verical-bar-chart	Gráfico de barras.

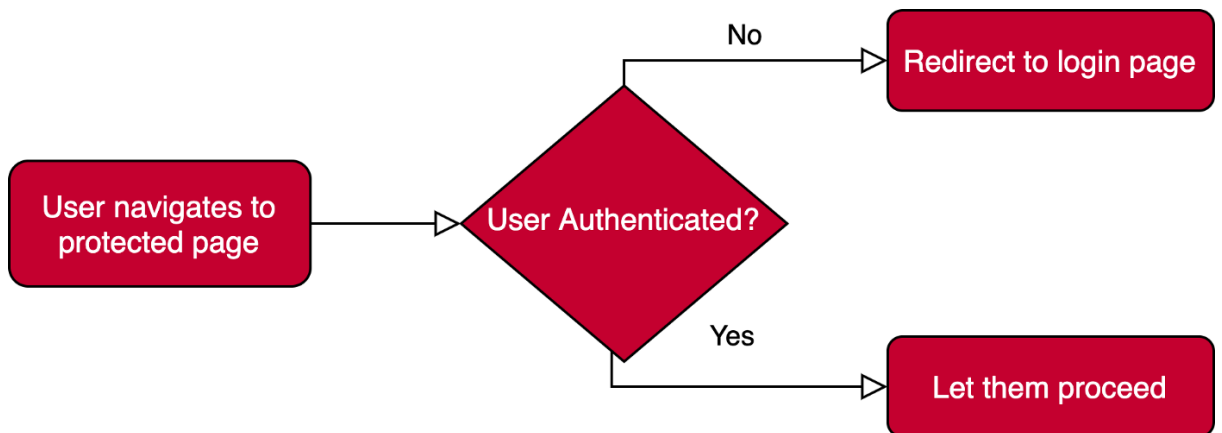
Directives	Descripción
number-only	Únicamente permite escribir números en un input.

Guards	Descripción
admin-auth	Limita las rutas de administrador a los usuarios con dicho rol.
user-auth	Limita las rutas de la aplicación a usuarios registrados.

Pipes	Descripción
sanitize-html	Utiliza DomSanitizer para prevenir brechas de seguridad en el HTML.
url	Construye correctamente las URLs.

SCSS	Descripción
colors	Colores de la aplicación.

Services	Descripción
analytic	Realizar las peticiones al backend relacionadas con las estadísticas.
cookie	Maneja las cookies del navegador.
date	Control de fechas.
notification	Ejecuta una notificación emergente.
string	Utilidades para strings.



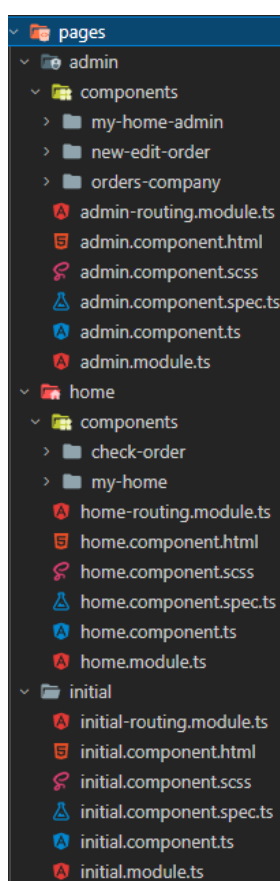
Ejemplo guard autenticación

- **users:** módulo funcional de los usuarios: componentes y servicios.

Components	Descripción
login-register	Inicio o registro del usuario.

Services	Descripción
user	Realizar las peticiones al backend relacionadas con los usuarios.

