



---

**Universidad de Valladolid**

# Escuela de Ingeniería Informática

## TRABAJO FIN DE GRADO

Grado en Ingeniería Informática  
Mención en Ingeniería de Software

# **Foodbook: red social para compartir experiencias con recetas de cocina y alimentos**

Alumno:  
**José María Lozano Olmedo**

Tutores:  
**César Pablo Gutiérrez Martínez**  
**Diego García Álvarez**



---

...



# Agradecimientos

A mi familia, por apoyarme durante estos cuatro años.

A mis tutores, en especial a Diego, por resolverme todas las dudas y ayudarme cuando era necesario. Sin vosotros esto no hubiese sido posible.

**Gracias a todos.**

*AGRADECIMIENTOS*

---

# Resumen

Este proyecto consiste en el desarrollo de una aplicación Web *fullstack* para publicar contenido relacionado con la alimentación que funcione como una red social. Este documento corresponde a la memoria del Trabajo de Fin de Grado del Grado en Ingeniería Informática en la mención de Ingeniería de *Software* de la Escuela de Ingeniería Informática de la Universidad de Valladolid.

Cada vez aparecen más redes sociales donde los usuarios suben contenido como pueden ser viajes, encuentros con amigos o cualquier otra cosa. También es habitual que los usuarios publiquen qué han comido o qué han cocinado. Sin embargo, estas redes sociales no están especialmente creadas para publicar recetas o información sobre alimentos.

Esto hace que la información sobre recetas o alimentos no esté bien organizada lo que al final provoca que sea de difícil acceso para los usuarios. Por otro lado, provoca que la sociedad no se pueda beneficiar de las experiencias de otros usuarios a la hora de cocinar, de recetas personales que se pasan de generación en generación que están escritas en papel o de los beneficios descubiertos de un alimento por cualquier persona para tratar un problema concreto.

El objetivo de este trabajo es crear una red social donde los usuarios puedan publicar las experiencias que tengan con una determinada receta, compartir recetas personales para que los demás usuarios las puedan ver y realizar publicaciones a partir de ellas. Además, los usuarios podrán publicar sus experiencias sobre diversos alimentos de manera que puedan ayudar a otros usuarios para su consumo. También los usuarios verán las publicaciones de la gente que siguen y podrán guardar sus recetas favoritas para que puedan acceder a ellas rápidamente.

El proyecto se ha desarrollado como una aplicación Web utilizando en la parte de *frontend* el *framework* Angular, con HTML, CSS, Typescript y la biblioteca Bootstrap para hacerlo adaptable a cualquier tamaño de pantalla. En la parte del *backend*, se ha utilizado NodeJS conectando con una base de datos MySQL. Se ha utilizado el marco de trabajo ágil Scrum para la monitorización y seguimiento del proyecto.

*RESUMEN*

---

# Abstract

This project consists on the development of a Fullstack Web application to post content related with food that works as a social network. This document corresponds to the documentation of the Final Degree Project of the degree in Computer Engineering in Software Engineering mention in the Computer Engineering college of University of Valladolid.

Every time appears more social networks where users upload content as travels, meetings with friends or anything else. It is also usual that users post what they have eaten or what they have cooked. However, this social networks are not specially created to post recipes or food information.

This makes that the information about recipes or foods is not well organized which in the end causes that it is difficult for users to access to it. On the other hand, causes that society cannot benefit of the experiencies of other users at cooking time, of personal recipes that are inherit from generation to generation that are written on paper or the benefits discovered of a food by anyone to deal with a specific problem.

The goal of this project is to create a social network where users can post the experiences that they have with a certain recipe, share personal recipes so other users can see them and publish from them. Moreover, users can post their experiences with various foods that can help other users for their consumption. In addition, users will see the posts of their followed users and they will be able to save their favourite recipes to access them quickly.

The project has been developed as a web application using in the frontend Angular framework with HTML, CSS, Typescript and Bootstrap library to make adaptative to any screen size. In the backend part, NodeJS has been used connected with a MySQL database. It has been used Scrum agile framework for monitoring and tracking the project.

*ABSTRACT*

---

# Índice general

<b>Agradecimientos</b>	<b>III</b>
<b>Resumen</b>	<b>V</b>
<b>Abstract</b>	<b>VII</b>
<b>Lista de figuras</b>	<b>XV</b>
<b>Lista de tablas</b>	<b>XIX</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Contexto . . . . .	1
1.2. Objetivos . . . . .	3
1.2.1. Objetivos de desarrollo . . . . .	3
1.2.2. Objetivos académicos . . . . .	3
1.3. Motivación . . . . .	4
1.4. Aplicaciones similares . . . . .	4
1.5. Estructura de la memoria . . . . .	6
<b>2. Planificación</b>	<b>7</b>
2.1. Introducción . . . . .	7
2.2. Scrum . . . . .	7

## ÍNDICE GENERAL

---

2.2.1. Definición y pilares . . . . .	7
2.2.2. Artefactos . . . . .	8
2.2.3. Roles . . . . .	9
2.2.4. Eventos . . . . .	9
2.3. Adaptación de Scrum al proyecto . . . . .	10
2.4. Análisis de riesgos . . . . .	12
2.5. Presupuesto . . . . .	26
2.5.1. Presupuesto simulado . . . . .	27
2.5.2. Presupuesto real . . . . .	28
2.6. <i>Product Backlog</i> inicial . . . . .	28
<b>3. Requisitos</b>	<b>33</b>
3.1. Introducción . . . . .	33
3.2. Requisitos Funcionales . . . . .	33
3.3. Requisitos No Funcionales . . . . .	35
3.4. Requisitos de Información . . . . .	35
3.5. Actores Principales . . . . .	36
3.5.1. Usuario . . . . .	36
3.5.2. Administrador . . . . .	37
3.6. Diagrama de Casos de Uso . . . . .	37
3.7. Descripción Casos de Uso . . . . .	37
<b>4. Análisis</b>	<b>61</b>
4.1. Introducción . . . . .	61
4.2. Modelo del Dominio . . . . .	61
4.2.1. Diagrama de Clases del Modelo del Dominio . . . . .	61
4.3. Modelo de Análisis . . . . .	61

## ÍNDICE GENERAL

---

4.3.1. Clases de Análisis . . . . .	63
4.4. Realización de Casos de Uso de Análisis . . . . .	63
4.4.1. Realización de Caso de Uso <i>Registrarse</i> . . . . .	63
4.4.2. Realización de Caso de Uso <i>Crear publicación</i> . . . . .	63
4.4.3. Realización de Caso de Uso <i>Crear receta</i> . . . . .	63
4.4.4. Realización de Caso de Uso <i>Buscar alimento</i> . . . . .	63
<b>5. Diseño</b>	<b>69</b>
5.1. Introducción . . . . .	69
5.2. Arquitectura Lógica del Sistema . . . . .	69
5.2.1. Arquitectura del Cliente . . . . .	70
5.2.2. Arquitectura del Servidor . . . . .	76
5.3. Patrones de Diseño . . . . .	80
5.3.1. Observador . . . . .	81
5.3.2. DAO/DTO . . . . .	81
5.3.3. <i>Singleton</i> . . . . .	82
5.4. Arquitectura Física del Sistema . . . . .	83
5.5. Diseño de la Base de Datos . . . . .	83
5.6. Diseño de la Interfaz Gráfica . . . . .	86
<b>6. Tecnologías utilizadas</b>	<b>105</b>
6.1. Introducción . . . . .	105
6.2. Base de datos . . . . .	105
6.2.1. Elección del tipo de base de datos . . . . .	105
6.2.2. MySQL . . . . .	107
6.3. <i>Frontend</i> . . . . .	107
6.3.1. Angular . . . . .	107

## ÍNDICE GENERAL

---

6.3.2. HTML5 . . . . .	109
6.3.3. CSS . . . . .	109
6.3.4. Bootstrap . . . . .	109
6.3.5. ChartJS . . . . .	110
6.3.6. Angular Material . . . . .	110
6.4. Backend . . . . .	110
6.4.1. NodeJS . . . . .	110
6.4.2. ExpressJS . . . . .	111
6.5. Desarrollo del proyecto . . . . .	111
6.5.1. Git . . . . .	111
6.5.2. Gitlab . . . . .	112
6.5.3. Astah . . . . .	112
6.5.4. Railway . . . . .	113
6.5.5. Cloudinary . . . . .	113
6.5.6. JWT . . . . .	113
6.5.7. Trello . . . . .	114
6.5.8. Visual Studio Code . . . . .	114
6.5.9. Balsamiq . . . . .	115
<b>7. Implementación y pruebas</b>	<b>117</b>
7.1. Introducción . . . . .	117
7.2. Estructura del código de <i>frontend</i> . . . . .	117
7.3. Estructura del código de <i>backend</i> . . . . .	120
7.4. Pruebas . . . . .	123
7.5. Problemas y dificultades encontradas . . . . .	124
7.6. Casos de Prueba . . . . .	125

## **ÍNDICE GENERAL**

---

<b>8. Seguimiento del proyecto</b>	<b>137</b>
8.1. Introducción . . . . .	137
8.2. Seguimiento del proyecto . . . . .	137
8.2.1. Sprint 1: 30/1/2023 - 16/2/2023 . . . . .	138
8.2.2. Sprint 2: 16/2/2023 - 5/3/2023 . . . . .	138
8.2.3. Sprint 3: 5/3/2023 - 22/3/2023 . . . . .	141
8.2.4. Sprint 4: 22/3/2023 - 8/4/2023 . . . . .	144
8.2.5. Sprint 5: 8/4/2023 - 25/4/2023 . . . . .	144
8.2.6. Sprint 6: 25/4/2023 - 12/5/2023 . . . . .	146
8.2.7. Sprint 7: 12/5/2023 - 29/5/2023 . . . . .	148
8.2.8. Sprint 8: 29/5/2023 - 15/6/2023 . . . . .	148
<b>9. Conclusiones</b>	<b>151</b>
9.1. Introducción . . . . .	151
9.2. Conclusiones . . . . .	151
9.3. Líneas de trabajo futuras . . . . .	153
<b>A. Manuales</b>	<b>155</b>
A.1. Manual de despliegue e instalación . . . . .	155
A.1.1. Requisitos para la instalación . . . . .	155
A.1.2. Instalación . . . . .	155
A.2. Manual de Usuario . . . . .	156
A.2.1. Registro e inicio de sesión . . . . .	156
A.2.2. Muro publicaciones, datos del Usuario y su edición . . . . .	156
A.2.3. Buscador de usuarios, detalles del Usuario y seguidos . . . . .	157
A.2.4. Buscador de recetas, detalles de la receta y favoritas . . . . .	157
A.2.5. Buscador de alimentos y detalles del alimento . . . . .	157

## *ÍNDICE GENERAL*

---

A.2.6. Crear receta y crear publicación . . . . .	157
A.2.7. Administrador . . . . .	158
A.2.8. Crear alimento y editar . . . . .	158
<b>B. Resumen de enlaces adicionales</b>	<b>167</b>
<b>Bibliografía</b>	<b>169</b>

# **Lista de Figuras**

3.1.	Primera parte del Diagrama de Casos de Uso para el actor Usuario. . . . .	38
3.2.	Segunda parte del Diagrama de Casos de Uso para el actor Usuario. . . . .	39
3.3.	Diagrama de Casos de Uso para el actor Administrador. . . . . . . . .	40
4.1.	Diagrama de Clases del Modelo de Dominio . . . . . . . . . . . . . . . . .	62
4.2.	Diagrama de Clases de Análisis .	64
4.3.	Diagrama de Secuencia del Caso de Uso <i>Registrarse</i> . . . . . . . . . . .	65
4.4.	Diagrama de Secuencia del Caso de Uso <i>Crear publicación</i> . . . . . . . . .	66
4.5.	Diagrama de Secuencia del Caso de Uso <i>Crear receta</i> . . . . . . . . . .	67
4.6.	Diagrama de Secuencia del Caso de Uso <i>Buscar alimento</i> . . . . . . . . .	68
5.1.	Arquitectura Lógica del Sistema. .	71
5.2.	Relaciones entre los componentes del patrón MVVM. . . . . . . . . . .	72
5.3.	Arquitectura del Cliente de la capa de presentación hecho en Angular correspondiente al <i>frontend</i> .	72
5.4.	<i>Decomposition&amp;Uses Style</i> del paquete <i>Componentes</i> . . . . . . . . . .	73
5.5.	<i>Decomposition&amp;Uses Style</i> del paquete <i>Servicios</i> . . . . . . . . . . .	74
5.6.	<i>Decomposition&amp;Uses Style</i> del paquete <i>ModelObjects</i> . . . . . . . . . .	75
5.7.	Arquitectura del Servidor de la capa lógica hecho en NodeJS correspondiente al <i>backend</i> .	76
5.8.	<i>Decomposition&amp;Uses Style</i> del paquete <i>Controlador</i> . . . . . . . . . .	77

## *LISTA DE FIGURAS*

---

5.9.	<i>Decomposition&amp;Uses Style</i> del paquete <i>ModelObjects</i> . . . . .	78
5.10.	<i>Decomposition&amp;Uses Style</i> del paquete <i>Middleware</i> . . . . .	79
5.11.	<i>Decomposition&amp;Uses Style</i> del paquete <i>DAO</i> . . . . .	80
5.12.	Patrón Observador . . . . .	81
5.13.	Patrón DAO/DTO . . . . .	82
5.14.	Patrón Singleton . . . . .	83
5.15.	Diagrama de Despliegue . . . . .	84
5.16.	Diseño Lógico de la Base de Datos . . . . .	85
5.17.	Boceto de la carta de una receta . . . . .	87
5.18.	Boceto de la carta de un Usuario . . . . .	88
5.19.	Boceto de la carta de una publicación . . . . .	88
5.20.	Boceto de la carta de un alimento . . . . .	89
5.21.	Boceto de la carta de un comentario . . . . .	89
5.22.	Boceto de la carta de un alimento incluido en una receta . . . . .	90
5.23.	Parte superior del boceto de la interfaz de descripción de la aplicación. . . . .	90
5.24.	Parte inferior del boceto de la interfaz de descripción de la aplicación. . . . .	91
5.25.	Boceto de la interfaz de registro de un Usuario. . . . .	91
5.26.	Boceto de la interfaz de identificación de un Usuario. . . . .	92
5.27.	Boceto de la interfaz de muro de publicaciones. . . . .	92
5.28.	Boceto de la interfaz de perfil del Usuario. . . . .	93
5.29.	Boceto de la interfaz de editar perfil del Usuario. . . . .	93
5.30.	Boceto de la interfaz de seguidores del Usuario. . . . .	94
5.31.	Boceto de la interfaz de seguidos del Usuario. . . . .	94
5.32.	Boceto de la interfaz de recetas favoritas. . . . .	95
5.33.	Boceto de la interfaz de detalles de la receta. . . . .	96
5.34.	Boceto de la interfaz de detalles del Usuario. . . . .	97

## *LISTA DE FIGURAS*

---

5.35. Boceto de la interfaz de detalles de la publicación. . . . .	97
5.36. Boceto de la interfaz de detalles del alimento. . . . .	98
5.37. Boceto de la interfaz de crear receta. . . . .	99
5.38. Boceto de la interfaz de crear publicación. . . . .	100
5.39. Boceto de la interfaz de crear alimento. . . . .	101
5.40. Boceto de la interfaz de buscador de recetas. . . . .	101
5.41. Boceto de la interfaz de buscador de alimentos. . . . .	102
5.42. Boceto de la interfaz de buscador de usuarios. . . . .	102
5.43. Boceto de la interfaz de vista de Administrador. . . . .	103
7.1. Estructura de ficheros en la parte del <i>frontend</i> en Angular. . . . .	118
7.2. Estructura de ficheros en la carpeta <b>servicios</b> . . . . .	120
7.3. Estructura de los ficheros en la carpeta <b>componentes</b> . . . . .	121
7.4. Estructura de los ficheros en la parte del <i>backend</i> en NodeJS. . . . .	122
A.1. Pantalla descripcion-aplicacion. . . . .	158
A.2. Pantalla registro-usuario. . . . .	159
A.3. Pantalla identificacion-usuario. . . . .	159
A.4. Pantalla muro-publicaciones. . . . .	159
A.5. Pantalla perfil-usuario. . . . .	160
A.6. Pantalla editar-usuario. . . . .	160
A.7. Pantalla buscar-usuario. . . . .	160
A.8. Pantalla detalles-usuario. . . . .	161
A.9. Pantalla seguidos-usuario. . . . .	161
A.10. Pantalla seguidores-usuario. . . . .	161
A.11. Pantalla buscar-receta. . . . .	162
A.12. Pantalla detalles-receta. . . . .	162

## *LISTA DE FIGURAS*

---

A.13.Pantalla favoritas-recetas. . . . .	162
A.14.Pantalla buscar-alimento. . . . .	163
A.15.Pantalla detalles-alimento. . . . .	163
A.16.Pantalla crear-receta. . . . .	163
A.17.Pantalla crear-publicacion. . . . .	164
A.18.Pantalla vista-admin en la pestaña usuarios. . . . .	164
A.19.Pantalla vista-admin en la pestaña alimentos. . . . .	164
A.20.Pantalla vista-admin en la pestaña recetas. . . . .	165
A.21.Pantalla vista-admin en la pestaña publicaciones. . . . .	165
A.22.Pantalla crear-alimento. . . . .	165
A.23.Pantalla editar-alimento. . . . .	166

# **Lista de Tablas**

2.1.	Planificación de los <i>sprints</i> . . . . .	12
2.2.	R-01. Falta de cualificación con las tecnologías utilizadas. . . . .	14
2.3.	R-02. Estimación de los plazos de tiempo incorrecta. . . . .	15
2.4.	R-03. Desarrollo de la interfaz de la aplicación incorrecta. . . . .	16
2.5.	R-04. Fallo en el equipo de trabajo donde se realiza el proyecto. . . . .	17
2.6.	R-05. Cambios tardíos y muy importantes de los requisitos de la aplicación Web. . . . .	18
2.7.	R-06. Enfermedad del alumno. . . . .	19
2.8.	R-07. Actualización con grandes cambios de las tecnologías utilizadas. . . . .	20
2.9.	R-08. Pérdida de datos del proyecto. . . . .	21
2.10.	R-09. Problemas con la conexión a internet. . . . .	22
2.11.	R-10. Falta de tiempo para trabajar en el proyecto. . . . .	23
2.12.	R-11. Problemas para contactar con los tutores o con otros profesores. . . . .	24
2.13.	R-12. Aparición de <i>bugs</i> con complicada solución. . . . .	25
2.14.	Matriz de Riesgos. . . . .	26
2.15.	Situación de los riesgos del proyecto en la Matriz de Riesgos. . . . .	26
2.16.	Presupuesto simulado . . . . .	28
2.17.	Presupuesto real . . . . .	28
2.18.	<i>Product Backlog</i> inicial. . . . .	30
2.19.	<i>Product Backlog</i> inicial. . . . .	31

## *LISTA DE TABLAS*

---

3.1. Descripción del Caso de Uso <i>Iniciar sesión.</i> . . . . .	41
3.2. Descripción del Caso de Uso <i>Registrarse.</i> . . . . .	42
3.3. Descripción del Caso de Uso <i>Cerrar sesión.</i> . . . . .	43
3.4. Descripción del Caso de Uso <i>Editar información del Usuario.</i> . . . . .	44
3.5. Descripción del Caso de Uso <i>Ver publicaciones usuarios seguidos.</i> . . . . .	45
3.6. Descripción del Caso de Uso <i>Crear receta.</i> . . . . .	46
3.7. Descripción del Caso de Uso <i>Buscar alimento.</i> . . . . .	47
3.8. Descripción del Caso de Uso <i>Buscar receta.</i> . . . . .	48
3.9. Descripción del Caso de Uso <i>Crear publicación.</i> . . . . .	49
3.10. Descripción del Caso de Uso <i>Buscar usuario.</i> . . . . .	50
3.11. Descripción del Caso de Uso <i>Añadir receta a favoritas.</i> . . . . .	51
3.12. Descripción del Caso de Uso <i>Consultar recetas favoritas.</i> . . . . .	51
3.13. Descripción del Caso de Uso <i>Borrar receta de favoritas.</i> . . . . .	52
3.14. Descripción del Caso de Uso <i>Consultar usuarios seguidos.</i> . . . . .	52
3.15. Descripción del Caso de Uso <i>Dejar de seguir Usuario.</i> . . . . .	53
3.16. Descripción del Caso de Uso <i>Seguir Usuario.</i> . . . . .	53
3.17. Descripción del Caso de Uso <i>Añadir comentario a publicación.</i> . . . . .	54
3.18. Descripción del Caso de Uso <i>Eliminar Usuario.</i> . . . . .	55
3.19. Descripción del Caso de Uso <i>Eliminar publicación.</i> . . . . .	56
3.20. Descripción del Caso de Uso <i>Eliminar alimento.</i> . . . . .	57
3.21. Descripción del Caso de Uso <i>Editar información de alimento.</i> . . . . .	58
3.22. Descripción del Caso de Uso <i>Eliminar receta.</i> . . . . .	59
3.23. Descripción del Caso de Uso <i>Crear alimento.</i> . . . . .	60
7.1. CP-01. Iniciar sesión. . . . .	125
7.2. CP-02. Registrarse. . . . .	126
7.3. CP-03. Cerrar sesión. . . . .	126

## *LISTA DE TABLAS*

---

7.4. CP-04. Editar información del Usuario. . . . .	127
7.5. CP-05. Crear receta. . . . .	128
7.6. CP-06. Ver publicaciones usuarios seguidos. . . . .	129
7.7. CP-07. Buscar alimento. . . . .	129
7.8. CP-08. Buscar receta. . . . .	130
7.9. CP-09. Crear publicación. . . . .	130
7.10. CP-10. Buscar usuario. . . . .	131
7.11. CP-11. Consultar recetas favoritas. . . . .	131
7.12. CP-12. Borrar receta de favoritas. . . . .	131
7.13. CP-13. Añadir receta a favoritas. . . . .	132
7.14. CP-14. Consultar usuarios seguidos. . . . .	132
7.15. CP-15. Dejar de seguir Usuario. . . . .	132
7.16. CP-16. Seguir Usuario. . . . .	133
7.17. CP-17. Añadir comentario a publicación. . . . .	133
7.18. CP-18. Eliminar Usuario. . . . .	133
7.19. CP-19. Eliminar publicación. . . . .	134
7.20. CP-20. Eliminar alimento. . . . .	134
7.21. CP-21. Eliminar receta. . . . .	134
7.22. CP-22. Crear alimento. . . . .	135
7.23. CP-23. Editar información de alimento. . . . .	136
 8.1. Tareas de <i>sprint 1</i> . . . . .	139
8.2. Tareas de <i>sprint 1</i> . . . . .	140
8.3. Tareas de <i>sprint 2</i> . . . . .	142
8.4. Tareas de <i>sprint 3</i> . . . . .	143
8.5. Tareas de <i>sprint 4</i> . . . . .	145
8.6. Tareas de <i>sprint 4</i> . . . . .	146

## *LISTA DE TABLAS*

---

8.7. Tareas de <i>sprint</i> 5 . . . . .	147
8.8. Tareas de <i>sprint</i> 6 . . . . .	149
8.9. Tareas de <i>sprint</i> 7 . . . . .	150
8.10. Tareas de <i>sprint</i> 8 . . . . .	150

*LISTA DE TABLAS*

---



# Capítulo 1

## Introducción

### 1.1. Contexto

Los usuarios de las redes sociales [1] crecieron 227 millones en el año 2021 alcanzando en julio de 2022, 4.700 millones de usuarios, lo que representa alrededor del 59 % de la población mundial. Sin embargo, se espera que el crecimiento en los próximos años no sea tan elevado, debido a que ya la mayoría de la gente del mundo está conectada y ya han incorporado las redes sociales como una parte más de su vida. Por lo tanto, la clave ahora es conocer cómo se están utilizando las redes sociales y cómo aprovechar las oportunidades que brindan debido a la constante evolución de las tecnologías.