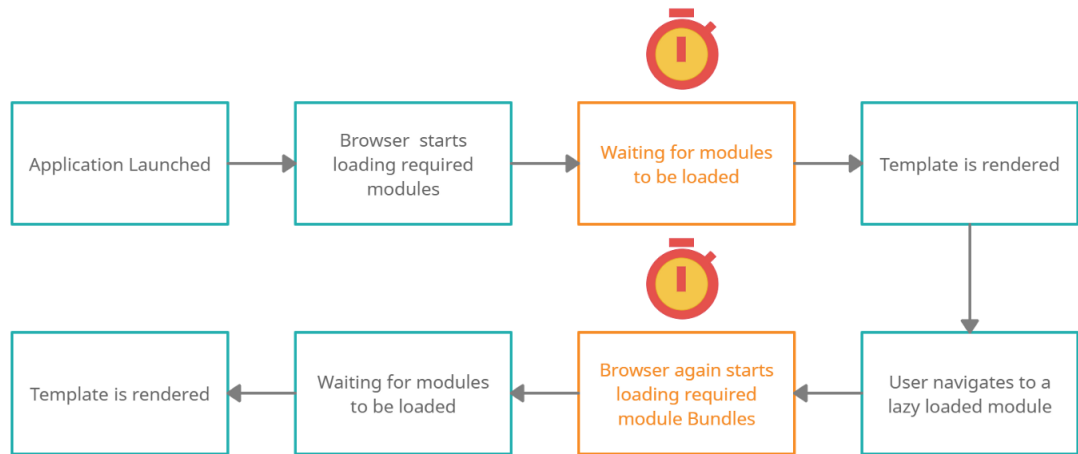
4.2.1.4 Pages



src/app/pages

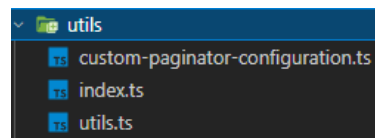
Contiene los módulos de las diferentes páginas de la aplicación.

Están configurados con *lazy loading* (carga diferida), lo cual permite que solo se carguen los módulos que van a ser utilizados. Es decir, si estamos en la vista de administrador, solo se cargará el módulo *admin.module.ts*.



Ejemplo lazy loading (carga diferida)

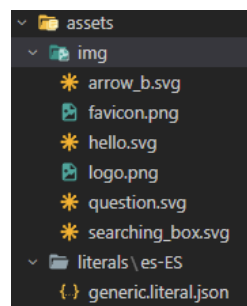
4.2.1.5 Utils



src/app/utils

Incluye funciones variadas útiles a nivel de aplicación.

4.2.1.6 Assets

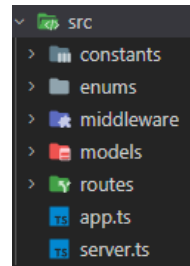


assets

Incluye las imágenes y todo el texto que se muestra al usuario final.

4.2.2 Backend

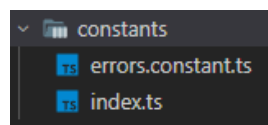
Como se ha comentado en el punto [3.2.2](#), se ha seguido la arquitectura de capas.



src

La aplicación se compone principalmente de:

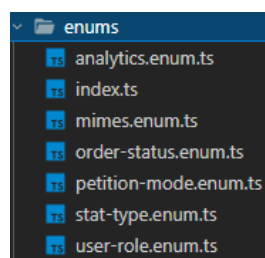
4.2.2.1 Constants



src/constants

Contiene las constantes de la aplicación.

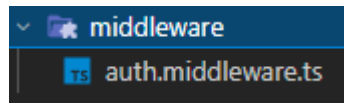
4.2.2.2 Enums



src/enums

Contiene los *enums* de la aplicación.