Un Usuario típico de redes sociales pasa dos horas y media en ellas lo que equivale a 75 horas al mes, que es más de tres días completos. Normalmente usa más de siete redes sociales y los *smartphones* son los principales dispositivos que utiliza, por lo tanto, es importante que estén optimizadas para que se puedan ver en cualquier tamaño de pantalla.

Cada red social ofrece distintos formatos para publicar su contenido, y la clave de las redes sociales es cómo muestran su contenido, en general entornos donde se publican vídeos y fotos acompañadas de textos.

La red social más conocida es Facebook. Con casi 3.000 millones de usuarios [2] es la líder indiscutible y es la preferida entre las personas adultas. La segunda red social con más usuarios es Youtube con 2500 millones de usuarios, debido a que es la más conocida para publicar todo tipo de vídeos. En tercer lugar, empatada con Whatsapp, se encuentra Instagram con 2.000 millones de usuarios. En estas redes sociales se puede publicar contenido de texto junto a imágenes o vídeos. De estas redes sociales destaca el crecimiento de Instagram con 522 millones de usuarios más en el 2022, lo que significa un 35 % de incremento. Instagram es la red social preferida entre los jóvenes. Una de sus ventajas es la buena organización que tienen las publicaciones, acompañadas de una fácil interfaz lo que facilita la navegación y el rápido acceso a una enorme cantidad de contenido.

## 1.1. CONTEXTO

---

Cabe destacar el crecimiento de otras redes sociales en los últimos años, como es el caso de Tiktok. Esta red social en 2021 fue la que más creció cruzando la barrera de los 1.000 millones de usuarios con un crecimiento del 45 %. La pandemia del coronavirus que provocó el confinamiento de gran parte de la población hizo que esta red social creciese de manera meteórica, siendo los principales usuarios adolescentes [3].

Las redes sociales en muchos casos son utilizadas como fuente de noticias, principalmente entre las personas más jóvenes. Sin embargo, las principales razones por la que se utilizan las redes sociales son mantenerse en contacto con familiares y amigos, pasar el tiempo libre, leer noticias y buscar contenidos. En muchos casos, estas redes sociales sirven para vender productos, debido a que las publicaciones con fotos y vídeos sirven como un escaparate. Además, la capacidad de promocionar un producto por una persona importante que tiene muchos seguidores ayuda a que el producto sea más popular [1].

Las recetas en papel como los libros de recetas han dejado paso a las nuevas tecnologías, debido a las ventajas que ofrecen estas últimas. Predominan los *blogs* de recetas y los vídeos de Youtube, ya que aprovechando los beneficios de las tecnologías se pueden ver con mayor detalle la preparación y además se puede interactuar con otros usuarios.

La alimentación saludable es una tendencia en auge en los últimos años, que ha ido creciendo desde que llegó la pandemia. Algunos estudios revelan que el 80 % de los consumidores gastan más en alimentación saludable que antes de la pandemia y en 2020 la salud era la principal preocupación para la mitad de los españoles. La alimentación saludable tiene un importante componente digital debido a que el 56 % de los consumidores buscan información de los productos en Internet, destacando las redes sociales ya que se estima que tres de cada diez personas siguen a algún *influencer* de alimentación [4].

Las redes sociales influyen en los hábitos alimenticios de los usuarios, en muchos casos para mal, pero en otros para bien. Una publicación de un *influencer* con muchos seguidores puede influir sobre mucha gente, debido al desconocimiento sobre la materia o porque consideran al *influencer* como una autoridad. Esto puede provocar daños en la salud, sobre todo en los seguidores más jóvenes que no tienen conocimientos de la alimentación.

En relación con esto, aparece la expresión *comer con los ojos*, ya que la exposición continua a alimentos poco saludables aumenta el consumo de estos lo que provoca un aumento de peso. Esta exposición, en gran parte, ocurre al consultar redes sociales. Pero esto se podría aplicar también a la inversa, exponiendo a los usuarios a alimentos saludables.

Un Usuario que vea que su entorno virtual consume alimentos saludables como frutas y verduras, es probable que se vea motivado a aumentar el consumo de estas. Las redes sociales son una herramienta para impulsar la conducta alimentaria, además pueden influir sobre la Salud Pública debido a los contenidos que se publican y que tanta gente ve. Puesto que muchos jóvenes pasan mucho tiempo viendo las redes sociales, se podrían aprovechar estas herramientas para promover hábitos de alimentación saludable [5].

La aplicación Web desarrollada en este trabajo consistirá en una red social para publicar contenido relacionado con la alimentación. Pretende aprovechar todo lo que se ha comentado anteriormente. Al ser una red social, es posible que tenga una buena acogida de los usuarios, además solucionar las desventajas que tienen otras redes sociales a la hora de publicar

contenido relacionado con comidas y recetas. Además, la aplicación Web se podrá utilizar como un almacén donde los usuarios puedan guardar sus recetas. De esta manera, así todos los usuarios podrán verlas y prepararlas. La aplicación Web facilitará la tarea al Usuario a la hora de crear una receta. Por otro lado, la aplicación Web almacenará alimentos con la información más relevante sobre ellos de esta forma los usuarios podrán ver cuáles son las características de este alimento. Todo esto hará que la aplicación ayude a las personas a llevar una vida más saludable, facilitando el acceso a miles de recetas y alimentos y poder ver qué comen los demás puede incentivar a las personas a tener hábitos más saludables en cuanto a la alimentación.

## 1.2. Objetivos

### 1.2.1. Objetivos de desarrollo

El principal objetivo de desarrollo es crear una aplicación Web que pueda ser utilizada por cualquier Usuario por su facilidad de uso y que de verdad pueda tener un beneficio en la sociedad. Para ello, los principales objetivos de desarrollo para la aplicación Web son:

1. **Desarrollar una aplicación que permita a los usuarios compartir sus recetas y publicaciones.** Se deberá desarrollar una aplicación Web que permita a los usuarios compartir sus recetas y publicaciones. Además la aplicación Web contará con más funcionalidad y la aplicación Web debe ser fácil de utilizar.
2. **Seguir el proceso de desarrollo de *software* para crear una aplicación Web.** Se deberá aplicar el proceso para el desarrollo de *software* para crear la aplicación Web realizando todos los pasos para que la aplicación Web obtenida sea robusta y cumpla las necesidades del cliente.
3. **Realizar la documentación del proyecto para su mantenimiento y escalabilidad en el futuro.** Se deberá realizar la documentación del proyecto para facilitar su mantenimiento en el futuro y su escalabilidad. En la documentación se deben justificar todas las decisiones tomadas.

### 1.2.2. Objetivos académicos

En cuanto a los objetivos académicos, el objetivo más importante que se debe tener presente es aplicar todos los conocimientos adquiridos en estos cuatro años, sobre todo los conocimientos adquiridos estos dos últimos años en la mención de Ingeniería de *Software*, pero el desglose de todos los objetivos personales para este proyecto es:

1. Desarrollar una aplicación Web con las principales tecnologías del mercado y que pueda ser útil a la sociedad.

### **1.3. MOTIVACIÓN**

---

2. Aplicar los conocimientos de Ingeniería de *Software* aprendidos en la universidad.
3. Realizar una interfaz fácil de utilizar y con un diseño atractivo para los usuarios.
4. Mejorar las habilidades en la parte de *frontend* con Angular, HTML y CSS.
5. Mejorar en el diseño de aplicaciones Web con Bootstrap para que sean utilizables en cualquier dispositivo.
6. Crear una base de datos MySQL que pueda almacenar el contenido de la aplicación Web de manera organizada.
7. Mejorar las habilidades en la parte de *backend* con NodeJS.
8. Desplegar la aplicación Web para que sea utilizable por cualquier usuario.
9. Aplicar la metodología Scrum a un proyecto de *software*.

## **1.3. Motivación**

La principal motivación de este trabajo surge después de un proceso de observación y análisis de publicaciones de usuarios en diferentes redes sociales, principalmente Instagram y Youtube. Se llegó a la conclusión de que las redes sociales no están preparadas para publicar cómo realizar una receta o los beneficios de un alimento. Estas redes sociales están preparadas para publicar contenido más generalista y por lo tanto quizás un contenido más específico puede no encajar correctamente. También al ver que existen aplicaciones donde se registran hábitos saludables para que lo vea todo el mundo, como las aplicaciones para registrar el ejercicio físico que se realiza.

Posterior a este análisis se observó una carencia de redes sociales adaptadas para publicar contenido relacionado con la alimentación.

De esta manera, se planteó que quizás ya que la mayor parte de los usuarios utiliza las redes sociales para publicar contenido de cualquier tipo, incluyendo comidas, se podría crear una red social solo para este tipo de contenido de manera que todas las publicaciones estén estructuradas para que así puedan ayudar a otras personas.

Por lo tanto, esta aplicación pretende ayudar a las personas a llevar una vida más saludable de manera que tengan acceso a muchas recetas y las experiencias de otros usuarios con estas recetas. De esta forma, en un solo sitio los usuarios tienen rápido acceso a todo lo relacionado con una determinada receta o alimento, pueden ver rápidamente sus beneficios y qué pasos se deben seguir para prepararla.

## **1.4. Aplicaciones similares**

No hay una aplicación que exactamente cumpla el cometido de este proyecto, pero si hay redes sociales que son utilizadas para subir publicaciones de cualquier contenido, entre los

que se incluyen recetas y comidas que es el objetivo de este proyecto. También predominan los *blogs* de comidas. Algunos de los ejemplos similares que son referentes a la hora de realizar este proyecto y que en un futuro serán competencia son estos:

- **Instagram:** usuarios suben todo tipo de publicaciones con texto y fotos. Es habitual encontrar publicaciones relacionadas con recetas o alimentos en esta red social. Cada vez más cuentas son exclusivas para publicar recetas y es habitual encontrar publicaciones de personalidades importantes relacionadas con la comida. Sus principales ventajas son la gran cantidad de información que se puede añadir a una publicación, principalmente fotos y vídeos. Además, la organización y el rápido acceso a los contenidos que hay en ella, la capacidad de comunicación entre todos los usuarios y su accesibilidad desde cualquier tipo de dispositivo hacen que sea una de las redes sociales más utilizadas en el mundo [6].
- **Youtube:** es una red social donde principalmente se publican vídeos de cualquier tipo, acompañados de un texto descriptivo. Hay una gran cantidad de vídeos de cocina con la explicación de cómo hacerlo. Como descripción, los usuarios pueden poner los ingredientes y en los comentarios suelen decir si los usuarios la han hecho. Youtube cuenta con más de 2.500 millones de usuarios, pero su crecimiento se ha estancado y en el último año ha sufrido una pérdida de usuarios [1]. Youtube tiene muchas ventajas respecto a sus competidores como la publicación de vídeos de más larga duración con contenido más detallado que con solo fotos. Además, sus contenidos están bien estructurados de manera que se puede acceder a ellos de una manera rápida mediante una búsqueda. También da la opción de crear comunidades de usuarios, de organizar todo el contenido de un canal y es utilizado por muchos medios de comunicación para publicar noticias [7].
- **Yelp:** en esta aplicación Web los usuarios hacen publicaciones donde suben fotos de lo que han comido pero enlazado al restaurante donde lo han comido. Entonces un Usuario puede buscar un restaurante y encontrar las publicaciones de los usuarios. Es de gran utilidad para los pequeños comercios ya que los comentarios de sus usuarios pueden atraer a nuevos clientes. Sus beneficios para los negocios registrados en ella son muy grandes, ya que tienen organizadas todas las opiniones de los clientes que los han visitado y estas opiniones tienen muchos detalles como fotos y descripciones de las comidas consumidas [8].
- **Recetas de rechupete:** es un *blog* donde se publican recetas con los pasos a seguir para elaborarla y los usuarios publican comentarios en esa receta. Los usuarios pueden publicar comentarios compartiendo sus experiencias relativas a la elaboración. Sin embargo, solo el administrador de la página puede publicar recetas. Pero aun así, este *blog* tiene una gran número de usuarios debido a la variedad de recetas que ofrece, a la buena estructura de la página que hace que el acceso a las recetas sea rápido, los detalles incluidos en la receta para su preparación y los comentarios publicados de los usuarios sobre cada receta [9].
- **Directo al paladar:** en este *blog* se suben artículos sobre ciertos alimentos y recetas con sus beneficios y propiedades. Además, se publican artículos relacionados con la alimentación. Las recetas publicadas vienen con muchos detalles y fotos que ayudan a

## **1.5. ESTRUCTURA DE LA MEMORIA**

---

su preparación y además los usuarios pueden realizar comentarios sobre estas recetas. Para publicar contenido en ella es necesario registrarse y todos sus contenidos son fáciles de acceder mediante una barra de búsqueda [10].

### **1.5. Estructura de la memoria**

Este documento se estructura de la siguiente forma:

**Capítulo 2 Planificación:** en este capítulo se describe la metodología utilizada para el desarrollo del proyecto y cómo se ha adaptado el proyecto. También se describe el presupuesto del proyecto, se analizan los riesgos que pueden aparecer y se define el *Product Backlog* inicial.

**Capítulo 3 Requisitos:** en este capítulo se recogen y especifican los requisitos que debe cumplir el sistema junto con los Actores Principales, los Diagramas de Casos de Uso y la Descripción de los Casos de Uso.

**Capítulo 4 Análisis:** en este capítulo se definen el Modelo de Dominio y el Modelo de Análisis. Además se expone la Realización de Casos de Uso de Análisis para los principales Casos de Uso mediante Diagramas de Secuencia.

**Capítulo 5 Diseño:** en este capítulo se explica la Arquitectura Lógica del Sistema junto a los Patrones de Diseño utilizados y la Arquitectura Física del Sistema. Finalmente se expone cómo se ha diseñado la base de datos y la interfaz gráfica.

**Capítulo 6 Tecnologías utilizadas:** en este capítulo se describen las tecnologías utilizadas para el proyecto, separándolas según sean para la base de datos, para el *frontend*, para el *backend* o para el seguimiento del proyecto.

**Capítulo 7 Implementación y pruebas:** en este capítulo se define la estructura del código de la aplicación, qué tipos de pruebas se van a realizar junto a los Casos de Prueba y finalmente los problemas que han aparecido en el desarrollo proyecto.

**Capítulo 8 Seguimiento del proyecto:** en este capítulo se detalla cómo ha evolucionado el proyecto, describiendo el trabajo realizado en cada *sprint*.

**Capítulo 9 Conclusiones:** en este capítulo se explican las conclusiones obtenidas del proyecto junto a las posibles líneas de trabajo futuras.

**Anexo A Manuales:** Incluye los manuales de despliegue e instalación y el manual de usuario.

**Anexo B Resumen de enlaces adicionales:** Incluye el enlace al repositorio donde está el código del proyecto y el enlace donde está la aplicación Web desplegada.

# Capítulo 2

## Planificación

### 2.1. Introducción

En este capítulo se define el marco de trabajo utilizado para el desarrollo del proyecto que es Scrum. Se explica cómo se ha adaptado Scrum al proyecto y qué riesgos se han encontrado. Finalmente, se define el presupuesto y el *Product Backlog* inicial.

### 2.2. Scrum

#### 2.2.1. Definición y pilares

Scrum [11] es un marco de trabajo para desarrollo ágil de *software* que pretende ayudar a las compañías y personas a generar valor con soluciones adaptativas a problemas complejos. Para ello se aplican unas normas definidas para así poder trabajar colaborativamente en equipo y obtener el mejor resultado posible del proyecto. Como se declaró en el Manifiesto Ágil de 2001, sus cuatro valores fundamentales son [12] [13]:

- Los individuos e interacciones por encima de procesos y herramientas.
- El *software* de trabajo está por encima de documentación completa.
- La colaboración del cliente por encima de negociación de contrato.
- La respuesta al cambio por encima de seguir un plan.

En Scrum se definen un conjunto de artefactos, eventos y roles. En Scrum equipos pequeños de tres a nueve personas entregan productos completos mediante un enfoque de

## 2.2. SCRUM

---

desarrollo iterativo e incremental utilizando bloques de tiempo cortos y fijos llamados *sprints* de una a cuatro semanas.

Scrum está diseñado para el desarrollo de *software* para un mercado competitivo, priorizando sacar el producto al mercado antes que los competidores antes que tener una amplia gama de funcionalidades que pueden no ser importantes. Está indicado para entornos complejos, donde se buscan resultados pronto y los requisitos pueden cambiar a lo largo del proyecto [14]. Los tres pilares de Scrum son:

- **Adaptación:** en el momento que se detecte que un aspecto del trabajo que se está ejecutando se desvía de los límites aceptables establecidos, se debe realizar un ajuste lo antes posible para evitar que la desviación siga aumentando.
- **Transparencia:** el trabajo que se realice debe estar visible tanto para los que realizan el trabajo como para los que lo reciben. Las reuniones permiten que todos conozcan el progreso del proyecto y de esta manera la información fluye de manera ágil en toda la organización. Las decisiones importantes tienen que basarse en el estado actual de los artefactos.
- **Inspección:** se deben realizar inspecciones sobre el progreso del proyecto. De esta manera si se detectan variaciones respecto a los objetivos establecidos, se puedan realizar correcciones para que el progreso del proyecto finalmente lleve a la meta definida. Estas revisiones deben realizarse con frecuencia y para detectar cuanto antes estas desviaciones.

### 2.2.2. Artefactos

Los artefactos representan trabajo o valor, garantizan la transparencia y registran toda la información fundamental del proceso de Scrum. Scrum tiene tres artefactos:

- **Product Backlog:** es una lista ordenada de las características y requisitos que debe tener el producto. El responsable es el *Product Owner*, que puede añadir y ordenar ítems. Se crea al principio del proyecto, pero no es fija y se pueden ir añadiendo elementos o cambiar su orden. Contiene todos los requisitos, mejoras, características y funcionalidades que se pueden realizar en el futuro sobre el proyecto. Los elementos de esta lista están expresados en forma de historias de usuario.
- **Sprint Backlog:** son los elementos seleccionados del *Product Backlog* en los que se va a trabajar durante el *sprint*, es decir contiene las historias de usuario en las que se va a trabajar en este *sprint*. El equipo de desarrollo es el responsable del *Sprint Backlog*, y sus elementos tienen que estar con suficiente nivel de detalle para que las reuniones diarias puedan ver el progreso de cada elemento. Es flexible y puede ser actualizado durante el *sprint*.
- **Incremento:** es el resultado de un *sprint*. El desarrollo ágil se basa en el desarrollo iterativo e incremental por lo tanto el resultado del *sprint* debe poderse añadir a los

incrementos de los *sprints* previos. El responsable de cada incremento es el equipo de desarrollo y el incremento a la finalización de cada *sprint* debe ser funcional y que se pueda utilizar.

### 2.2.3. Roles

El equipo de Scrum está formado por un grupo de trabajo compuesto por desarrolladores, el *Scrum Master* y el *Product Owner*. Cada uno tiene unas responsabilidades. Es importante que los equipos de Scrum sean multifuncionales, de manera que cada miembro tenga las habilidades necesarias para crear valor en cada *sprint* y que sean autoorganizados de manera que internamente decidan quién hace cada tarea, cómo la hace y cuándo la hace.

- **Scrum Master:** es el encargado de que el proyecto siga los pilares de Scrum. Para ello, se asegura de que todos los integrantes del equipo entiendan los pilares de Scrum y lo pongan en práctica. Además, se encarga de eliminar los impedimentos que puedan surgir durante la ejecución del proyecto, y que afectan al producto entregado. Se encarga de guiar al equipo de desarrollo para que sea autoorganizado y multifuncional. Ayuda al *Product Owner* a gestionar el *Product Backlog* de manera eficiente.
- **Product Owner:** es el encargado de maximizar y optimizar el valor del producto. Actúa como interlocutor con los *stakeholders* y comunica las peticiones de los clientes al equipo de desarrollo. Se encarga de gestionar el *Product Backlog* expresando claramente los elementos que la componen y ordenándolos según prioridad para que el resultado final sea el esperado.
- **Equipo de desarrollo:** se encargan de desarrollar el producto. En cada *sprint*, crean un incremento a partir de los elementos seleccionados del *Product Backlog* y que colocan en el *Sprint Backlog*. Deben conocer su rol y las responsabilidades que tienen asignadas. Debe ser lo suficientemente grande como para poder realizar el trabajo a tiempo, pero de manera que se puedan comunicar y gestionar de manera correcta, por lo tanto, el tamaño adecuado es de entre tres a nueve miembros.

### 2.2.4. Eventos

Los eventos se realizan para inspeccionar y adaptar el estado actual de los artefactos de Scrum. Los eventos se encargan de que la transparencia se cumpla y están definidos de manera regular para así evitar reuniones no planeadas. Es conveniente que se realicen en el mismo tiempo y en el mismo lugar siempre. Los eventos en Scrum son los siguientes:

- **Sprint:** el desarrollo del producto es realizado mediante iteraciones sucesivas llamadas *sprints*. En cada *sprint* se debe declarar el objetivo y qué se va a entregar al finalizar este. Es el período en el cual se lleva el trabajo en sí. La duración de los *sprint* es recomendable que sea constante, definida por el equipo y entre una y cuatro semanas. Esta duración se puede ir ajustando en base al ritmo de trabajo del equipo. Al final

## 2.3. ADAPTACIÓN DE SCRUM AL PROYECTO

---

del *sprint*, se debe generar un incremento que funcione y se pueda entregar al cliente. Cada *sprint* comienza nada más terminar el anterior.

- **Sprint Planning:** reunión inicial al comienzo de cada *sprint* donde se define qué se va a llevar a cabo. Es importante que participen todos los miembros del equipo de Scrum y se define qué se va a hacer en el *sprint*, cuáles son los objetivos del *sprint* y se define el incremento que se entregará al final del *sprint*. Se seleccionan los elementos del *Product Backlog* que darán lugar al incremento y se posicionan en el *Sprint Backlog*. Se deben responder a las preguntas: ¿Qué puede hacerse en este sprint? y ¿Cómo se conseguirá realizar el trabajo seleccionado?
- **Daily Scrum:** es una reunión realizada diariamente por el equipo Scrum con una duración de 15 minutos. El objetivo es que los miembros del equipo mantengan actualizados a los demás sobre su trabajo, comentando qué problemas han encontrado y qué planean hacer para el día actual. Es recomendable que se realice siempre a la misma hora y en el mismo lugar siempre. Esta reunión ayuda a evaluar el progreso del proyecto. Cada miembro del equipo debe comentar qué hizo ayer, qué hará hoy y qué impedimentos ve.
- **Sprint Review:** es una reunión que se realiza al final del *sprint*. Los miembros del equipo Scrum presentan el incremento realizado en el *sprint* a los *stakeholders* y se proponen nuevas modificaciones y mejoras para los próximos *sprints*. Si es necesario se puede adaptar el *Product Backlog*. Sus objetivos son facilitar la retroalimentación de la información y fomentar la colaboración. El *Product Owner* debe comentar qué elementos no se han podido completar y cuáles si y el equipo de desarrollo debe comentar qué problemas se han encontrado y cómo los resolvieron.
- **Sprint Retrospective:** es la última reunión de un *sprint*. Se realiza al final del *sprint* y en ella se analiza cómo ha ido el último *sprint* y se valoran mejoras para los siguientes. Principalmente los temas tratados están relacionados sobre qué ha ido bien y qué ha ido mal en el último *sprint*, qué problemas se han encontrado y se crea un plan para implementar las mejoras de cara a los próximos *sprints*.

## 2.3. Adaptación de Scrum al proyecto

Se ha tenido que adaptar la metodología Scrum a este proyecto ya que no se dispone de todos los roles que se definen en Scrum y lo mismo sucede con los eventos de Scrum ya que se han tenido que adaptar a la magnitud de este proyecto, ya que este proyecto se trata de un Trabajo de Fin de Grado. Sin embargo, hay muchas razones por la que aplicar Scrum a este proyecto debido a que pese no se vaya a desarrollar un proyecto para un cliente que vaya a salir al mercado y que no se disponga de todos los roles que se definen en Scrum, se pueden realizar adaptaciones para que los pilares básicos de la metodología Scrum se puedan aplicar con éxito al proyecto.

El alumno tendrá los roles de equipo de desarrollo y *Product Owner*, debido a que será el alumno quien desarrollará la aplicación Web y también definirá los requisitos e historias de usuario que después desarrollará en los diferentes *sprints*. Los tutores actuarán como *Scrum*

Master ya que se encargarán de revisar que se cumpla la metodología Scrum y además servirán de apoyo al alumno para resolver dudas que puedan aparecer durante el desarrollo del proyecto.

Para este proyecto, Scrum es la metodología más adecuada ya que los requisitos no están definidos desde el inicio y son susceptibles de cambiar a lo largo del desarrollo. Esto se debe a que gran parte de las decisiones del desarrollo serán tomadas por parte del desarrollador con la ayuda de los tutores y al inicio solo se parte de una idea general de cómo se quiere que sea la aplicación Web, pero según se vayan implementando las diferentes partes del proyecto puede que los requisitos cambien.

La duración de los *sprints* será de dos semanas y media aproximadamente, realizándose al final de cada *sprint* la *Sprint Review* y *Sprint Retrospective* para revisar cómo se ha desarrollado el *sprint* y qué se puede mejorar de cara a los próximos. Al inicio de cada *sprint* se realizará la *Sprint Planning* para ajustar los objetivos del *sprint* y las historias de usuario que se van a desarrollar. El final de cada *sprint* coincidirá con el inicio del siguiente. En la *Sprint Retrospective* y *Sprint Planning* únicamente participará el alumno mientras que en la *Sprint Review* participarán los tutores también de manera que se les mostrará el incremento generado por el *sprint* y se comentarán posibles mejoras para si hay que realizar algún cambio en el *Product Backlog*. La *Daily Scrum* la realizará el alumno individualmente cada día que avance en el desarrollo del proyecto comentando qué se hizo el día anterior y hasta donde se llegó, qué problemas se tienen que solucionar y qué se va a realizar en el siguiente día.

Por último, todos los artefactos de Scrum, que son el *Product Backlog*, *Sprint Backlog* y los incrementos son responsabilidad del alumno ya que ejerce los papeles de equipo de desarrollo y *Product Owner*.

Este proyecto es para el Trabajo de Fin de Grado del Grado en Ingeniería Informática de la Universidad de Valladolid. Esta asignatura corresponde con 12 créditos lo que equivale a 300 horas.

El objetivo es tener el proyecto finalizado a mediados de junio para tener cierto margen de tiempo hasta la entrega y defensa por si surgen imprevistos o si aparecen mejoras posibles. De esta manera si algún riesgo de los descritos aparece y hace que el proyecto se retrase, utilizando la fecha de inicio del proyecto y de cada *sprint* como la fecha de inicio más temprana posible, se tiene un margen de tiempo lo que da flexibilidad para imprevistos. Así se podría realizar un *sprint* de refuerzo si es necesario antes de la fecha límite para la entrega del proyecto. Este *sprint* de refuerzo puede ser dedicado a realizar tareas que no se han realizado previamente en los anteriores *sprint* o para mejoras en los entregables finales en caso de que ya disponga de una versión estable que cumpla con los requisitos más importantes y definidos al principio del proyecto.

La carga de trabajo en horas de cada *sprint* es variable y va de más a menos. Los dos primeros *sprints* tienen una carga muy elevada debido a que es cuando el equipo de desarrollo que es el alumno tiene más disponibilidad para avanzar en el proyecto. Por el contrario, los dos últimos *sprints* tienen una duración más reducida ya que se estima que el equipo de desarrollo tenga menos disponibilidad para trabajar en el proyecto.

La Tabla 2.1 muestra la planificación de los *sprints* del proyecto.

## 2.4. ANÁLISIS DE RIESGOS

---

Sprint	Fecha de inicio	Fecha de fin	Carga de trabajo
1	30/1/2023	16/2/2023	80 horas
2	16/2/2023	5/3/2023	80 horas
3	5/3/2023	22/3/2023	30 horas
4	22/3/2023	8/4/2023	30 horas
5	8/4/2023	25/4/2023	30 horas
6	25/4/2023	12/5/2023	30 horas
7	12/5/2023	29/5/2023	10 horas
8	29/5/2023	15/6/2023	10 horas

Tabla 2.1: Planificación de los *sprints*.

## 2.4. Análisis de riesgos

Un riesgo [15] es un evento o condición incierta que, si sucede, tiene un efecto negativo o positivo en alguno de los objetivos del proyecto. Los riesgos tienen una o varias causas que son el detonante y provocan una o varias consecuencias que son sus resultados. Para cada riesgo se debe establecer un plan de mitigación para reducir la probabilidad de que sucedan las causas que provocan el riesgo y un plan de contingencia para reducir el impacto de las consecuencias cuando el riesgo finalmente se produce. Para evitar que los riesgos ocurran o en caso de que ocurran afecten lo menos posible al proyecto, hay que realizar un análisis de los riesgos.

Para cada riesgo se definen:

- **Identificador:** identificador para situarlo en la Matriz de Riesgos.
- **Título:** nombre de cuál es el riesgo.
- **Descripción:** breve comentario explicando el riesgo.
- **Probabilidad:** la probabilidad de que finalmente el riesgo ocurra. Puede ser baja, media o alta.
- **Impacto:** importancia que tienen las consecuencias del riesgo en el desarrollo del proyecto si finalmente el riesgo se materializa. Puede ser bajo, medio o alto.
- **Plan de mitigación:** estrategias para reducir la probabilidad de que el riesgo finalmente ocurra.
- **Plan de contingencia:** estrategias que se utilizarán solo si el riesgo se produce y cuya finalidad será reducir el impacto del riesgo.

Los riesgos se pueden dividir en dos clases:

- **Riesgos de proyecto:** están relacionados con los objetivos del proyecto como pueden ser los relacionados con los plazos de tiempo establecidos para llevar a cabo el proyecto y el presupuesto establecido para realizar el proyecto.
- **Riesgos de negocio:** están relacionados con el éxito del producto en el mercado. Algunos riesgos de negocio en este proyecto son la aceptación que tenga por parte de los usuarios la aplicación Web o las mejoras realizadas por las aplicaciones que compitan con esta aplicación que pueden hacer que la aplicación Web realizada en este proyecto no sea la mejor del mercado.

Principalmente se va a prestar atención a los riesgos de proyecto ya que son actualmente los que se pueden controlar, también debido al alcance del proyecto que es para un Trabajo de Fin de Grado.

Las tareas principales que se van a utilizar para lidiar con los riesgos son las siguientes:

- **Identificación de los riesgos:** en primer paso, se enumerarán todos los riesgos posibles que puedan aparecer durante el desarrollo del proyecto. Para ello, se realizará una lluvia de ideas y se aplicará parte de la experiencia de proyectos pasados.
- **Análisis y establecimiento de prioridades:** se ordenarán los riesgos según su probabilidad e impacto para saber en cuáles se debe poner mayor atención y cuáles pueden tener un impacto mayor de cara a la finalización con éxito del proyecto.
- **Planificación de riesgos:** se establecerá cómo actuar para evitar que estos ocurran mediante planes de mitigación y cómo actuar en caso de que finalmente ocurran con planes de contingencia.
- **Monitorización de riesgos:** es una actividad continua durante la realización del proyecto donde se debe vigilar si un riesgo está ocurriendo o si en un futuro cercano sus posibilidades de ocurrir han aumentado, para así aplicar los planes predefinidos para enfrentarlo. Esto permitirá responder de manera adecuada y eficiente ante cualquier riesgo.

Las Tablas 2.2 a 2.13 muestran los riesgos encontrados para el proyecto.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-01
Título	Falta de cualificación con las tecnologías utilizadas.
Descripción	El equipo de desarrollo no tiene los conocimientos necesarios para realizar la aplicación Web con las tecnologías seleccionadas.
Probabilidad	Media.
Impacto	Medio.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Trabajar para tener la cualificación necesaria en las tecnologías que se van a utilizar antes del comienzo del proyecto.</li> <li>■ Utilizar una tecnología con la que el equipo de desarrollo si esté familiarizada para el desarrollo del proyecto.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Consultar con expertos o los profesores los problemas que puedan aparecer en el proyecto con determinadas tecnologías.</li> <li>■ Cambiar el desarrollo del proyecto a una tecnología que el equipo de desarrollo controle.</li> </ul>

Tabla 2.2: R-01. Falta de cualificación con las tecnologías utilizadas.

## CAPÍTULO 2. PLANIFICACIÓN

---

Identificador	R-02
Título	Estimación de los plazos de tiempo incorrecta.
Descripción	La estimación inicial para llevar a cabo cada parte del proyecto no se ajusta al ritmo que finalmente se lleva a la hora de ejecutar el proyecto.
Probabilidad	Baja.
Impacto	Alto.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Iniciar el proyecto lo antes posible para tener un margen de tiempo al final para enfrentar posibles imprevistos.</li> <li>■ Realizar estimaciones con intervalos, con una estimación pesimista y una estimación optimista.</li> <li>■ Utilizar Scrum para el desarrollo del proyecto, realizando diferentes <i>sprints</i> y al final de cada <i>sprint</i> entregando un incremento que funcione.</li> <li>■ Evaluar en la <i>Sprint Retrospective</i> posibles mejoras para los siguientes <i>sprints</i>.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Disminuir el alcance y la funcionalidad del proyecto y centrar los esfuerzos en solo la funcionalidad más importante.</li> <li>■ Tratar de realizar más horas en horarios no habituales.</li> <li>■ Posponer las tareas de un <i>sprint</i> que no se puedan realizar, al siguiente <i>sprint</i>.</li> </ul>

Tabla 2.3: R-02. Estimación de los plazos de tiempo incorrecta.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-03
Título	Desarrollo de la interfaz de la aplicación incorrecta.
Descripción	La interfaz de la aplicación no es fácilmente usable y eficiente y por lo tanto los usuarios no entienden el funcionamiento de la aplicación Web lo que hace que la abandonen.
Probabilidad	Baja.
Impacto	Medio.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Realizar estudios de cómo son las interfaces de las aplicaciones Web similares a la de este proyecto.</li> <li>■ Realizar bocetos antes de empezar a desarrollar la parte de la interfaz.</li> <li>■ Consultar diferentes potenciales usuarios y expertos sobre cómo debe ser la interfaz.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Modificar la interfaz para que sea fácilmente usable por un Usuario.</li> </ul>

Tabla 2.4: R-03. Desarrollo de la interfaz de la aplicación incorrecta.

Identificador	R-04
Título	Fallo en el equipo de trabajo donde se realiza el proyecto.
Descripción	El equipo de trabajo donde se desarrolla el proyecto, tanto el código como la documentación, deja de funcionar por completo o su funcionamiento se ve claramente afectado lo que dificulta el trabajo con él.
Probabilidad	Baja.
Impacto	Alto.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Tener un equipo suplente por si el equipo de trabajo principal falla.</li> <li>■ Realizar copias de seguridad para mantener los datos del equipo a salvo.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Obtener un equipo de trabajo para seguir trabajando y reemplazar el averiado.</li> <li>■ Hacer las tareas que se puedan realizar sin un equipo de trabajo y después cuando se obtenga un nuevo equipo, añadir este contenido.</li> </ul>

Tabla 2.5: R-04. Fallo en el equipo de trabajo donde se realiza el proyecto.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-05
Título	Cambios tardíos y muy importantes de los requisitos de la aplicación Web.
Descripción	El <i>Product Owner</i> que en este caso es el alumno quiere realizar cambios en los requisitos y afectan considerablemente al trabajo ya realizado. Además, estos cambios se han comunicado cerca de las fechas de finalización del proyecto.
Probabilidad	Baja.
Impacto	Alto.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Definir al comienzo del proyecto de la manera más detallada posible todos los aspectos del proyecto.</li> <li>■ Justificar todas las decisiones que se tomen respecto al diseño.</li> <li>■ Utilizar Scrum ya que es la metodología más adecuada cuando hay incertidumbre respecto a los requisitos.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Realizar solo los cambios más importantes y que más beneficios pueden proporcionar a la aplicación Web.</li> <li>■ Evaluar el beneficio y el coste que llevaría realizar estos nuevos cambios.</li> </ul>

Tabla 2.6: R-05. Cambios tardíos y muy importantes de los requisitos de la aplicación Web.

## CAPÍTULO 2. PLANIFICACIÓN

---

Identificador	R-06
Título	Enfermedad del alumno.
Descripción	El alumno puede tener problemas de salud debido a su historial previo lo que puede retrasar el desarrollo del proyecto.
Probabilidad	Media.
Impacto	Medio.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Llevar una vida saludable cuidando la salud y evitar los lugares con alta concentración de virus o las personas que actualmente están contagiadas.</li> <li>■ Tener márgenes de tiempo y flexibilidad en la planificación para que si surge algún imprevisto se tenga tiempo para compensar posibles retrasos.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Mover las tareas que no se hayan podido realizar a los siguientes <i>sprints</i>.</li> <li>■ Disminuir el alcance del proyecto y la funcionalidad a implementar.</li> </ul>

Tabla 2.7: R-06. Enfermedad del alumno.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-07
Título	Actualización con grandes cambios de las tecnologías utilizadas.
Descripción	Las tecnologías utilizadas están en constante actualización y alguna de estas actualizaciones introducen nuevas funcionalidades que cambian radicalmente lo que se tenía implementado en el proyecto.
Probabilidad	Baja.
Impacto	Bajo.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Preparar el equipo de desarrollo para aprender nuevas funcionalidades y tecnologías en caso de que se necesite.</li> <li>■ Revisar antes de seleccionar las tecnologías que se van a utilizar en el proyecto, las posibles actualizaciones que prevén publicar en los próximos meses.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Adaptar el código desarrollado a las nuevas funcionalidades.</li> <li>■ No actualizar el código desarrollado a la nueva versión y mantenerlo en la versión que funcione.</li> </ul>

Tabla 2.8: R-07. Actualización con grandes cambios de las tecnologías utilizadas.

Identificador	R-08
Título	Pérdida de datos del proyecto.
Descripción	Fallo en el equipo de trabajo o un fallo humano a la hora de guardar datos hace que el trabajo que se haya avanzado se pierda.
Probabilidad	Baja.
Impacto	Alto.
Plan de mitigación	<ul style="list-style-type: none"><li>■ Realizar copias de seguridad del código realizado en un repositorio de Gitlab y guardar el contenido de la documentación en la nube.</li></ul>
Plan de contingencia	<ul style="list-style-type: none"><li>■ Tratar de recuperar el contenido perdido mediante aplicaciones para este cometido.</li><li>■ Rehacer el trabajo hecho.</li></ul>

Tabla 2.9: R-08. Pérdida de datos del proyecto.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-09
Título	Problemas con la conexión a internet.
Descripción	La conexión a internet puede que sea lenta o se pierda debido a la localización donde el equipo de desarrollo va a realizar el proyecto.
Probabilidad	Baja.
Impacto	Medio.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Preparar lugares alternativos donde se pueda trabajar con conexión a internet.</li> <li>■ Definir un conjunto de tareas que se puedan realizar sin conexión a internet para si sucede este problema.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Realizar tareas que no requieran conexión a internet.</li> <li>■ Retrasar las tareas que requieran conexión a internet.</li> <li>■ Desplazarse otro espacio de trabajo con conexión a internet.</li> </ul>

Tabla 2.10: R-09. Problemas con la conexión a internet.