4.2.2.3 Middleware

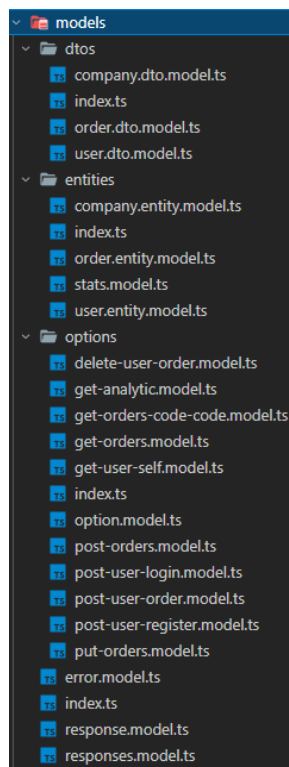


src/middleware

Contiene los *middlewares* de la aplicación.

- auth.middleware: verificar el token del usuario y los permisos para acceder a la ruta solicitada.

4.2.2.4 Models

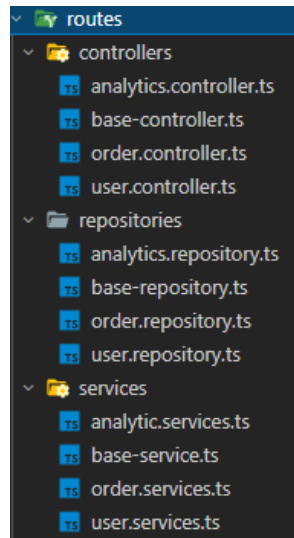


src/models

Contiene los modelos de la aplicación.

- dtos: modelos que son devueltos en la petición.
- entities: modelos de la base de datos.
- options: modelos de entrada de la petición.

4.2.2.5 Routes



src/routes

Contiene las rutas de la aplicación, con sus controladores, servicios y repositorios.

Se ha utilizado TypeORM [14] para conectar y manejar la base de datos.

4.3 Configuración servidor y despliegue

Para el servidor se ha optado por un VPS Linux de 1 vCore, 1GB RAM, 25GB SSD y tráfico ilimitado, en el cual se ha instalado Ubuntu 20.04.4 LTS.

El primer paso ha sido crear un nuevo usuario y proporcionarle permisos. A continuación, se ha instalado Node.js con NVM (gestor de versiones de Node.js) [19].

Se ha instalado el paquete *mysql-server* para la base de datos, el cual permite ejecutar una configuración inicial con el comando `mysql_secure_installation`. Para su control se ha utilizado MySQL Workbench [20].

Para subir el backend se ha utilizado Filezilla [21], al tener Node.js instalado en el servidor, para iniciarlo basta con ejecutar el comando `npm run start` en la carpeta contenedora. Se ha utilizado PM2 [22] para autoiniciar la aplicación Express.JS si el servidor se reinicia.

La aplicación Angular se expone con Nginx [23], el cual es un servidor web de código abierto. Como punto a destacar, se ha utilizado su configuración `try_files` para la compatibilidad con la SPA, ya que solo hay un archivo HTML estático y Angular es quien controla la navegación.

5. PRUEBAS

5.1 Pruebas en navegadores

Se han realizado multitud de pruebas en varios navegadores: Google Chrome, Microsoft Edge y Mozilla Firefox.

USUARIOS (FRONTEND)	RESULTADO
No se puede registrar un usuario sin correo.	OK
No se puede registrar un usuario sin contraseña.	OK
No se puede registrar un usuario sin nombre y apellidos.	OK
El correo electrónico ha de seguir el patrón usuario@servidor.	OK
La contraseña ha de tener como mínimo 6 caracteres.	OK
Se puede cambiar entre la vista de usuario y empresa.	OK
El token JWT [11] se obtiene y se guarda como cookie.	OK
Se permite <i>loguear</i> con el token JWT [11] para guardar la sesión.	OK

CONSULTAS (FRONTEND)	RESULTADO
Se puede realizar una consulta rápida desde la página de inicio, sin requerir un usuario.	OK
Se obtienen las consultas guardadas del usuario.	OK
Se permite el filtrado por código de la lista de consultas del usuario.	OK
Se permite el filtrado por estado de la lista de consultas del usuario.	OK
Se permite el filtrado por fecha de la lista de consultas del usuario.	OK
Se obtienen las consultas creadas por la empresa.	OK
Se permite el filtrado por código de la lista de consultas de la empresa.	OK
Se permite el filtrado por estado de la lista de consultas de la empresa.	OK