Identificador	R-10
Título	Falta de tiempo para trabajar en el proyecto.
Descripción	El alumno que compone el equipo de desarrollo no tiene tiempo para avanzar en el proyecto debido a las demás asignaturas que tiene que realizar.
Probabilidad	Media.
Impacto	Alto.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Comenzar el desarrollo del proyecto lo antes posible para así tener margen de tiempo al finalizar.</li> <li>■ Definir un horario de trabajo con los diferentes áreas que tiene que trabajar el alumno.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Centrar los esfuerzos en implementar la funcionalidad más importante del proyecto y reducir el alcance del proyecto.</li> <li>■ Trabajar en el proyecto en horas no habituales realizando un esfuerzo extra.</li> <li>■ Posponer el trabajo que no se pueda realizar en un <i>sprint</i> al siguiente <i>sprint</i>.</li> </ul>

Tabla 2.11: R-10. Falta de tiempo para trabajar en el proyecto.

## 2.4. ANÁLISIS DE RIESGOS

---

Identificador	R-11
Título	Problemas para contactar con los tutores o con otros profesores.
Descripción	Es difícil acordar reuniones con los tutores para revisar el avance del proyecto y hay incompatibilidad de horarios para hablar con algún profesor sobre alguna funcionalidad específica del proyecto.
Probabilidad	Baja.
Impacto	Bajo.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Tratar de establecer flexibilidad de horarios para acordar estos encuentros.</li> <li>■ Buscar varios contactos que puedan ayudar a resolver los problemas que se encuentren y tener alternativas en caso de que un contacto no esté disponible.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Seguir con otras tareas que no requieran del visto bueno o de la ayuda de los expertos.</li> <li>■ Realizar reuniones más esporádicas y de menor duración tratando los asuntos más importantes.</li> </ul>

Tabla 2.12: R-11. Problemas para contactar con los tutores o con otros profesores.

Identificador	R-12
Título	Aparición de <i>bugs</i> con complicada solución.
Descripción	En el desarrollo de la aplicación aparecen problemas en los que no se consigue encontrar una solución y que no permiten realizar la funcionalidad deseada.
Probabilidad	Media.
Impacto	Medio.
Plan de mitigación	<ul style="list-style-type: none"> <li>■ Formar al equipo de desarrollo en las tecnologías a utilizar para así saber enfrenar todos los problemas que puedan surgir.</li> <li>■ Definir diferentes opciones para las partes más complicadas del proyecto para en caso de que alguna no se pueda implementar.</li> </ul>
Plan de contingencia	<ul style="list-style-type: none"> <li>■ Contactar con expertos que puedan ayudar a solucionar el problema.</li> <li>■ Buscar soluciones alternativas al problema de manera que se pueda lograr el objetivo de otra forma.</li> </ul>

Tabla 2.13: R-12. Aparición de *bugs* con complicada solución.

Una vez identificados los riesgos y definido para cada uno su impacto, probabilidad y planes de mitigación y contingencia, se procede a crear una Matriz de Riesgos. Esta es una herramienta que se utiliza para evaluar la probabilidad e impacto de un riesgo, lo que ayuda a determinar la prioridad de cada riesgo y gestionarlos de la mejor manera posible [16].

La Tabla 2.14 muestra una Matriz de Riesgos.

## 2.5. PRESUPUESTO

---

Probabilidad / impacto	Bajo	Medio	Alto
Baja	Monitorizar	Monitorizar	Aplicar los planes de mitigación
Media	Monitorizar	Aplicar los planes de mitigación	Aplicar los planes de mitigación y tener preparados los planes de contingencia para si son necesario aplicarlos
Alta	Aplicar los planes de mitigación	Aplicar los planes de mitigación y tener preparados los planes de contingencia para si son necesario aplicarlos	Aplicar los planes de mitigación y tener preparados los planes de contingencia para si son necesario aplicarlos

Tabla 2.14: Matriz de Riesgos.

De esta manera, situando los riesgos definidos para este proyecto en la Matriz de Riesgos se podrá definir la prioridad de cada uno de ellos para así saber cómo actuar dependiendo el impacto y la probabilidad que tienen.

La Tabla 2.15 sitúa los riesgos del proyecto en la Matriz de Riesgos.

Probabilidad / impacto	Bajo	Medio	Alto
Baja	R7,R11	R3,R9	R2,R4,R5,R8
Media		R1,R6,R12	R10
Alta			

Tabla 2.15: Situación de los riesgos del proyecto en la Matriz de Riesgos.

## 2.5. Presupuesto

Por último, se debe definir el presupuesto del proyecto. Como este proyecto es para un Trabajo de Fin de Grado con fines académicos, se han definido dos presupuestos. El primer presupuesto consiste en un presupuesto simulado, donde se recoge el coste que tendría el proyecto en el ámbito empresarial si el proyecto fuese a salir al mercado. El segundo presupuesto consiste en el presupuesto real, que recoge el coste que realmente tendrá el proyecto al tratarse de un Trabajo de Fin de Grado.

### 2.5.1. Presupuesto simulado

El coste en sueldos pagados a los desarrolladores para este proyecto únicamente se debe al sueldo pagado al alumno que será el único miembro del equipo de desarrollo. Como la experiencia es un factor muy importante a la hora de determinar el sueldo en este sector, el alumno que conforma el equipo de desarrollo se considera un programador *junior* ya que no tiene experiencia laboral. El sueldo de un programador *junior* en enero de 2023 es de 22.875€ al año en España [17]. Hay que tener en cuenta la contribución de la empresa con la seguridad social que actualmente se sitúa en el 28 % del sueldo bruto, por lo tanto, el coste a la empresa por cada trabajador por año sería de 29280€ [18].

Teniendo en cuenta que se trabajan alrededor de 1800 horas al año y que la jornada laboral máxima a la semana es de 40 horas, se obtiene que el coste de trabajo de un programador *junior* es de 16,26€ por hora. Considerando que el proyecto a realizar tiene 300 horas aproximadamente, el coste en trabajadores para este proyecto será de 4878€ [19].

Para realizar este proyecto se utiliza el portátil personal del alumno que conforma el equipo de desarrollo que es un Lenovo Ideapad 3 valorado en 500€. Es necesario recompensar al trabajador por la amortización de su equipo y teniendo en cuenta que este portátil tiene una vida útil de cuatro años, lo que son 48 meses, se obtiene un coste de 10,41€ al mes y como la duración de este proyecto será de en torno a cuatro meses, la amortización que debe recibir el trabajador por el uso de su equipo es de 41,64€.

En cuanto al software utilizado, para el desarrollo del proyecto se van a utilizar Visual Studio Code, Microsoft Word, Gitlab, Astah Professional y Balsamiq.

Astah Professional tiene un coste de 2,5€/mes para su licencia, por lo tanto para los cuatro meses equivaldría a 10€. En el caso de Balsamiq, tiene un coste de 8,41€ al mes, lo que se convierte para los cuatro meses de la duración del proyecto en 33,64€. Las versiones de Visual Studio Code son gratuitas. Gitlab que se usará para almacenar el código tiene un coste de 17,7 € al mes, por lo tanto 70,8€ para los cuatro meses de la duración del proyecto. Finalmente, el paquete de Microsoft Office que incluye Word tiene un precio de 5,75€ al mes, lo que generaría un coste de 23€ para los cuatro meses del proyecto.

Railway se utilizará para el despliegue de la aplicación Web. Debido a las características de la aplicación tendría un coste de 4,67€ al mes su despliegue, y por lo tanto serían 18,68€ para los cuatro meses.

Sumando todos los costes darían en total 5075,76€ de coste total para el proyecto. A estos costes se debe añadirle un 25 % a modo de fondo de maniobra, como medida para tener capacidad de hacer pago a todos estos costes en caso de que algún riesgo aparezca y el coste del proyecto se incremente [20]. Por lo tanto el coste total del proyecto es de 6344,7€.

La tabla 2.16 muestra los costes del presupuesto simulado.

## 2.6. PRODUCT BACKLOG INICIAL

---

Elemento	Coste	Duración	Total
Sueldo desarrollador	16,26€/h	300 horas	4878€
Ordenador desarrollador	10,41€/mes	4 meses	41,64€
Licencia Astah Professional	2,5€/mes	4 meses	10€
Licencia Balsamiq	8,41€/mes	4 meses	33,64€
Licencia Gitlab	17,7€/mes	4 meses	70,8€
Licencia Microsoft Office	5,75€/mes	4 meses	23€
Despliegue Railway	4,67€/mes	4 meses	18,68€
<b>TOTAL</b>			<b>5075,76€</b>
<b>TOTAL +25 %</b>			<b>6344,7€</b>

Tabla 2.16: Presupuesto simulado

### 2.5.2. Presupuesto real

Para el coste real del proyecto, como el alumno está realizando el Trabajo de Fin de Grado con fines académicos no recibirá remuneración alguna.

Para las herramientas de *software* utilizadas en el proyecto, todas se van a utilizar en su versión gratuita, ya sea mediante la versión con un periodo de prueba que proporcionan o con la licencia que proporciona la Escuela de Ingeniería Informática para estudiantes. Tanto Gitlab, Astah y Microsoft Office la Escuela de Ingeniería Informática proporciona licencias de estudiantes y para Balsamiq se utilizará la versión de prueba gratuita.

Por otro lado, para el despliegue se utilizará Railway que también tiene un plan gratuito para el despliegue de aplicaciones Web de reducido alcance como es el caso de este proyecto.

Por lo tanto el único coste real del proyecto sería la amortización del equipo de trabajo del estudiante que era de 41,64€.

La tabla 2.17 muestra los costes del presupuesto real.

Elemento	Coste	Duración	Total
Ordenador desarrollador	10,41€/mes	4 meses	41,64€
<b>TOTAL</b>			<b>41,64€</b>

Tabla 2.17: Presupuesto real

## 2.6. Product Backlog inicial

Como el *Product Owner* será el alumno, él mismo se encarga de definir las historias de usuario del *Product Backlog*. Una historia de usuario [21] es una explicación general de

una función de *software* que debe tener el producto desde la perspectiva del Usuario final. Consisten en pocas frases que describen un resultado deseado y por lo tanto no entran en detalles. Una historia de usuario suele expresarse en una frase con la siguiente estructura: “como [perfil],[quiero][para]” donde el “perfil” hace referencia al perfil de la persona, “quiero” se refiere a la intención, lo que está intentando lograr, y “para” define el problema que quiere resolver o el beneficio que quiere lograr.

El *Product Backlog* inicial se puede ver en las Tablas 2.18 y 2.19:

## 2.6. PRODUCT BACKLOG INICIAL

---

Título	Historia de usuario
Registro	Como Usuario quiero poder registrarme en la aplicación.
Identificación	Como Usuario quiero poder identificarme en la aplicación.
Consulta publicaciones usuarios seguidos	Como Usuario quiero ver las publicaciones de los usuarios a los que sigo.
Seguir Usuario	Como Usuario quiero poder seguir a un Usuario.
Dejar de seguir Usuario	Como Usuario quiero poder dejar de seguir a un Usuario.
Crear receta	Como Usuario quiero crear recetas.
Buscar recetas	Como Usuario quiero buscar recetas.
Buscar publicaciones de una receta	Como Usuario quiero ver las publicaciones de una receta.
Ver detalles de una publicación	Como Usuario quiero ver los detalles de una publicación.
Subir publicación	Como Usuario quiero subir una publicación.
Ver mis datos	Como Usuario quiero ver mis datos de la cuenta.
Editar datos personales	Como Usuario quiero editar mis datos de la cuenta.
Añadir receta a favoritas	Como Usuario quiero añadir una receta a favoritas.
Ver recetas favoritas	Como Usuario quiero ver mis recetas favoritas.
Eliminar receta favorita	Como Usuario quiero poder eliminar una receta de mis favoritas.
Ver detalles de un Usuario	Como Usuario quiero poder los detalles de un Usuario.
Ver detalles receta	Como Usuario quiero ver los detalles de una receta.

Tabla 2.18: *Product Backlog* inicial.

Título	Historia de usuario
Dejar comentario en una publicación	Como Usuario quiero poder dejar un comentario en una publicación.
Buscar alimentos	Como Usuario quiero buscar un alimento.
Buscar publicaciones de un alimento	Como Usuario quiero ver las publicaciones de un alimento.
Eliminar Usuario	Como Administrador quiero poder eliminar un Usuario.
Eliminar receta	Como Administrador quiero poder eliminar una receta.
Eliminar publicación	Como Administrador quiero poder eliminar una publicación.
Eliminar alimento	Como Administrador quiero poder eliminar un alimento.
Editar alimento	Como Administrador quiero editar un alimento.
Crear alimento	Como Administrador quiero crear un alimento.

Tabla 2.19: *Product Backlog* inicial.

## *2.6. PRODUCT BACKLOG INICIAL*

---

# Capítulo 3

## Requisitos

### 3.1. Introducción

En este capítulo se exponen los actores que se relacionarán con el sistema y se describen los diferentes Casos de Uso que se podrán realizar. Se especifican los Requisitos del Sistema tanto Funcionales, No Funcionales y de Información.

### 3.2. Requisitos Funcionales

Los Requisitos Funcionales son declaraciones de cómo se comporta el sistema, es decir, define lo que el sistema debe hacer para satisfacer las necesidades de los clientes. Es la funcionalidad que tendrá el sistema que se va a diseñar y proporcionan una descripción de cómo el sistema responderá ante las acciones de los usuarios [22].

Los Requisitos Funcionales para este sistema son:

- **RF-01** El sistema deberá permitir a un Usuario registrarse en la aplicación Web.
- **RF-02** El sistema deberá permitir a un Usuario identificarse en la aplicación Web.
- **RF-03** El sistema deberá permitir a un Usuario consultar las publicaciones de los usuarios a los que sigue.
- **RF-04** El sistema deberá permitir a un Usuario seguir a otro Usuario.
- **RF-05** El sistema deberá permitir a un Usuario dejar de seguir a un Usuario.
- **RF-06** El sistema deberá permitir a un Usuario crear una receta.
- **RF-07** El sistema deberá permitir a un Usuario consultar recetas por nombre.

### **3.2. REQUISITOS FUNCIONALES**

---

- **RF-08** El sistema deberá permitir a un Usuario consultar las publicaciones enlazadas a una receta.
- **RF-09** El sistema deberá permitir a un Usuario consultar los detalles de una publicación.
- **RF-10** El sistema deberá permitir a un Usuario subir una publicación.
- **RF-11** El sistema deberá permitir a un Usuario consultar los datos de su cuenta.
- **RF-12** El sistema deberá permitir a un Usuario editar los datos de su cuenta.
- **RF-13** El sistema deberá permitir a un Usuario añadir una receta a favoritas.
- **RF-14** El sistema deberá permitir a un Usuario consultar sus recetas favoritas.
- **RF-15** El sistema deberá permitir a un Usuario eliminar una receta de sus favoritas.
- **RF-16** El sistema deberá permitir a un Usuario consultar los detalles de otro Usuario.
- **RF-17** El sistema deberá permitir a un Usuario consultar los detalles de una receta.
- **RF-18** El sistema deberá permitir a un Usuario dejar un comentario en una publicación.
- **RF-19** El sistema deberá permitir a un Usuario consultar alimentos por nombre.
- **RF-20** El sistema deberá permitir a un Usuario consultar las publicaciones enlazadas a un alimento.
- **RF-21** El sistema deberá permitir al Administrador eliminar un Usuario.
- **RF-22** El sistema deberá permitir al Administrador eliminar un alimento.
- **RF-23** El sistema deberá permitir al Administrador eliminar una receta.
- **RF-24** El sistema deberá permitir al Administrador eliminar una publicación.
- **RF-25** El sistema deberá permitir al Administrador editar un alimento.
- **RF-26** El sistema deberá permitir al Administrador crear un alimento.
- **RF-27** El sistema deberá mostrar al Administrador una tabla con todos los alimentos almacenados.
- **RF-28** El sistema deberá mostrar al Administrador una tabla con todos los usuarios registrados.
- **RF-29** El sistema deberá mostrar al Administrador una tabla con todas las recetas creadas por los usuarios.
- **RF-30** El sistema deberá mostrar al Administrador una tabla con todas las publicaciones creadas por los usuarios.
- **RF-31** El sistema deberá permitir a un Usuario cerrar su sesión.

### 3.3. Requisitos No Funcionales

Los Requisitos No Funcionales son las restricciones impuestas al sistema y especifican atributos de calidad del *software* entre otros. Se ocupan de problemas como la escalabilidad, mantenibilidad, rendimiento, portabilidad y seguridad y por lo tanto abordan cuestiones vitales de calidad para los sistemas *software* [22].

Los Requisitos No Funcionales para este sistema son:

- **RNF-01** El sistema debe estar implementado con Angular para la parte de *frontend*.
- **RNF-02** El sistema debe poderse desplegar en Railway.
- **RNF-03** El sistema debe poderse utilizar en cualquier dispositivo.
- **RNF-04** El sistema debe utilizar una base de datos relacional SQL.
- **RNF-05** El sistema debe adaptarse a cualquier tamaño de pantalla.
- **RNF-06** El sistema debe almacenar las contraseñas encriptadas en la base de datos.
- **RNF-07** El sistema debe tener un diseño de interfaz de usuario fácil de aprender.
- **RNF-08** El sistema debe tener un flujo de navegación sencillo.
- **RNF-09** El sistema debe tener una interfaz sencilla de utilizar.
- **RNF-10** El sistema debe ser accesible desde los principales navegadores Web.
- **RNF-11** El sistema debe almacenar todos los datos de la aplicación Web de manera segura.

### 3.4. Requisitos de Información

Los Requisitos de Información describen la información que se va a almacenar y gestionar en el sistema [23]. Los Requisitos de Información para este sistema son:

- **RI-01** El sistema debe almacenar la información correspondiente a cada usuario, en concreto: identificador(id), *username*, *password*, foto, *email*, rol y descripción.
- **RI-02** El sistema debe almacenar la información correspondiente a cada receta, en concreto: identificador(id), resumen, tiempo, título, foto, dificultad e identificador(id) del Usuario creador.
- **RI-03** El sistema debe almacenar la información correspondiente a cada alimento, en concreto: identificador(id), nombre, descripción, foto, calorías, grasas, carbohidratos, proteínas, cantidad, medida y enlace.

### 3.5. ACTORES PRINCIPALES

- **RI-04** El sistema debe almacenar la información correspondiente a cada comentario, en concreto: identificador(id), identificador(id) del Usuario que lo realizó, identificador(id) de la publicación a la que pertenece y comentario.
- **RI-05** El sistema debe almacenar la información correspondiente a cada seguidor, en concreto: identificador(id), identificador(id) del Usuario que ejerce como seguidor e identificador(id) del Usuario que ejerce como seguido.
- **RI-06** El sistema debe almacenar la información correspondiente a cada receta favorita, en concreto: identificador(id), identificador(id) del Usuario al que corresponde e identificador(id) de la receta favorita.
- **RI-07** El sistema debe almacenar la información correspondiente a cada alimento en receta, en concreto: identificador(id), identificador(id) del alimento al que corresponde, identificador(id) de la receta a la que pertenece, cantidad y medida.
- **RI-08** El sistema debe almacenar la información correspondiente a cada paso, en concreto: identificador(id), identificador(id) de la receta a la que pertenece, paso y orden.
- **RI-09** El sistema debe almacenar la información correspondiente a cada publicación, en concreto: identificador(id), descripción, foto, fecha de publicación, título, identificador(id) del Usuario que la creó, identificador(id) del alimento con la que está enlazada e identificador(id) de la receta con la que está enlazada.

## **3.5. Actores Principales**

Se han identificado dos actores que se relacionarán con el sistema, el Usuario y el Administrador. A continuación, se define cada uno de ellos.

### **3.5.1. Usuario**

El Actor Principal de la aplicación Web es el Usuario. El Usuario es un cliente de la aplicación que la utiliza tanto para crear contenidos como recetas o publicaciones como para consumir contenido de otros usuarios. Por lo tanto, el Usuario puede producir contenido mediante la creación de una receta añadiendo toda la información relevante para esta receta para que otros usuarios la puedan realizar, como son los alimentos para su preparación, sus pasos y su dificultad. También puede producir contenido mediante la creación de una publicación enlazada a un alimento o receta, incluyendo en la publicación información sobre su experiencia que puede ser de gran ayuda a otros usuarios.

Por otro lado, el Usuario puede consumir contenido consultando publicaciones de otros usuarios y viendo los detalles de estas publicaciones. Puede consultar las recetas creadas por otros usuarios con todos los detalles para su preparación y de esta manera obtener la información necesaria para prepararlas. Puede añadir las recetas que deseé a favoritas para así poder acceder a su información de una manera más rápida. Puede ver los alimentos

almacenados en la aplicación con toda su información relacionada para así poder encontrar información valiosa sobre ese alimento. Además, cada Usuario puede ver los perfiles de los demás usuarios para así obtener información sobre otros usuarios, seguir a los usuarios que más le gusten y poder estar al tanto de sus publicaciones.

### **3.5.2. Administrador**

El Administrador de la aplicación se encargará de que todos los datos que muestre la aplicación y que se almacenen en la base de datos sean adecuados a este tipo de aplicación y cumplan las normas de convivencia entre los usuarios de la aplicación. De esta manera, podrá gestionar todos los contenidos almacenados en la aplicación Web, como pueden ser las recetas, publicaciones, usuarios y alimentos y eliminar los contenidos que no sean adecuados para esta aplicación Web. Además, tiene la capacidad de crear los alimentos que se muestran en la aplicación para que así los usuarios de la aplicación los puedan usar en sus recetas y puedan ver su información relevante.

## **3.6. Diagrama de Casos de Uso**

Los Casos de Uso son la descripción de una acción o actividad que se puede realizar en el sistema para llevar a cabo un proceso [24]. Mediante la identificación y descripción de los Casos de Uso se definen las acciones más relevantes que se ejecutarán en el sistema y cómo se comportará durante la realización de estas acciones.

En la Figura 3.1 se muestra la primera parte del Diagrama de Casos de Uso para el actor Usuario.

En la Figura 3.2 se muestra la segunda parte del Diagrama de Casos de Uso para el actor Usuario.

En la Figura 3.3 se muestra el Diagrama de Casos de Uso para el actor Administrador.

## **3.7. Descripción Casos de Uso**

En la Descripción de los Casos de Uso se especificará el flujo normal de los principales Casos de Uso, la realización exitosa de una acción o actividad y aquellos flujos alternativos a la hora de realizar cada uno de los Casos de Uso.

Las Tablas 3.1 a 3.23 muestran las Descripciones de los Casos de Uso.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

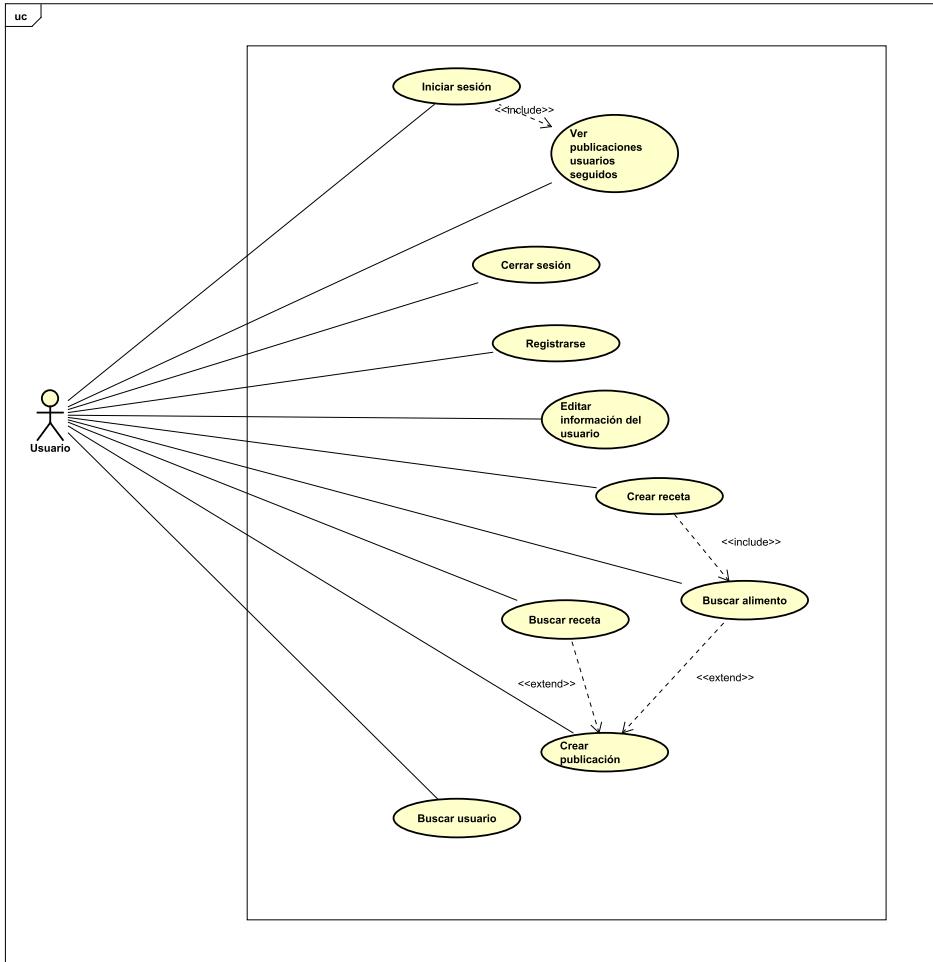


Figura 3.1: Primera parte del Diagrama de Casos de Uso para el actor Usuario.

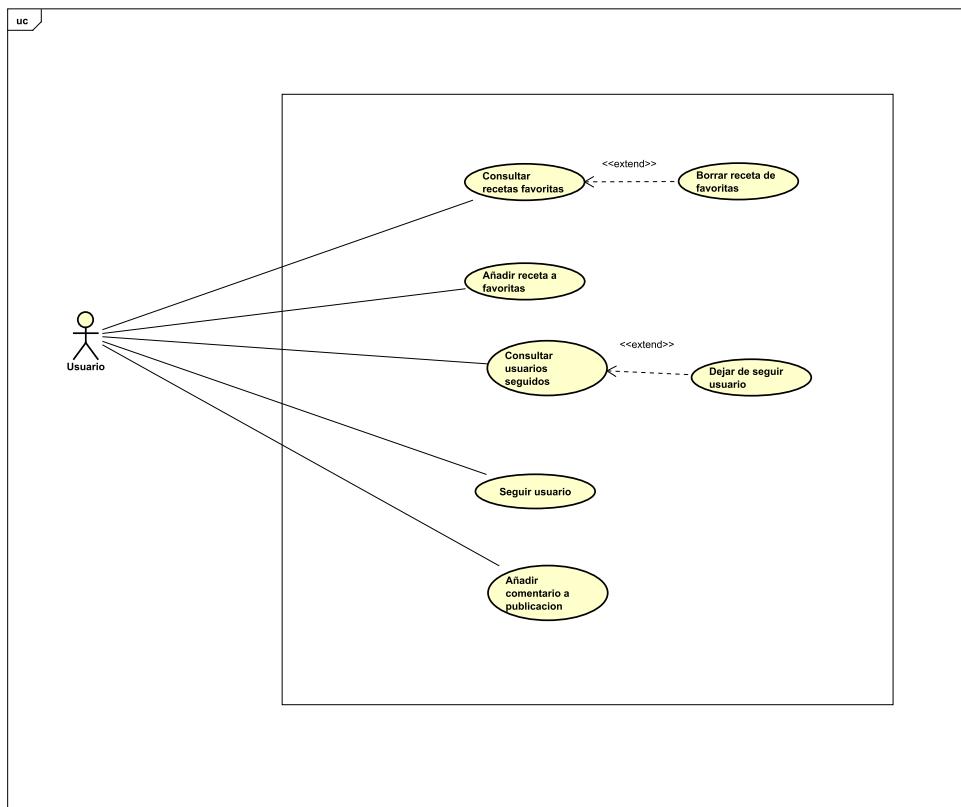


Figura 3.2: Segunda parte del Diagrama de Casos de Uso para el actor Usuario.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

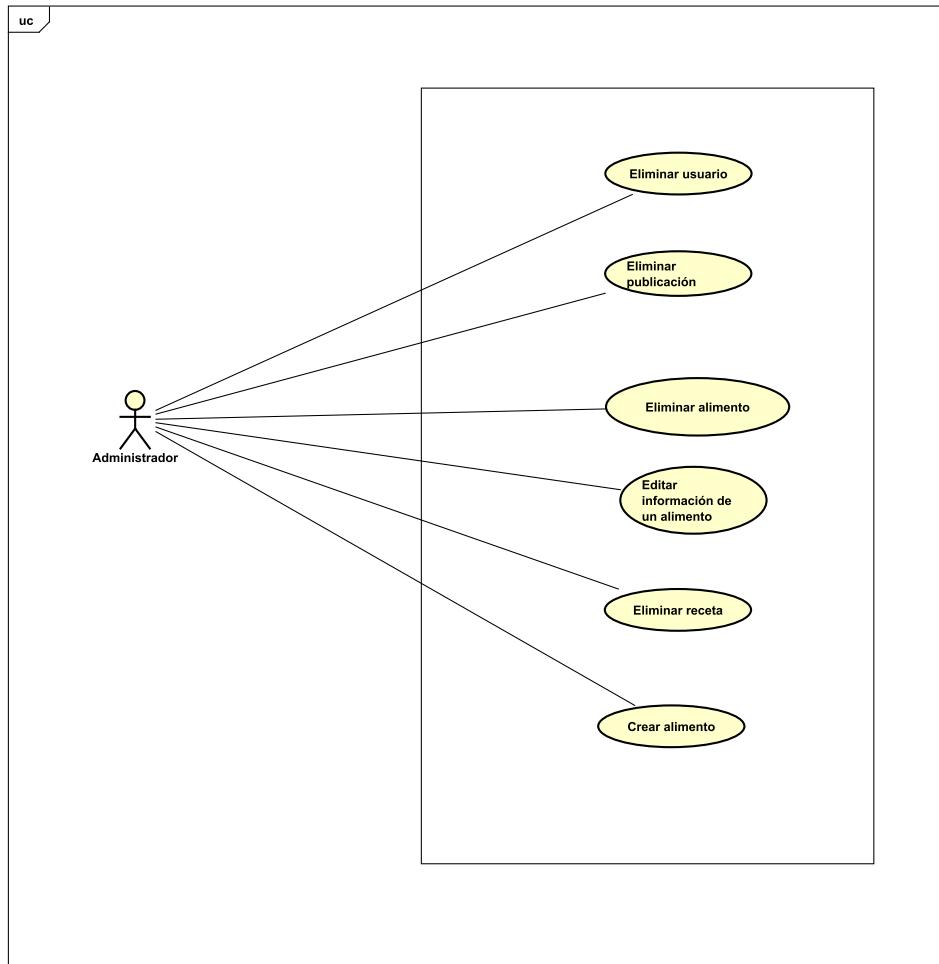


Figura 3.3: Diagrama de Casos de Uso para el actor Administrador.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-1 Iniciar sesión
<b>Descripción:</b>	El Usuario introduce sus datos para identificarse en el sistema y así poder acceder a su funcionalidad.
<b>Actores:</b>	Usuario y Administrador.
<b>Pre-condiciones:</b>	El Usuario no se ha identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario se ha identificado en el sistema y puede acceder a su funcionalidad.

### Secuencia normal

---

1. El Usuario comienza el proceso para iniciar sesión.
2. El sistema solicita *username* y *password*.
3. El Usuario introduce los datos.
4. El sistema valida el formato de los datos introducidos.
5. El sistema solicita confirmación.
6. El Usuario confirma el inicio de sesión.
7. El sistema comprueba que los datos introducidos corresponden a los de un Usuario registrado.
8. El sistema informa al Usuario que ha iniciado sesión con éxito.
9. El sistema carga la información con la que el Usuario se ha identificado.

**Punto de inclusión: CU-5 Ver publicaciones usuarios seguidos.**

11. El Caso de Uso finaliza.

### Secuencias alternativas

---

- 3.a, 6.a)** El Usuario cancela el proceso para identificarse y el Caso de Uso queda sin efecto.

### Excepciones

---

**4.a)** El sistema comprueba que el formato de los datos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

**7.a)** El sistema comprueba que los datos introducidos no corresponden a ningún Usuario, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

Tabla 3.1: Descripción del Caso de Uso *Iniciar sesión*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-2 Registrarse
<b>Descripción:</b>	Un Usuario nuevo desea registrarse en la aplicación para poder acceder a toda la funcionalidad.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario no se ha identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario se ha registrado en el sistema y puede identificarse en él mediante su <i>username</i> y <i>password</i> para acceder a la funcionalidad.

#### Secuencia normal

---

1. El Usuario comienza el proceso para registrarse en la aplicación.
2. El sistema solicita *username*, *password* y *email* al Usuario.
3. El Usuario introduce los datos solicitados.
4. El sistema valida el formato de los datos solicitados.
5. El sistema comprueba que el *email* no ha sido utilizado por otro Usuario.
6. El sistema comprueba que el *username* no ha sido utilizado por otro Usuario.
7. El sistema solicita confirmación para el registro.
8. El Usuario confirma el registro.
9. El sistema informa al Usuario de que se ha registrado con éxito.
10. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 3.a, 8.a)** El Usuario cancela el proceso de registro y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 4.a)** El sistema comprueba que el formato de los datos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.
- 5.a)** El sistema comprueba que el *email* ha sido utilizado por otro Usuario, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.
- 6.a)** El sistema comprueba que el *username* ha sido utilizado por otro Usuario, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

Tabla 3.2: Descripción del Caso de Uso *Registrarse*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-3 Cerrar sesión
<b>Descripción:</b>	Un Usuario identificado en el sistema quiere salir de la aplicación y cerrar su sesión.
<b>Actores:</b>	Usuario y Administrador.
<b>Pre-condiciones:</b>	El Usuario se ha identificado en el sistema y tiene una sesión activa.
<b>Post-condiciones:</b>	Se ha cerrado la sesión del Usuario y ya no podrá acceder a la funcionalidad del sistema hasta que se vuelva a identificar.

### Secuencia normal

---

1. El Usuario solicita cerrar sesión.
2. El sistema solicita confirmación.
3. El Usuario confirma el cierre de sesión.
4. El sistema cierra sesión y elimina la información sobre el Usuario que haya cargado.
5. El Caso de Uso finaliza.

### Secuencias alternativas

---

- 3.a)** El Usuario cancela el proceso de cerrar sesión y el Caso de Uso queda sin efecto.
- 

Tabla 3.3: Descripción del Caso de Uso *Cerrar sesión*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-4 Editar información del Usuario
<b>Descripción:</b>	El Usuario desea cambiar la información de su perfil y los datos de su cuenta
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario se ha identificado en el sistema.
<b>Post-condiciones:</b>	La información del perfil del Usuario y los datos de su cuenta se habrán actualizado con los nuevos campos introducidos.

#### Secuencia normal

---

1. El Usuario selecciona editar la información de su cuenta.
2. El sistema muestra la información del perfil del Usuario y los datos de su cuenta.
3. El sistema solicita los nuevos valores para estos campos.
4. El Usuario introduce los nuevos valores.
5. El sistema valida el formato de los nuevos valores.
6. El sistema solicita confirmación.
7. El Usuario confirma el cambio.
8. El sistema guarda la nueva información del Usuario.
9. El sistema informa al Usuario de que la operación se ha realizado con éxito.
10. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 4.a,7.a) El Usuario cancela el proceso de editar información de su cuenta y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 5.a) El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 4.

Tabla 3.4: Descripción del Caso de Uso *Editar información del Usuario*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-5 Ver publicaciones usuarios seguidos
<b>Descripción:</b>	El Usuario desea ver las últimas publicaciones de los usuarios a los que sigue.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario podrá ver las publicaciones de los usuarios a los que sigue ordenadas por fecha de publicación mostrando las más recientes primero.

### Secuencia normal

---

1. El Usuario selecciona ver las últimas publicaciones de los usuarios a los que sigue.
2. El sistema busca los usuarios a los que sigue.
3. El sistema busca las publicaciones de los usuarios a los que sigue.
4. El sistema ordena las publicaciones por fecha de publicación colocando las más recientes primero.
5. El sistema muestra las publicaciones de los usuarios a los que sigue ordenadas por fecha de publicación al Usuario.
6. El Caso de Uso finaliza.

---

### Excepciones

---

- 2.a)** El sistema detecta que el Usuario no sigue a ningún otro usuario, el sistema informa al Usuario, el Caso de Uso finaliza.
- 3.a)** El sistema detecta que no hay publicaciones de los usuarios a los que sigue, el sistema informa al Usuario, el Caso de Uso finaliza.

Tabla 3.5: Descripción del Caso de Uso *Ver publicaciones usuarios seguidos*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-6 Crear receta
<b>Descripción:</b>	El Usuario desea registrar una receta en la aplicación para que así todos los usuarios la puedan ver y puedan realizarla.
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario se ha identificado en el sistema.
<b>Post-condiciones:</b>	La receta se ha guardado en el sistema y su información es accesible por todos los usuarios de la aplicación.

#### Secuencia normal

---

1. El Usuario selecciona crear una receta.
2. El sistema solicita título, resumen, tiempo de preparación, foto y dificultad.
3. El Usuario introduce los datos solicitados.
4. El sistema valida el formato de los datos introducidos.
5. El sistema solicita los pasos para realizar la receta.
6. El Usuario introduce los pasos para realizar la receta.
7. El sistema valida que el formato de los pasos introducidos sea correcto.
8. El sistema solicita los alimentos para la preparación de la receta.

#### Punto de inclusión: CU-7 Buscar alimento.

10. El Usuario selecciona uno de alimentos encontrados en la búsqueda.
11. El sistema solicita la cantidad de ese alimento utilizada en la preparación de la receta.
12. El Usuario introduce la cantidad del alimento necesaria para la preparación de la receta.
13. El sistema valida que el formato de la cantidad introducida sea correcta.
14. El sistema solicita confirmación para crear la receta.
15. El Usuario confirma la creación de la receta.
16. El sistema guarda la información de la receta.
17. El Caso de Uso finaliza.

#### Secuencias alternativas

---

**3.a,6.a,10.a,12.a,15.a)** El Usuario cancela el proceso de crear receta y el Caso de Uso queda sin efecto.

**15.b)** El Usuario desea añadir más alimentos para la preparación de la receta, el Caso de Uso continúa en 8.

#### Excepciones

---

**4.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

**7.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 6.

**13.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 12.

<b>Caso de uso:</b>	CU-7 Buscar alimento
<b>Descripción:</b>	Un Usuario pretende encontrar un determinado alimento de la aplicación buscando por su nombre.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario obtiene una lista de alimentos con nombre similar al realizado en la búsqueda.

---

#### Secuencia normal

1. El Usuario inicia el proceso de búsqueda de alimentos.
2. El sistema solicita al Usuario que introduzca el nombre del alimento a buscar.
3. El Usuario introduce el nombre del alimento a buscar.
4. El sistema valida el nombre introducido por el Usuario.
5. El sistema busca los alimentos en el sistema con el nombre introducido.
6. El sistema muestra los resultados de la búsqueda al Usuario.
7. El Caso de Uso finaliza.

---

#### Secuencias alternativas

- 3.a)** El Usuario cancela el proceso de buscar alimento y el Caso de Uso queda sin efecto.

---

#### Excepciones

- 4.a)** El sistema comprueba que el formato del dato introducido no es correcto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.
- 5.a)** El sistema no encuentra ningún resultado que satisfaga la búsqueda, el sistema informa al Usuario, el Caso de Uso finaliza.

---

Tabla 3.7: Descripción del Caso de Uso *Buscar alimento*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-8 Buscar receta
<b>Descripción:</b>	Un Usuario pretende encontrar una determinada receta de la aplicación buscando por su título de entre todas las recetas registradas en la aplicación.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario obtiene una lista de recetas con título similar al realizado en la búsqueda.

#### Secuencia normal

---

1. El Usuario inicia el proceso de búsqueda de recetas.
2. El sistema solicita al Usuario que introduzca el título de la receta a buscar.
3. El Usuario introduce el título de la receta a buscar.
4. El sistema valida el título introducido por el Usuario.
5. El sistema busca las recetas en el sistema con el título introducido.
6. El sistema muestra los resultados de la búsqueda al Usuario.
7. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 3.a)** El Usuario cancela el proceso de búsqueda de recetas y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 4.a)** El sistema comprueba que el formato del dato introducido no es correcto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.
- 5.a)** El sistema no encuentra ningún resultado que satisfaga la búsqueda, el sistema informa al Usuario, el Caso de Uso finaliza.

Tabla 3.8: Descripción del Caso de Uso *Buscar receta*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-9 Crear publicación
<b>Descripción:</b>	El Usuario desea guardar una publicación en la aplicación para que así todos los usuarios la puedan ver.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario se ha identificado en el sistema.
<b>Post-condiciones:</b>	La publicación se ha guardado en el sistema y es visible por todos los usuarios de la aplicación.

### Secuencia normal

---

1. El Usuario selecciona crear una publicación.
2. El sistema solicita título, descripción y foto.
3. El Usuario introduce los datos solicitados.
4. El sistema valida el formato de los datos introducidos.
5. El sistema solicita si la publicación se va a enlazar con una receta o un alimento.
6. El Usuario introduce si la publicación se va a enlazar con una receta o un alimento.
7. Si el Usuario selecciona receta.

#### Punto de extensión: CU-8 Buscar receta.

8. El Usuario selecciona una de las recetas encontradas en la búsqueda.
9. El sistema solicita confirmación para crear la publicación.
10. El Usuario confirma la creación de la publicación.
11. El sistema guarda la publicación.
12. El Caso de Uso finaliza.

---

### Secuencias alternativas

**3.a,6.a,8.a,10.a)** El Usuario cancela el proceso de crear publicación y el Caso de Uso queda sin efecto.

**7.a)** Si el Usuario selecciona alimento. **Punto de extensión: CU-7 Buscar alimento.** El Usuario selecciona uno de los alimentos encontrados en la búsqueda, el Caso de Uso continúa en 9.

---

### Excepciones

**4.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

---

Tabla 3.9: Descripción del Caso de Uso *Crear publicación*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-10 Buscar Usuario
<b>Descripción:</b>	El Usuario pretende encontrar un determinado Usuario de la aplicación buscando por su <i>username</i> .
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario obtiene una lista de usuarios con <i>username</i> similar al realizado en la búsqueda.