Se permite el filtrado por fecha de la lista de consultas de la empresa.	OK
Para crear un nuevo pedido (consulta) es necesario el título, estado y fecha prevista de entrega.	OK
Para modificar un pedido es necesario que tenga los campos requeridos y al menos uno modificado.	OK
Se puede actualizar el estado del pedido al siguiente desde la lista.	OK
En la página principal del administrador se informa del número de pedidos listos para recogida del día actual, permite redirigir a la lista.	OK
En la página principal del administrador se informa del estado de los pedidos para mañana.	OK
En la página principal del administrador se informa un historial del número de pedidos creados en los últimos siete días.	OK

BACKEND	RESULTADO
Se comprueban que los parámetros de entrada cumplan con los requeridos.	OK
Se comprueba el token JWT y si éste tiene permisos para acceder a la ruta solicitada.	OK
Se realiza un control de los errores para facilitar el origen del error.	OK

5.2 Evaluación

En la fase de diseño se tuvieron en cuenta las diversas opiniones de amigos y familiares tras realizar el prototipo y plasmar una idea más sólida. Se ha solicitado a las mismas personas que realicen distintas acciones sin ayuda alguna:

- Registrarse en la aplicación.
- Iniciar sesión.
- Realizar una consulta.
- Guardar una consulta.
- Crear una consulta como administrador.
- Modificar consultas existentes como administrador.

Una vez hecho esto, han respondido varias preguntas y todos han llegado a la misma conclusión: el funcionamiento es sencillo, directo; pero agradable, intuitivo. Cabe destacar que a todos les ha parecido muy útil.

6. CONCLUSIONES

El resultado de realizar este proyecto ha sido satisfactorio a nivel personal pues he ampliado mis conocimientos y asentado algunos que tenía olvidados.

Cosas como organizar adecuadamente la aplicación Angular, utilizar correctamente *lazy loading* y otras peculiaridades de Angular requiere de un estudio previo, como ocurre con el uso de la seguridad JWT en Express.JS, sus middlewares y rutas, y TypeORM [14].

La planificación ha sido bastante mayor de las 40h del proyecto. Realizar una aplicación web (frontend y backend) e intentar seguir unas líneas de calidad sin perder de vista la futura escalabilidad requiere mucho tiempo y esfuerzo.

Como se ha comentado en el punto [2.1 Objetivos específicos](#), los objetivos deseables son mejoras que tenía pensadas para la aplicación pero que por tiempo no es posible su implementación. He intentado pues, que sea una aplicación sencilla pero sólida con una visión verdaderamente útil.