#### Secuencia normal

---

1. El Usuario inicia el proceso de búsqueda de usuarios.
2. El sistema solicita al Usuario que introduzca el *username* del Usuario a buscar.
3. El Usuario introduce el *username* del Usuario a buscar.
4. El sistema valida el *username* introducido por el Usuario.
5. El sistema busca los usuarios en el sistema con el *username* introducido.
6. El sistema muestra los resultados de la búsqueda al Usuario.
7. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 3.a)** El Usuario cancela el proceso de búsqueda de usuarios y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 4.a)** El sistema comprueba que el formato del dato introducido no es correcto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.
- 5.a)** El sistema no encuentra ningún resultado que satisfaga la búsqueda, el sistema informa al Usuario, el Caso de Uso finaliza.

Tabla 3.10: Descripción del Caso de Uso *Buscar usuario*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-11 Añadir receta a favoritas
<b>Descripción:</b>	El Usuario desea guardar una receta como favorita para así poder acceder a ella más rápidamente.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	La lista de recetas favoritas del Usuario incluye la nueva receta favorita.

### Secuencia normal

---

1. El Usuario selecciona guardar la receta como favorita.
  2. El sistema guarda la receta en la lista de recetas favoritas del Usuario.
  3. El sistema informa al Usuario.
  4. El Caso de Uso finaliza.
- 

Tabla 3.11: Descripción del Caso de Uso *Añadir receta a favoritas*.

<b>Caso de uso:</b>	CU-12 Consultar recetas favoritas
<b>Descripción:</b>	El Usuario desea ver su lista de recetas favoritas.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario podrá ver la lista de recetas que ha marcado como favoritas.

### Secuencia normal

---

1. El Usuario selecciona ver sus recetas favoritas.
2. El sistema busca sus recetas favoritas.
3. El sistema muestra sus recetas favoritas. Si el Usuario selecciona eliminar una receta de favoritas.

**Punto de extensión: CU-13 Borrar receta de favoritas.**

4. El Caso de Uso finaliza.
- 

### Excepciones

---

- 2.a)** El sistema detecta que el Usuario no tiene ninguna receta favorita, el sistema informa al Usuario, el Caso de Uso finaliza.
- 

Tabla 3.12: Descripción del Caso de Uso *Consultar recetas favoritas*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-13 Borrar receta de favoritas
<b>Descripción:</b>	El Usuario decide que una de las recetas que ha marcado como favoritas ya no la quiere guardar en su lista de favoritas.
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema y se ha ejecutado el Caso de Uso CU-12 Consultar recetas favoritas.
<b>Post-condiciones:</b>	La lista de recetas favoritas del Usuario no contará con la receta eliminada de favoritas.

#### Secuencia normal

---

1. El Usuario selecciona la receta a eliminar de sus recetas favoritas.
  2. El sistema elimina la receta de su lista de recetas favoritas.
  3. El sistema informa al Usuario.
  4. El Caso de Uso finaliza.
- 

Tabla 3.13: Descripción del Caso de Uso *Borrar receta de favoritas*.

<b>Caso de uso:</b>	CU-14 Consultar usuarios seguidos
<b>Descripción:</b>	El Usuario desea ver los usuarios que sigue.
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	El Usuario podrá ver la lista de usuarios que sigue.

#### Secuencia normal

---

1. El Usuario selecciona ver los usuarios que sigue.
2. El sistema busca los usuarios que sigue.
3. El sistema muestra los usuarios que sigue. Si el Usuario selecciona dejar de seguir a un Usuario.

**Punto de extensión: CU-15 Dejar de seguir Usuario.**

4. El Caso de Uso finaliza.
- 

#### Excepciones

---

- 2.a) El sistema detecta que el Usuario no tiene ningún Usuario que sigue, el sistema informa al Usuario, el Caso de Uso finaliza.
- 

Tabla 3.14: Descripción del Caso de Uso *Consultar usuarios seguidos*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-15 Dejar de seguir Usuario
<b>Descripción:</b>	El Usuario decide que ya no quiere continuar siguiendo uno de los usuarios a los que sigue.
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema y se ha ejecutado el Caso de Uso CU-14 Consultar usuarios seguidos.
<b>Post-condiciones:</b>	La lista de usuarios seguidos por el Usuario no incluye el Usuario que se ha dejado de seguir.

### Secuencia normal

- 
1. El Usuario selecciona el Usuario que quiere dejar de seguir.
  2. El sistema elimina el Usuario de los usuarios a los que sigue.
  3. El sistema informa al Usuario.
  4. El Caso de Uso finaliza.

Tabla 3.15: Descripción del Caso de Uso *Dejar de seguir Usuario*.

<b>Caso de uso:</b>	CU-16 Seguir Usuario
<b>Descripción:</b>	El Usuario desea seguir a un Usuario para conocer sus publicaciones.
<b>Actores:</b>	Usuario.
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	La lista de usuarios seguidos del Usuario incluye el nuevo Usuario.

### Secuencia normal

- 
1. El Usuario selecciona seguir al Usuario.
  2. El sistema guarda el Usuario en la lista de usuarios que sigue el Usuario.
  3. El sistema informa al Usuario.
  4. El Caso de Uso finaliza.

Tabla 3.16: Descripción del Caso de Uso *Seguir Usuario*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-17 Añadir comentario a publicación
<b>Descripción:</b>	El Usuario desea guardar un comentario en una publicación.
<b>Actores:</b>	Usuario
<b>Pre-condiciones:</b>	El Usuario está identificado en el sistema.
<b>Post-condiciones:</b>	La publicación tiene el nuevo comentario guardado.

#### Secuencia normal

---

1. El Usuario selecciona crear un comentario.
2. El sistema solicita el comentario a guardar.
3. El Usuario introduce el comentario.
4. El sistema valida el contenido del comentario introducido.
5. El sistema solicita confirmación.
6. El Usuario confirma la acción para que se guarde el comentario.
7. El sistema guarda el comentario de la publicación.
8. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 3.a,6.a)** El Usuario cancela el proceso de crear comentario y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 4.a)** El sistema comprueba que el contenido del comentario es incorrecto, el sistema informa al Usuario, el Caso de Uso continúa en el paso 3.

Tabla 3.17: Descripción del Caso de Uso *Añadir comentario a publicación*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-18 Eliminar Usuario
<b>Descripción:</b>	El Administrador decide que un Usuario registrado en la aplicación debe ser eliminado.
<b>Actores:</b>	Administrador.
<b>Pre-condiciones:</b>	El Administrador está identificado en el sistema.
<b>Post-condiciones:</b>	El Administrador ha eliminado el Usuario y ya no aparecerá en la aplicación.

### Secuencia normal

---

1. El Administrador selecciona el Usuario a eliminar.
2. El sistema solicita confirmación de la acción.
3. El Administrador confirma que quiere eliminar ese usuario.
4. El sistema elimina el usuario.
5. El sistema elimina la lista de seguidores que le siguen, la lista de seguidos que sigue, sus recetas favoritas, sus comentarios realizados, sus recetas creadas y sus publicaciones creadas.
6. El sistema informa al Administrador.
7. El Caso de Uso finaliza.

### Secuencias alternativas

---

- 3.a)** El Administrador cancela el proceso de eliminar Usuario y el Caso de Uso queda sin efecto.
- 

Tabla 3.18: Descripción del Caso de Uso *Eliminar Usuario*.

### 3.7. DESCRIPCIÓN CASOS DE USO

<b>Caso de uso:</b>	CU-19 Eliminar publicación
<b>Descripción:</b>	El Administrador decide que una publicación mostrada en la aplicación y creada por un Usuario debe ser eliminada.
<b>Actores:</b>	Administrador.
<b>Pre-condiciones:</b>	El Administrador está identificado en el sistema.
<b>Post-condiciones:</b>	El Administrador ha eliminado la publicación y ya no aparecerá en la aplicación.

#### **Secuencia normal**

- 1.** El Administrador selecciona la publicación a eliminar.
- 2.** El sistema solicita confirmación de la acción.
- 3.** El Administrador confirma que quiere eliminar esa publicación.
- 4.** El sistema elimina la publicación.
- 5.** El sistema elimina los comentarios realizados en esa publicación.
- 6.** El sistema informa al Administrador.
- 7.** El Caso de Uso finaliza.

#### **Secuencias alternativas**

- 3.a)** El Administrador cancela el proceso de eliminar publicación y el Caso de Uso queda sin efecto.

Tabla 3.19: Descripción del Caso de Uso *Eliminar publicación*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-20 Eliminar alimento
<b>Descripción:</b>	El Administrador decide que un alimento mostrado en la aplicación debe ser eliminado.
<b>Actores:</b>	Administrador.
<b>Pre-condiciones:</b>	El Administrador está identificado en el sistema.
<b>Post-condiciones:</b>	El Administrador ha eliminado el alimento y ya no aparecerá en la aplicación.

### Secuencia normal

---

1. El Administrador selecciona el alimento a eliminar.
2. El sistema solicita confirmación de la acción.
3. El Administrador confirma que quiere eliminar ese alimento.
4. El sistema elimina el alimento.
5. El sistema elimina las publicaciones enlazadas con este alimento y de las listas de alimentos utilizados en cada receta los lugares donde se utilice este alimento.
6. El sistema informa al Administrador.
7. El Caso de Uso finaliza.

### Secuencias alternativas

---

- 3.a)** El Administrador cancela el proceso de eliminar alimento y el Caso de Uso queda sin efecto.
- 

Tabla 3.20: Descripción del Caso de Uso *Eliminar alimento*.

### **3.7. DESCRIPCIÓN CASOS DE USO**

---

<b>Caso de uso:</b>	CU-21 Editar información de alimento
<b>Descripción:</b>	El Administrador desea cambiar la información de un alimento.
<b>Actores:</b>	Administrador
<b>Pre-condiciones:</b>	El Administrador se ha identificado en el sistema.
<b>Post-condiciones:</b>	La información del alimento se habrá actualizado con los nuevos campos introducidos.

#### **Secuencia normal**

---

- 1.** El Administrador selecciona editar la información de un alimento.
- 2.** El sistema muestra la información del alimento.
- 3.** El sistema solicita los nuevos valores para estos campos.
- 4.** El Administrador introduce los nuevos valores.
- 5.** El sistema valida el formato de los datos de los nuevos valores.
- 6.** El sistema solicita confirmación.
- 7.** El Administrador confirma el cambio.
- 8.** El sistema guarda la nueva información del alimento.
- 9.** El sistema informa al Administrador de que la operación se ha realizado con éxito.
- 10.** El Caso de Uso finaliza.

#### **Secuencias alternativas**

---

- 4.a,7.a)** El Administrador cancela el proceso de editar información del alimento y el Caso de Uso queda sin efecto.

#### **Excepciones**

---

- 5.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Administrador, el Caso de Uso continúa en el paso 4.

Tabla 3.21: Descripción del Caso de Uso *Editar información de alimento*.

## CAPÍTULO 3. REQUISITOS

---

<b>Caso de uso:</b>	CU-22 Eliminar receta
<b>Descripción:</b>	El Administrador decide que una receta mostrada en la aplicación y creada por un Usuario debe ser eliminada.
<b>Actores:</b>	Administrador.
<b>Pre-condiciones:</b>	El Administrador está identificado en el sistema.
<b>Post-condiciones:</b>	El Administrador ha eliminado la receta y ya no aparecerá en la aplicación.

### Secuencia normal

---

1. El Administrador selecciona la receta a eliminar.
2. El sistema solicita confirmación de la acción.
3. El Administrador confirma que quiere eliminar esa receta.
4. El sistema elimina la receta.
5. El sistema elimina las publicaciones enlazadas con esa receta, elimina de la lista de recetas favoritas de cada Usuario esta receta y elimina los alimentos que se utilizan como ingredientes para la receta.
6. El sistema informa al Administrador.
7. El Caso de Uso finaliza.

### Secuencias alternativas

---

- 3.a)** El Administrador cancela el proceso de eliminar receta y el Caso de Uso queda sin efecto.
- 

Tabla 3.22: Descripción del Caso de Uso *Eliminar receta*.

### 3.7. DESCRIPCIÓN CASOS DE USO

---

<b>Caso de uso:</b>	CU-23 Crear alimento
<b>Descripción:</b>	El Administrador desea guardar un alimento en la aplicación para que así todos los usuarios lo puedan ver.
<b>Actores:</b>	Administrador.
<b>Pre-condiciones:</b>	El Administrador se ha identificado en el sistema.
<b>Post-condiciones:</b>	El alimento se ha guardado en el sistema y es visible por todos los usuarios de la aplicación.

#### Secuencia normal

---

1. El Administrador selecciona crear un alimento.
2. El sistema solicita nombre, descripción, calorías, foto, enlace, grasas, carbohidratos, proteínas, cantidad y medida.
3. El Administrador introduce los datos solicitados.
4. El sistema valida que el formato de los datos introducidos sea correcto.
5. El sistema solicita confirmación para crear el alimento.
6. El Administrador confirma la creación del alimento.
7. El sistema guarda el alimento.
8. El Caso de Uso finaliza.

#### Secuencias alternativas

---

- 3.a,6.a)** El Administrador cancela el proceso de crear alimento y el Caso de Uso queda sin efecto.

#### Excepciones

---

- 4.a)** El sistema comprueba que el formato de los datos introducidos es incorrecto, el sistema informa al Administrador, el Caso de Uso continúa en el paso 3.

Tabla 3.23: Descripción del Caso de Uso *Crear alimento*.

# Capítulo 4

## Análisis

### 4.1. Introducción

En este capítulo se exponen tanto el Modelo del Dominio como el Modelo de Análisis del proyecto, ambos acompañados de sus correspondientes diagramas. Finalmente, se detalla la Realización de Casos de Análisis mediante Diagramas de Secuencia para los principales Casos de Uso del sistema.

### 4.2. Modelo del Dominio

El Modelo de Dominio [25] es un modelo conceptual que tiene como objetivo representar el vocabulario y los conceptos más importantes del dominio del problema. En él se describen las entidades, sus atributos, sus papeles y relaciones junto a las restricciones que rigen el dominio del sistema.

#### 4.2.1. Diagrama de Clases del Modelo del Dominio

En la Figura 4.1 se muestra el Diagrama de Clases del Modelo de Dominio.

### 4.3. Modelo de Análisis

El Modelo de Análisis [26] se utiliza para describir la estructura de la aplicación o el sistema que se está creando. Incluye el Diagrama del Modelo de Dominio con los métodos asociados a cada clase junto a los diagramas de la Realización de Casos de Uso de Análisis que

#### 4.3. MODELO DE ANÁLISIS

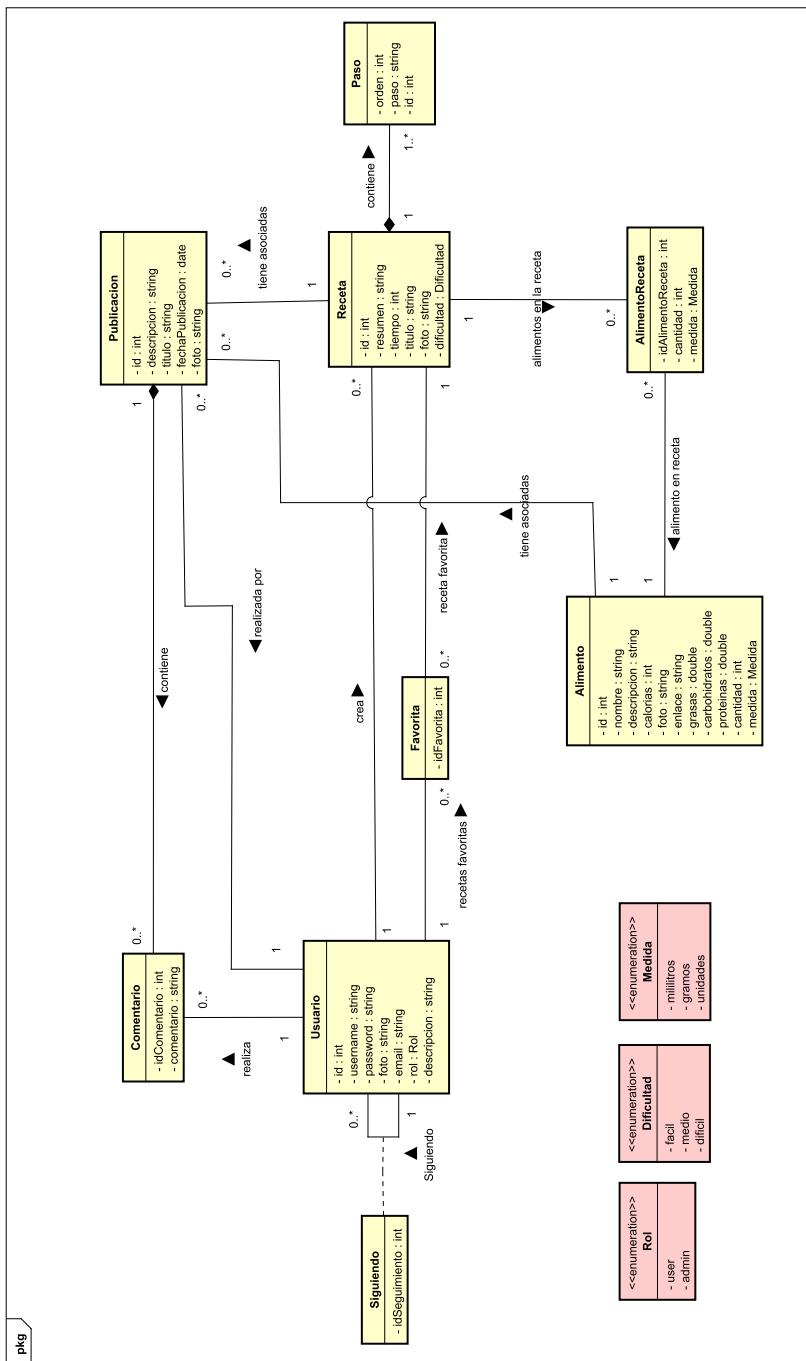


Figura 4.1: Diagrama de Clases del Modelo de Dominio

pueden ser Diagramas de Secuencia. Por lo tanto extiende el Modelo de Dominio y muestra el comportamiento del sistema.

### 4.3.1. Clases de Análisis

En la Figura 4.2 se muestra el Diagrama de Clases de Análisis.

## 4.4. Realización de Casos de Uso de Análisis

La Realización de Casos de Uso de Análisis sirve para describir cómo se realizará un Caso de Uso particular incluyendo los métodos, clases y actores que participarán en él. Para ello, se van a utilizar los Diagramas de Secuencia [27] que muestran la interacción entre un conjunto de objetos a lo largo del tiempo en la aplicación.

### 4.4.1. Realización de Caso de Uso *Registrarse*

En la Figura 4.3 se muestra el Diagrama de Secuencia del Caso de Uso *Registrarse*.

### 4.4.2. Realización de Caso de Uso *Crear publicación*

En la Figura 4.4 se muestra el Diagrama de Secuencia del Caso de Uso *Crear publicación*.

### 4.4.3. Realización de Caso de Uso *Crear receta*

En la Figura 4.5 se muestra el Diagrama de Secuencia del Caso de Uso *Crear receta*.

### 4.4.4. Realización de Caso de Uso *Buscar alimento*

En la Figura 4.6 se muestra el Diagrama de Secuencia del Caso de Uso *Buscar alimento*.

## 4.4. REALIZACIÓN DE CASOS DE USO DE ANÁLISIS

---

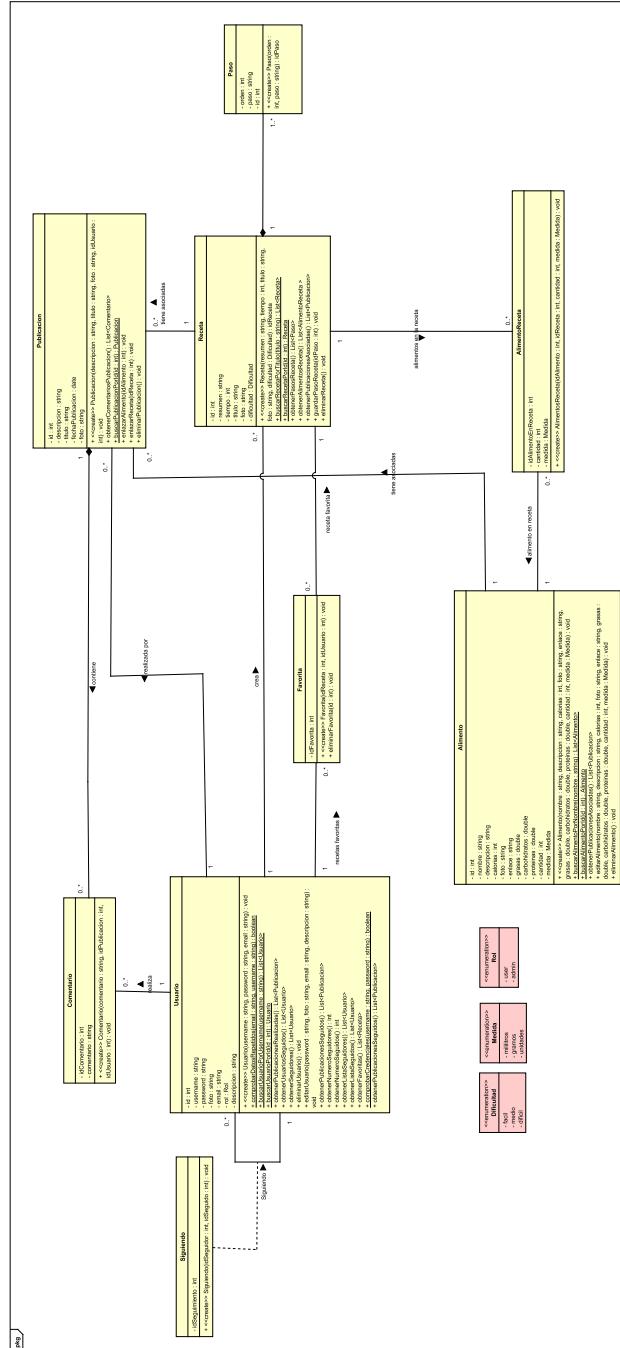


Figura 4.2: Diagrama de Clases de Análisis

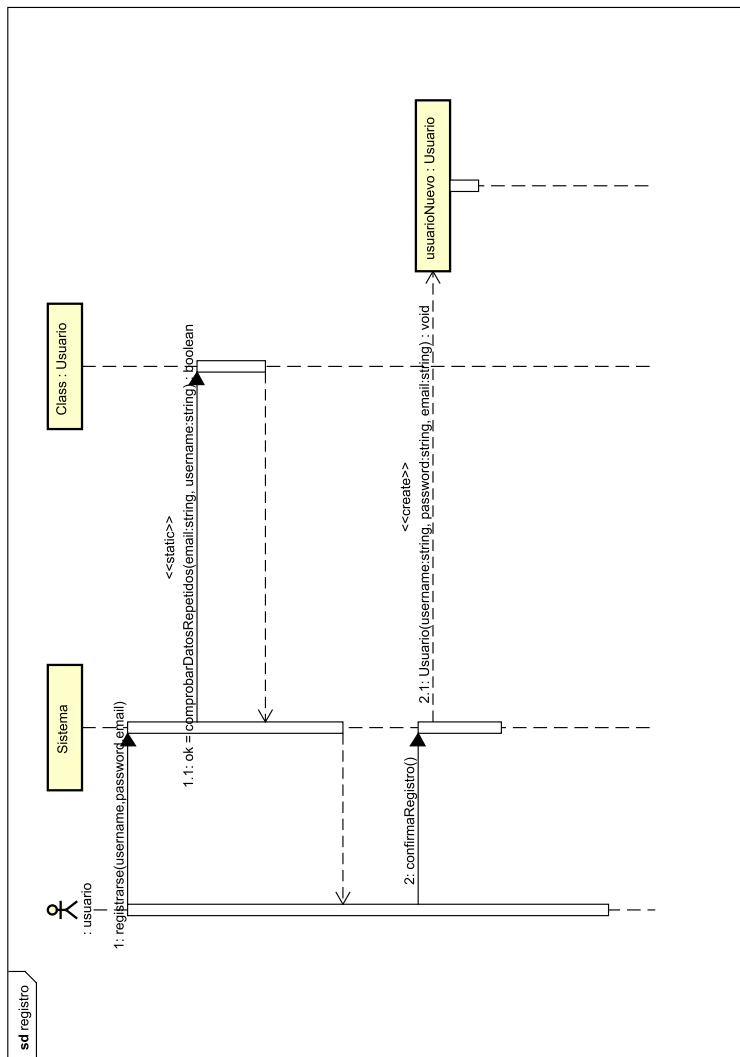


Figura 4.3: Diagrama de Secuencia del Caso de Uso *Registrarse*.

#### 4.4. REALIZACIÓN DE CASOS DE USO DE ANÁLISIS

---

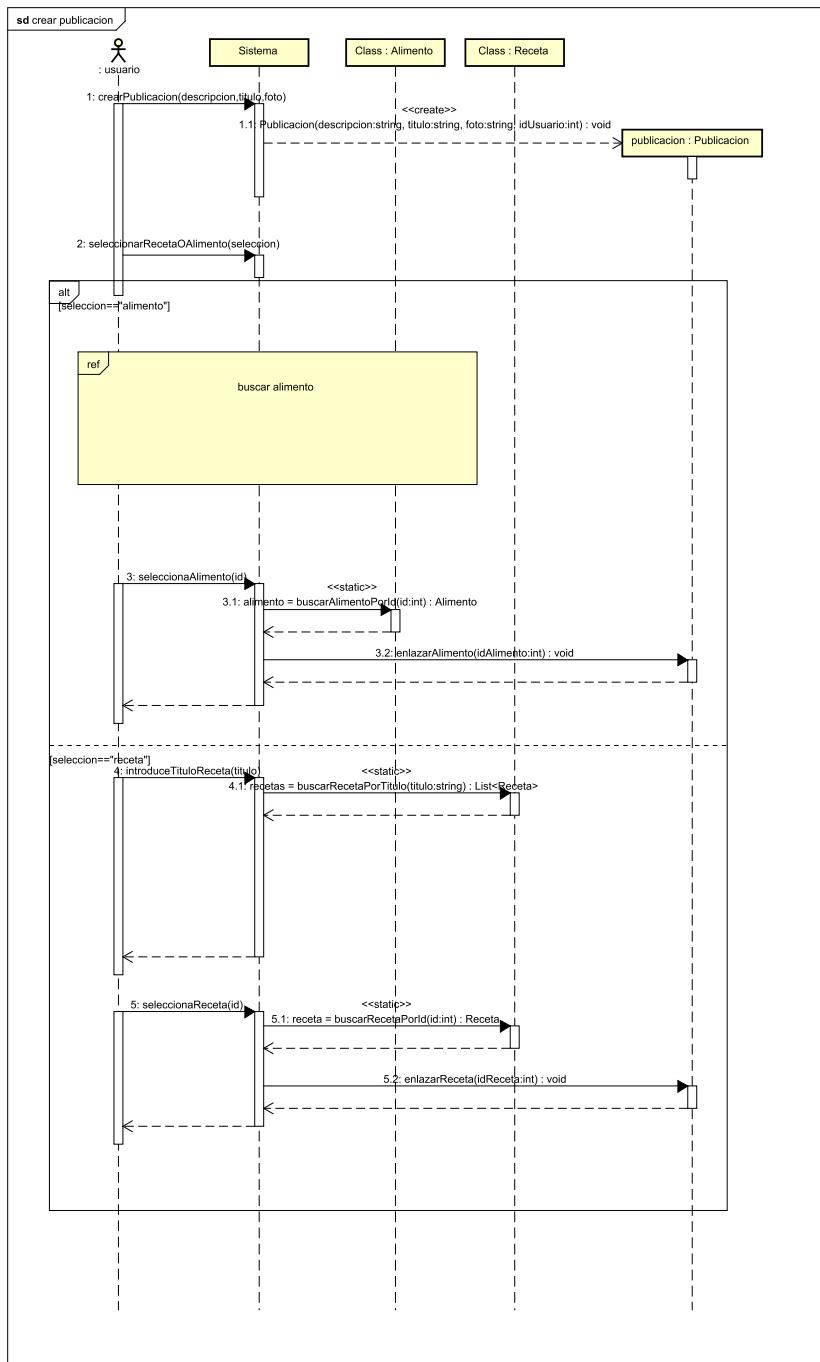


Figura 4.4: Diagrama de Secuencia del Caso de Uso *Crear publicación*.

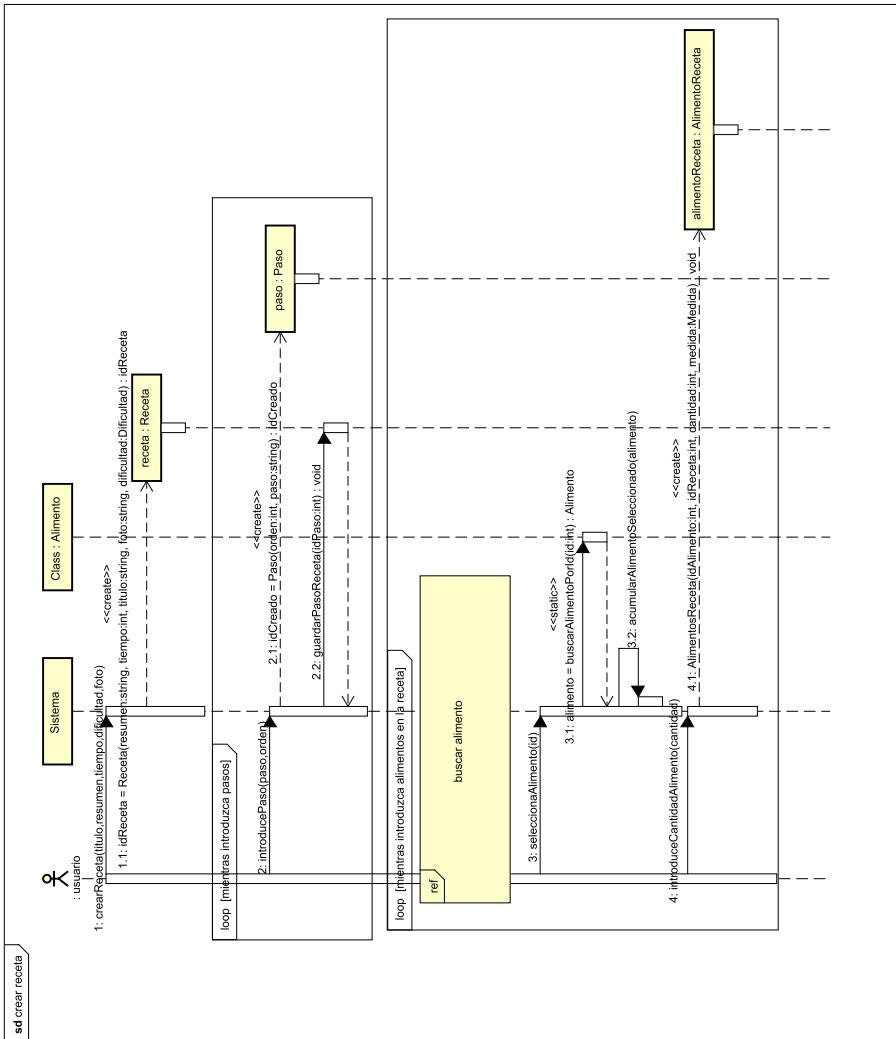


Figura 4.5: Diagrama de Secuencia del Caso de Uso *Crear receta*.

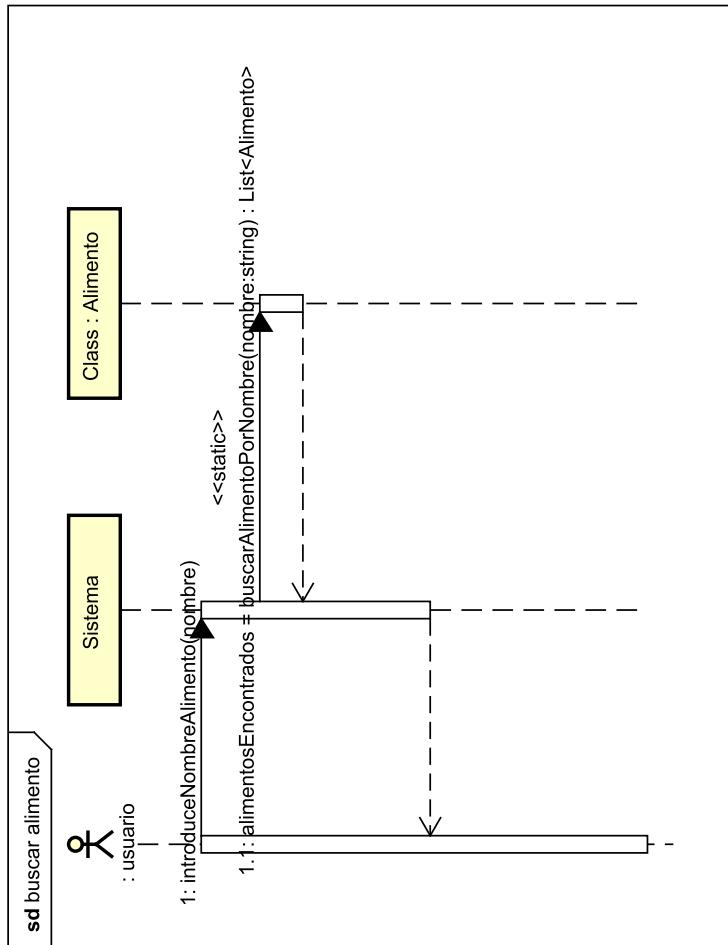


Figura 4.6: Diagrama de Secuencia del Caso de Uso *Buscar alimento*.

# Capítulo 5

## Diseño

### 5.1. Introducción

En este capítulo se explica la Arquitectura Lógica del Sistema, definiendo la Arquitectura del Cliente y la Arquitectura del Servidor junto a los Patrones de Diseño aplicados. Además se define la Arquitectura Física donde se despliega, así como, el Modelo de Datos de la Base de Datos. Por último, se desarrolla el Diseño de la Interfaz Gráfica mediante las interfaces de la aplicación Web con bocetos.

### 5.2. Arquitectura Lógica del Sistema

Se ha utilizado una arquitectura cliente-servidor [28] de tres capas para este sistema. Este estilo arquitectónico está compuesto por una capa de presentación, una capa lógica y una capa de datos. La capa lógica consiste en un servidor que proporciona unos recursos y servicios que son consumidos por la capa de presentación. La capa de presentación es el cliente que se encarga de representar los datos que obtiene del servidor. El cliente se comunica con el servidor mediante peticiones HTTP (*Hypertext Transfer Protocol*) que definen las acciones que el cliente quiere realizar. El servidor realiza esas acciones sobre la capa de datos que es la base de datos y envía al cliente una respuesta.

La arquitectura cliente-servidor utilizada para este proyecto se basa en el patrón arquitectónico en tres capas [29]. La arquitectura en capas permite separar responsabilidades. De esta manera, el servidor será el único que podrá acceder a los datos almacenados en la base de datos para así proteger los datos y facilitar la lógica de su procesamiento. En cambio, el cliente solo se encarga de mostrar los datos que le envía el servidor.

Para este proyecto, el cliente, el servidor y la base de datos están en la misma máquina y están desarrollados en tres partes diferentes. La parte del cliente está desarrollada en Angular

## 5.2. ARQUITECTURA LÓGICA DEL SISTEMA

---

y se corresponde con la parte del *frontend*, la parte del servidor esta desarrollada en NodeJS y se corresponde con la parte del *backend* y la parte de la base de datos se corresponde con la base de datos MySQL.

En la Figura 5.1 se muestra el Diagrama con la Arquitectura Lógica del Sistema.

### 5.2.1. Arquitectura del Cliente

La parte del cliente de la arquitectura del sistema en la capa de presentación corresponde con la parte desarrollada en Angular que se define como el *frontend* de la aplicación. Se ha escogido el patrón Modelo–Vista–Modelo de Vista ya que Angular, que es el *framework* que se ha utilizado para desarrollar el cliente en la parte de *frontend*, propone ese patrón. En la sección 6.3.1 se explica en profundidad Angular.

El patrón Modelo–Vista–Modelo de Vista [30], también conocido como MVVM es un patrón de arquitectura de *software* cuyo objetivo principal es desacoplar lo máximo posible la interfaz del usuario, es decir, lo que el Usuario ve, de la lógica de la aplicación, que son las operaciones que tiene que realizar internamente la aplicación. Este patrón es una variación del patrón MVC. El patrón MVVM consta de tres partes:

- **Modelo:** consiste en la lógica de negocio, por lo tanto, contiene la información con la que trabajará la aplicación. Representará el Modelo de Dominio de la aplicación y las clases del Modelo se pueden usar junto a servicios para encapsular el acceso a los datos.
- **Vista:** representa la información al Usuario definiendo la estructura, diseño y apariencia de lo que ve el Usuario y se encarga de capturar los eventos que produzca el Usuario al interactuar con la aplicación.
- **Modelo de Vista:** es el intermediario entre el Modelo y la Vista y contiene la lógica de presentación. La Vista y el Modelo de Vista se comunican mediante enlaces de datos. Tiene las propiedades y comandos que definirán la funcionalidad que permitirá la interfaz de usuario y que permiten a la Vista enlazar datos. El Modelo de Vista también notifica a la Vista los cambios de estado.

Este patrón permite enlace de datos o *Data Binding*, de manera que los datos están sincronizados de manera automática entre el Modelo y la Vista por medio del Modelo de la Vista. De esta forma, cuando hay un cambio en el Modelo, la Vista cambia y se actualiza, pero también sucede al contrario, cuando hay un cambio en la Vista, el Modelo también se actualizará, ambos casos mediante el intermediario Modelo de Vista [31].

Para el *framework* Angular utilizado para el *frontend*, el patrón MVVM define que la Vista son los archivos HTML ya que serán lo que se muestre al Usuario. El Modelo se corresponde con los objetos creados de las clases del Modelo de Dominio y los servicios. El Modelo de Vista se corresponde con los archivos Typescript, que se encargarán de enlazar los datos con la Vista y notificar a los servicios del Modelo para realizar la comunicación con el *backend*.

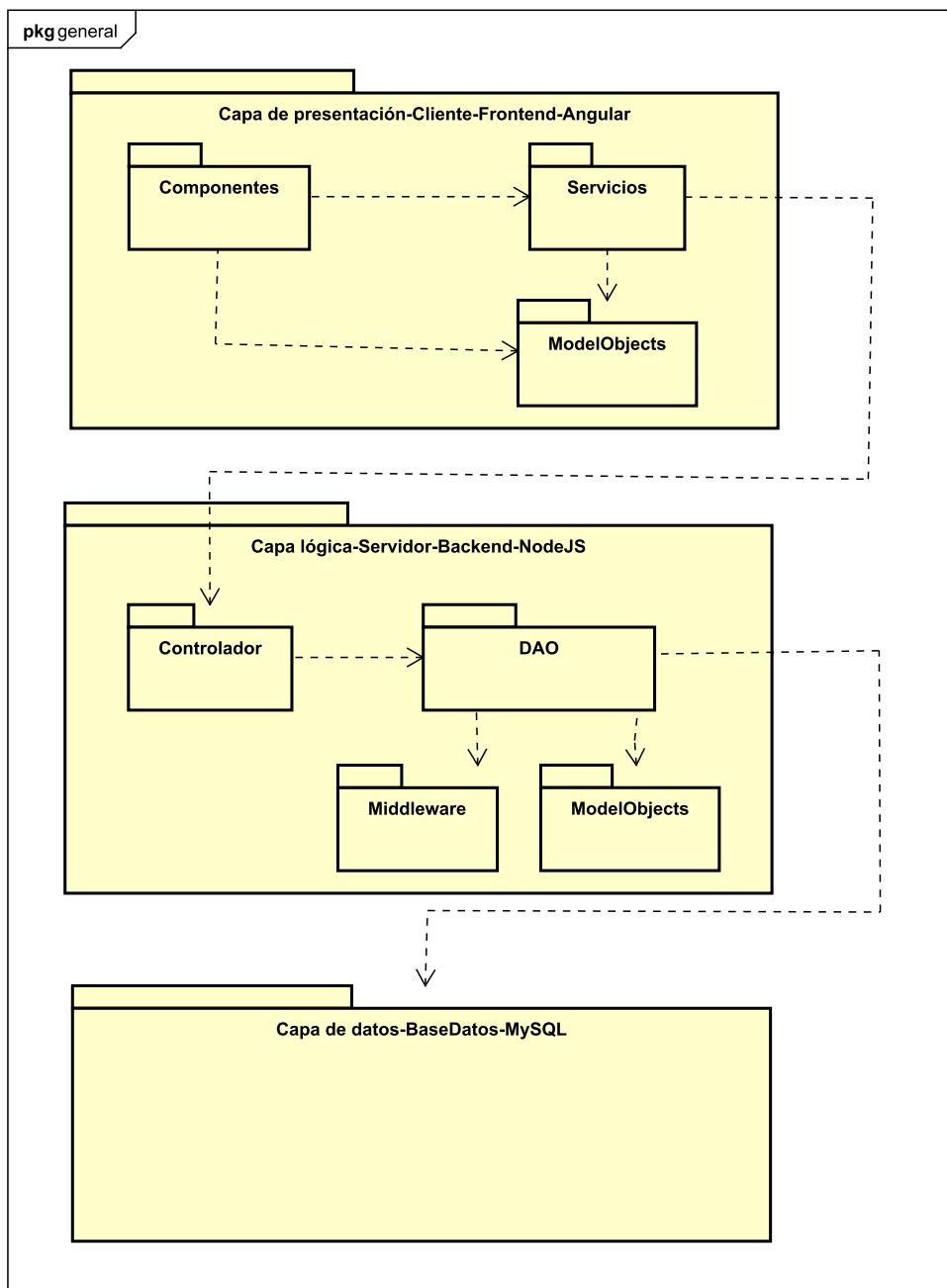


Figura 5.1: Arquitectura Lógica del Sistema.