7. REFERENCIAS

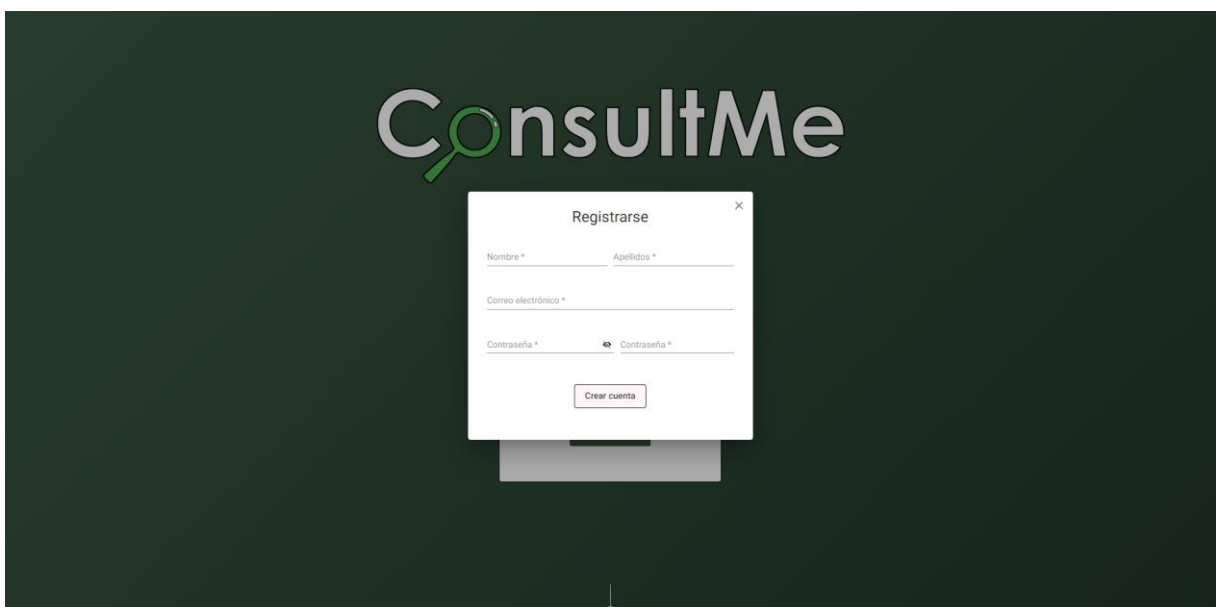
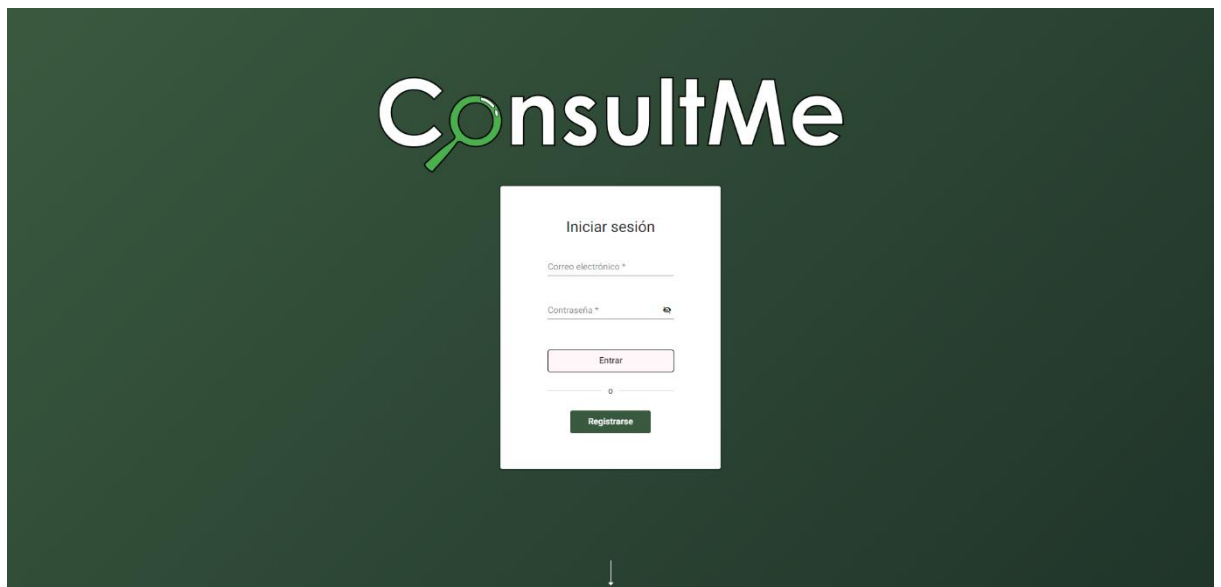
- [1] - [Visual Studio Code](#)
- [2] - [Node.js](#)
- [3] - [Angular](#)
- [4] - [dialog \(ngneat\)](#)
- [5] - [ngx-charts \(swimlane\)](#)
- [6] - [Moment.js](#)
- [7] - [NGX Cookie Service](#)
- [8] - [RxJS](#)
- [9] - [Express.js](#)
- [10] - [dotenv](#)
- [11] - [jsonwebtoken](#)
- [12] - [Lodash](#)
- [13] - [mysql2](#)
- [14] - [TypeORM](#)
- [15] - [Git](#)
- [16] - [Fluid UI](#)
- [17] - [Notion](#)
- [18] - [Visual Paradigm Online](#)
- [19] - [NVM](#)
- [20] - [MySQL Workbench](#)
- [21] - [Filezilla](#)
- [22] - [PM2](#)
- [23] - [Nginx](#)

8. ANEXOS

8.1 Manual de usuario

8.1.1 Iniciar sesión / registrarse

Para iniciar sesión o registrarnos se hará desde el menú principal.



8.1.2 Realizar consulta / guardar consulta

Para realizar una consulta (buscar un pedido) se debe pulsar en “Consultar” de la barra superior, tras insertar el código y pulsar en “Buscar”, saldrá la información del pedido. Para guardarlo, se pulsará en el icono “+”.

The image shows two screenshots of the ConsultMe web application. The top screenshot shows the search interface with a red arrow pointing to the 'Consultar' button. The bottom screenshot shows the results for a specific order.

Top Screenshot (Search Interface):

- Header: ConsultMe logo, search icon, and 'Consultar' button.
- Form fields: 'Código' (empty), 'Estado' (dropdown: 'Todos (excepto Entrega...)', 'Fecha' (dropdown: 'Todas').
- Result card: BMW 320 d 2006, 12085 16526 90125 01614, 'Listo para recogida' (15/05/2022).

Bottom Screenshot (Order Details):

- Header: ConsultMe logo, search icon, and 'Consultar' button.
- Form fields: 'Código del pedido *' (12085165269012501614), 'Buscar' button.
- Result card: BMW 320 d 2006, 12085 16526 90125 01614, 'Listo para recogida' (15/05/2022), 85,75 €, 'Cambio de aceite', 'Taller Mecánico El Morenos' (Traenos tu coche y te regalamos un café mientras esperas el diagnóstico), 'taller@morenos.com', 999666999, 'www.gva.es/va/inicio/presentac', 'Rúa de Bolivia, 4 , 36203 Vigo, Pontevedra, España'.

De la misma manera es posible realizar una consulta rápida desde la página principal, sin tener un usuario registrado.


Haz una consulta rápida


Código del pedido *
12085165269012501614 X


20/20


Buscar


BMW 320 d 2006
12085 16526 90125 01614


 **Listo para recogida**


 15/05/2022


 85,75 €


 Cambio de aceite

**Taller Mecánico El Morenos**
Traenos tu coche y te regalamos un café mientras esperas el diagnóstico

 taller@morenos.com

 999666999


 www.gva.es/va/inicio/presentac




 Rúa de Bolivia, 4 , 36203 Vigo, Pontevedra, España

¡Regístrate para seguir la consulta!


8.1.3 Borrar consulta guardada

Para borrar una consulta bastará con pulsar sobre el icono de la papelera en un pedido de la lista.

**BMW 320 d 2006**
12085 16526 90125 01614
Taller Mecánico El Morenos

Listo para recogida 
15/05/2022 
85,75 € 

Cambio de aceite



8.2 Manual de usuario administrador

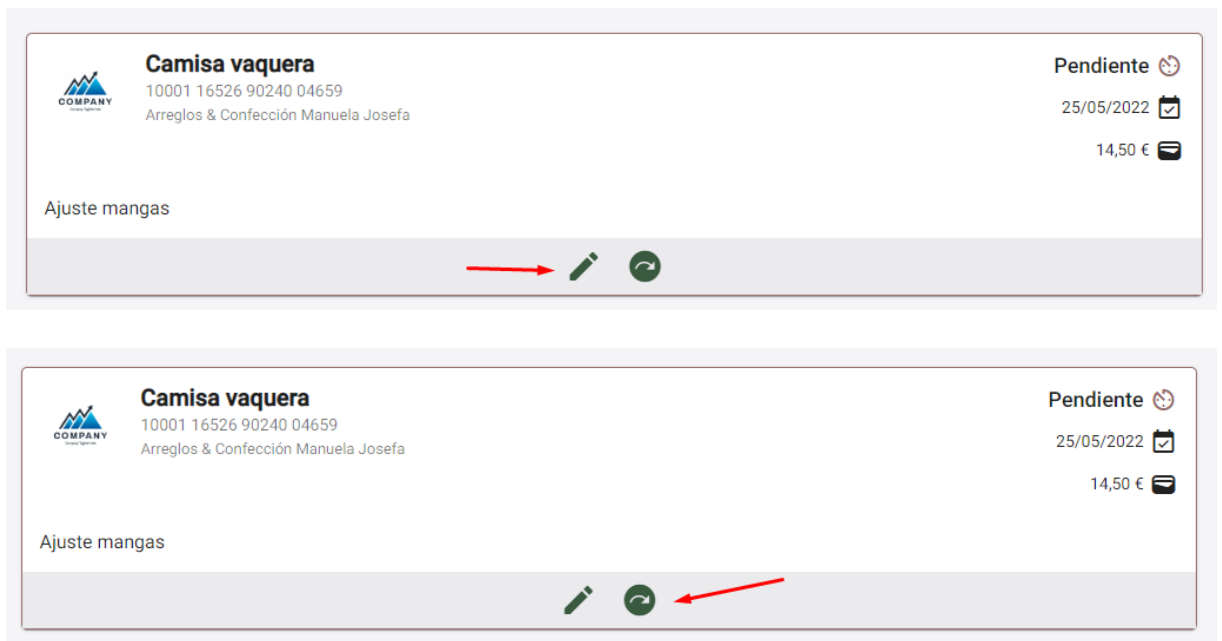
8.2.1 Crear consulta

Para crear una consulta de un negocio se hará desde la vista del modo administrador. Habrá que pulsar en el botón “Nuevo pedido”, tras rellenar los datos se podrá pulsar en “Crear”.

The screenshot shows the 'ConsultMe' administrator interface. At the top, there is a dark green header with the 'ConsultMe' logo on the left and three buttons on the right: 'Pedidos' (with a list icon), 'Nuevo pedido' (with a plus icon), and a user profile icon. A red arrow points to the 'Nuevo pedido' button. Below the header, the main content area has a light gray background. It starts with the text 'Código del pedido' followed by the large number '10001 16540 28102 35611'. Below this is a form with several fields: 'Titulo *' with the value 'Vestido de comunión azul claro' and a character count '30/30'; 'Descripción' with a large text area and a character count '0/254'; 'Estado *' with a dropdown menu showing 'Pendiente'; 'Fecha prevista de entrega *' with the date '29/6/2022' and a calendar icon; and 'Cantidad a abonar' with the value '465 €' and a currency icon. At the bottom center of the form is a dark green button labeled 'Crear', with a red arrow pointing to it.

8.2.2 Modificar consulta

Para modificar una consulta se debe pulsar en el icono del “lápiz” en un pedido de la lista. Si se quiere avanzarla directamente al siguiente estado, se podrá utilizar el botón de “flecha a la derecha”.



8.2.3 Estadísticas

Desde la página principal de la vista de administrador es posible ver diferentes estadísticas. Si se pulsa sobre el mensaje de “*Hoy tienes X pedidos listos para recogida*” se podrá acceder directamente a la lista de esos pedidos.