## 5.2. ARQUITECTURA LÓGICA DEL SISTEMA

---

En la Figura 5.2 se muestran las relaciones entre los componentes del patrón [32].

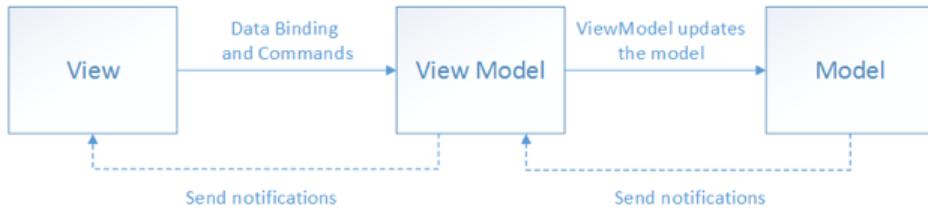


Figura 5.2: Relaciones entre los componentes del patrón MVVM.

El cliente de este proyecto contiene los siguientes paquetes:

- **Componentes:** en este paquete se encuentran los componentes de Angular. Estos componentes se agrupan en otros paquetes según su funcionalidad. Este paquete se corresponde con la Vista y el Modelo de la Vista del patrón MVVM.
- **Servicios:** en este paquete se encuentran los servicios que se comunican con el servidor que es el *backend*. Cada uno de estos servicios manda peticiones HTTP al *backend* para realizar operaciones sobre una entidad del Modelo de Dominio. Este paquete corresponde al Modelo del patrón MVVM.
- **ModelObjects:** en este paquete se encuentran los DTO utilizados para gestionar los datos en el *frontend*. Este paquete también corresponde al Modelo del patrón MVVM.

En la Figura 5.3 se muestra la Arquitectura del Cliente. En las Figuras 5.4 a 5.6 se pueden ver los diagramas *Decomposition&Uses Style* de los paquetes definidos en la Arquitectura del Cliente.

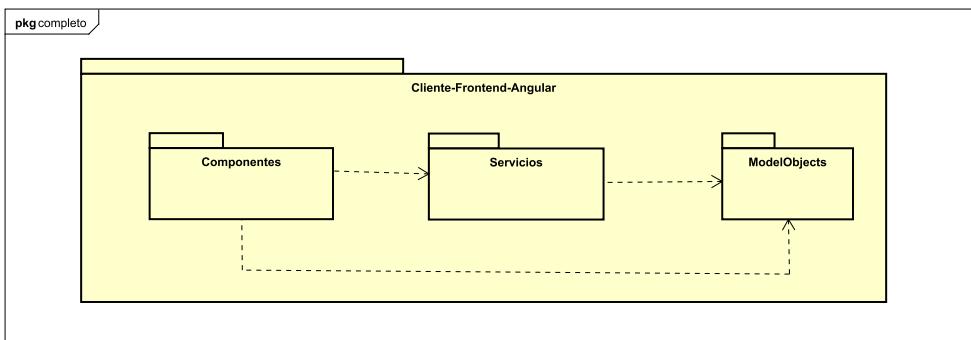


Figura 5.3: Arquitectura del Cliente de la capa de presentación hecho en Angular correspondiente al *frontend*.

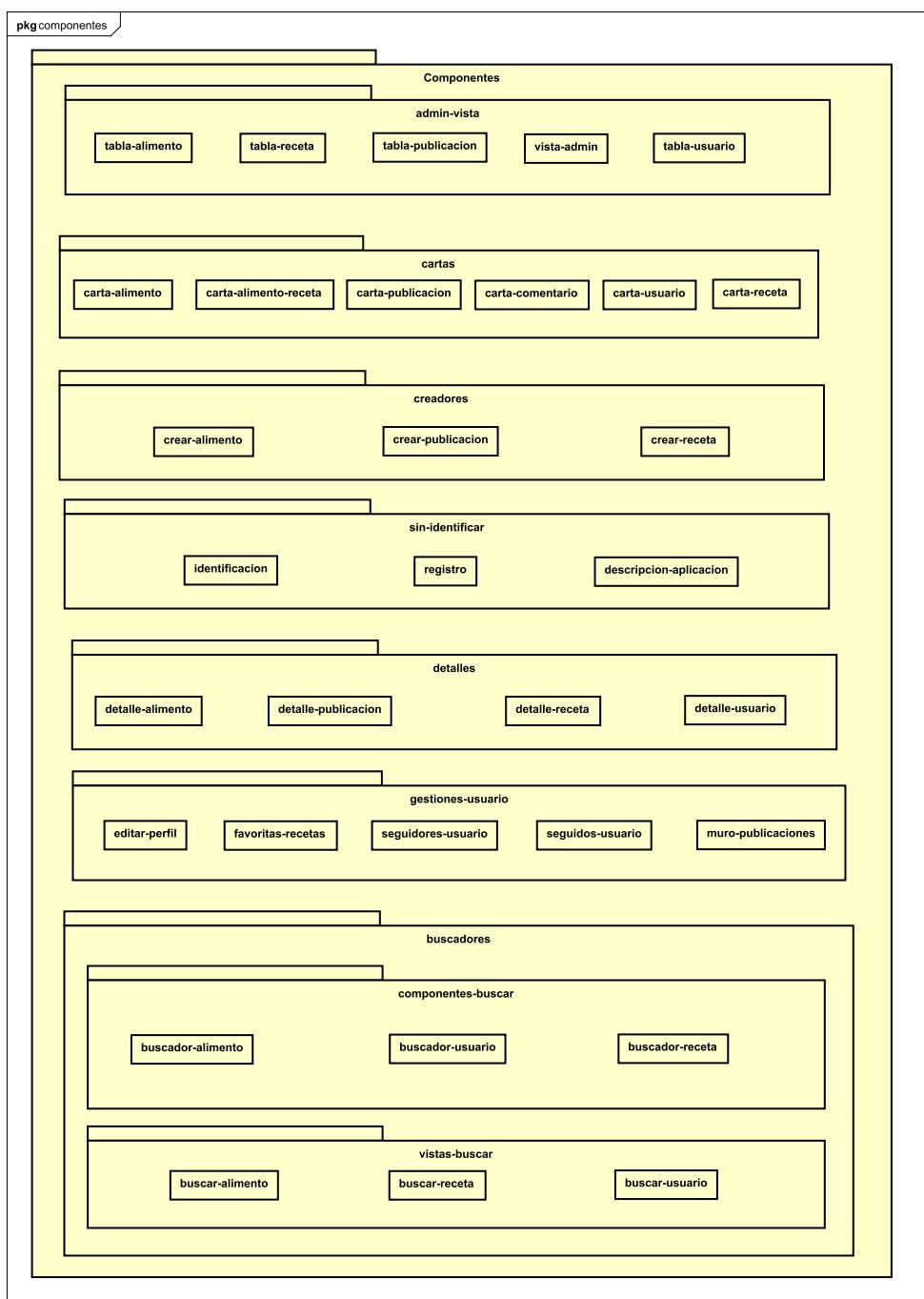


Figura 5.4: *Decomposition&Uses Style* del paquete *Componentes*.

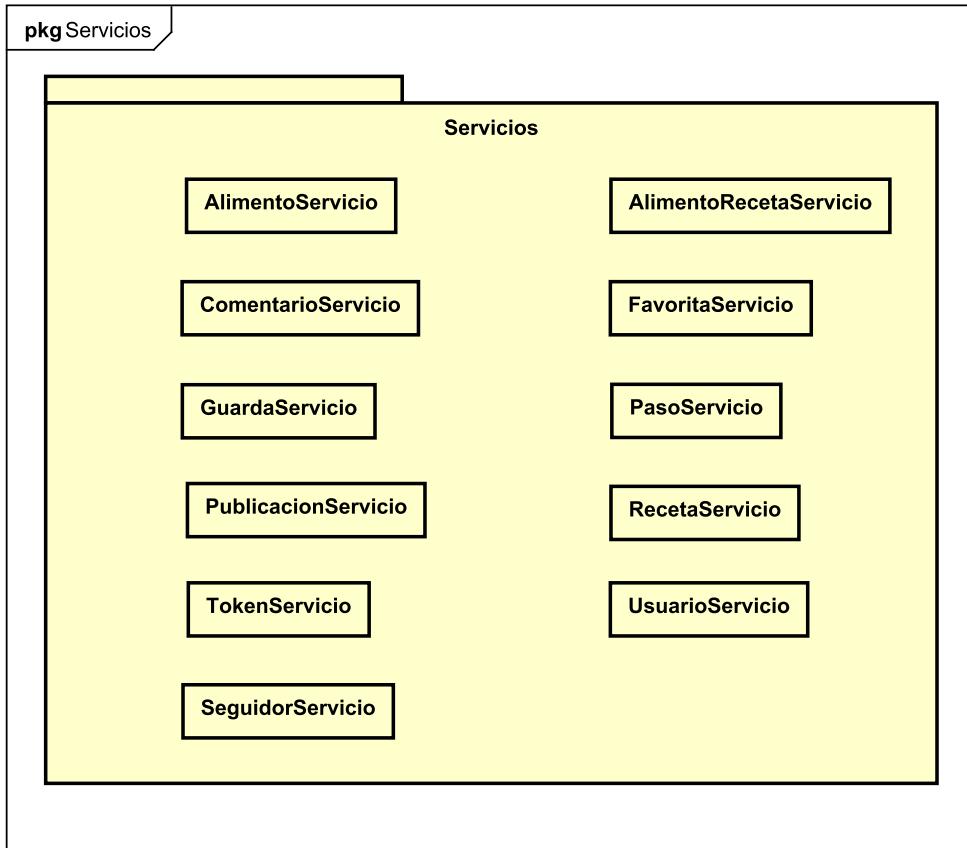


Figura 5.5: *Decomposition&Uses Style* del paquete *Servicios*.

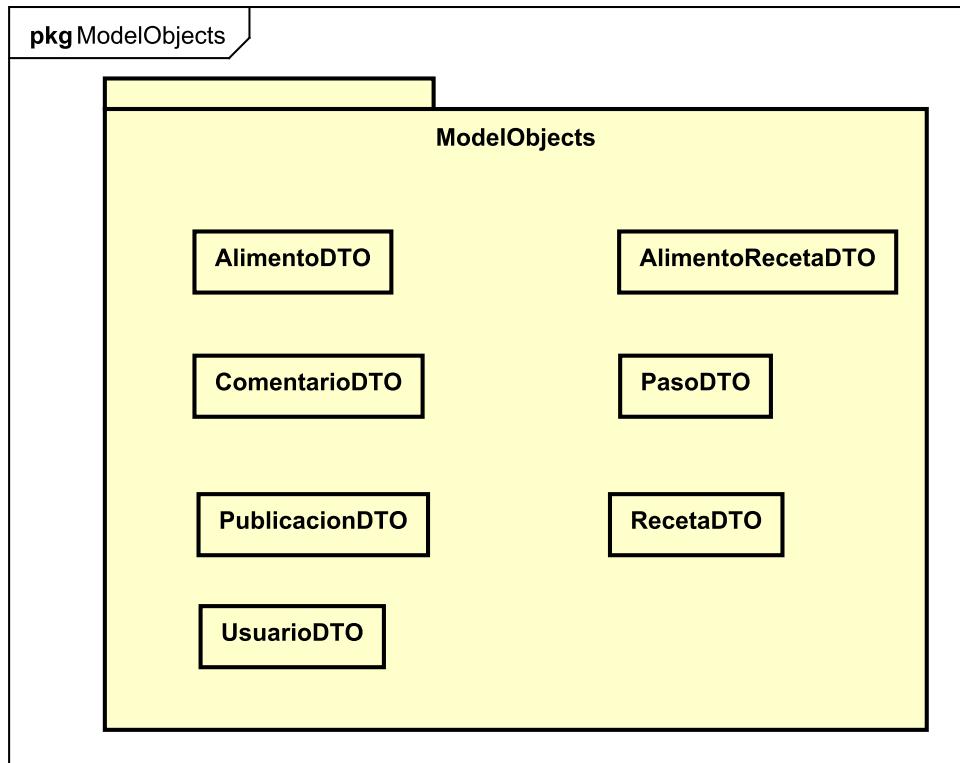


Figura 5.6: *Decomposition&Uses Style* del paquete *ModelObjects*.

### 5.2.2. Arquitectura del Servidor

En el servidor que se corresponde con la parte del *backend* desarrollada en NodeJS en la capa lógica se pueden encontrar los siguientes paquetes:

- **Controlador:** en este paquete se encuentra la clase que se encarga de recibir las peticiones HTTP del cliente y de llamar al DAO que realice la operación solicitada.
- **ModelObjects:** en este paquete se encuentran los DTO utilizados para gestionar los datos en el *backend*.
- **DAO:** en este paquete se encuentran los DAO que realizan las operaciones sobre las tablas de la base de datos.
- **Middleware:** en este paquete se encuentran las clases que se encargarán de procesar las imágenes para almacenarlas en la nube.

En la Figura 5.7 se muestra la Arquitectura del Servidor. En las Figuras 5.8 a 5.11 se pueden ver los diagramas *Decomposition&Uses Style* de los paquetes definidos en la Arquitectura del Servidor.

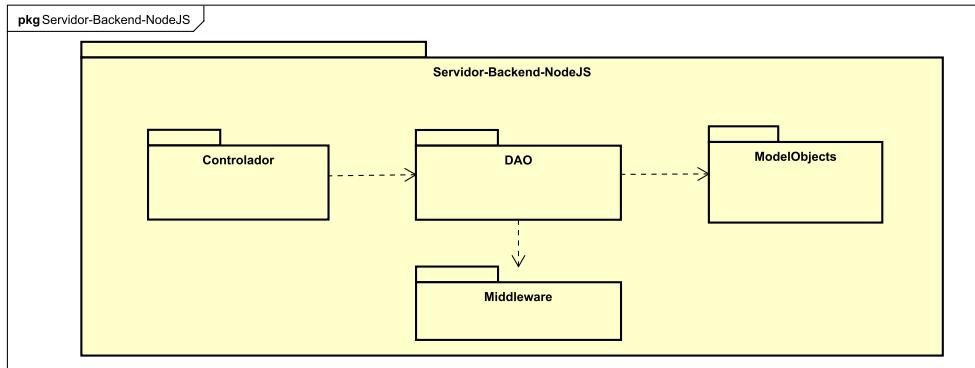


Figura 5.7: Arquitectura del Servidor de la capa lógica hecha en NodeJS correspondiente al *backend*.

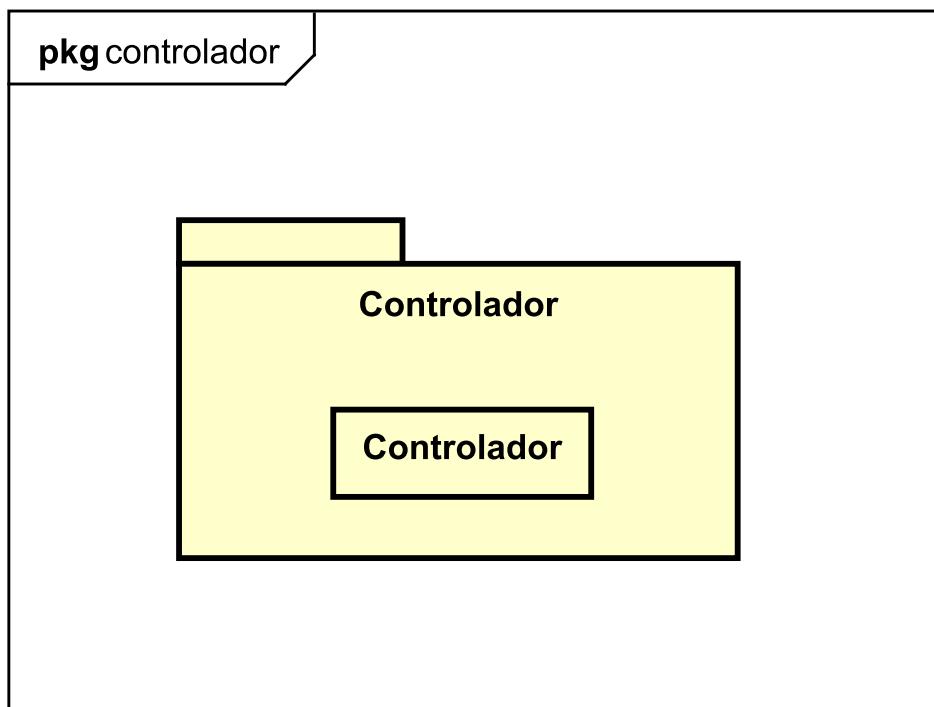


Figura 5.8: *Decomposition&Uses Style* del paquete *Controlador*.

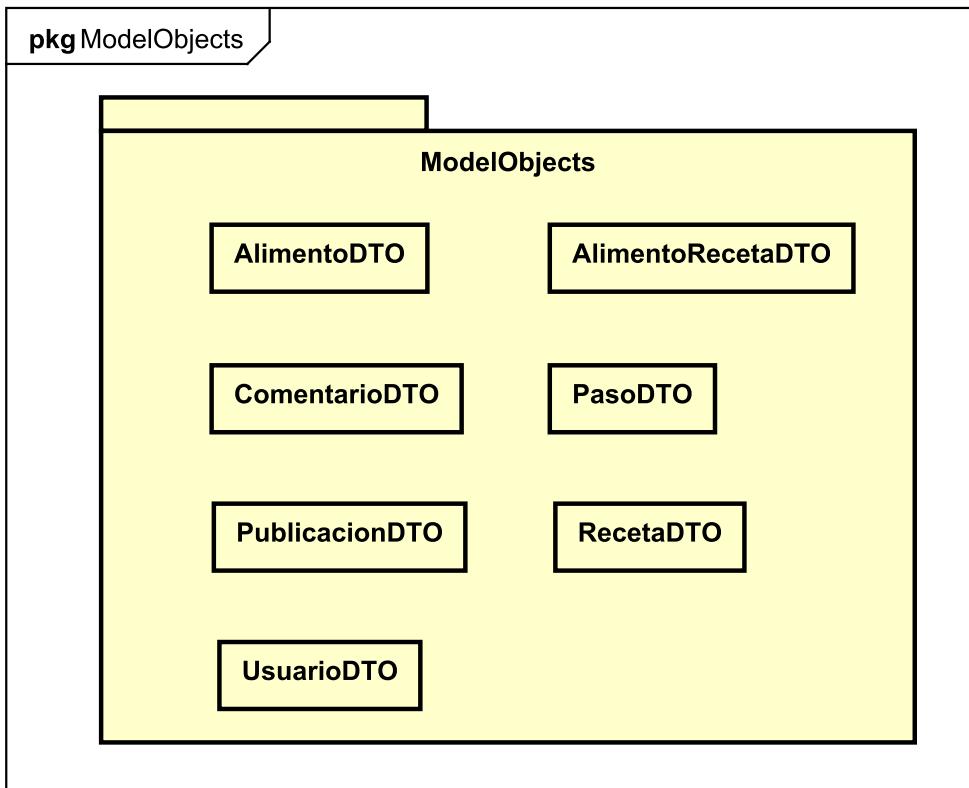


Figura 5.9: *Decomposition&Uses Style* del paquete *ModelObjects*.

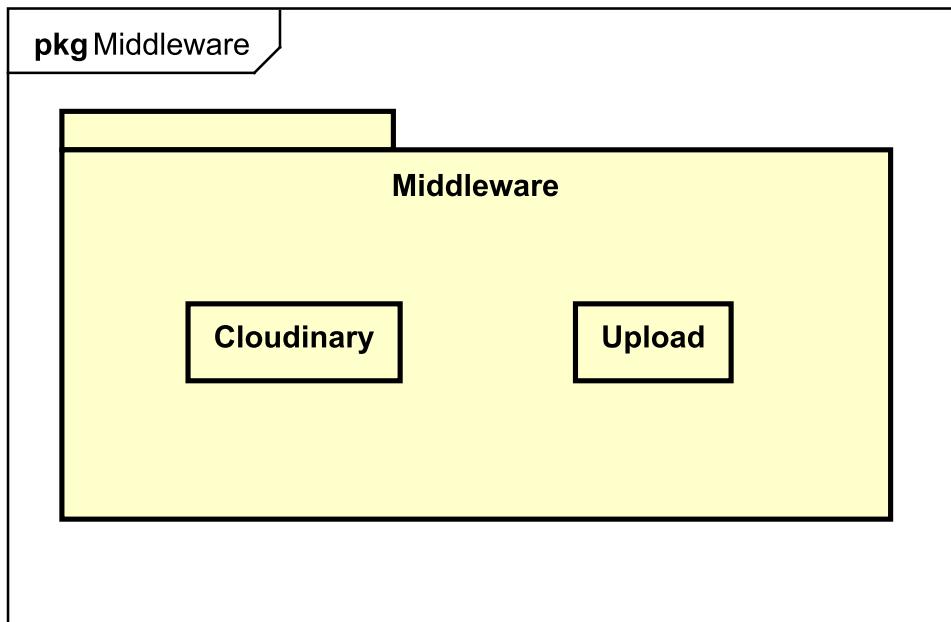


Figura 5.10: *Decomposition&Uses Style* del paquete *Middleware*.

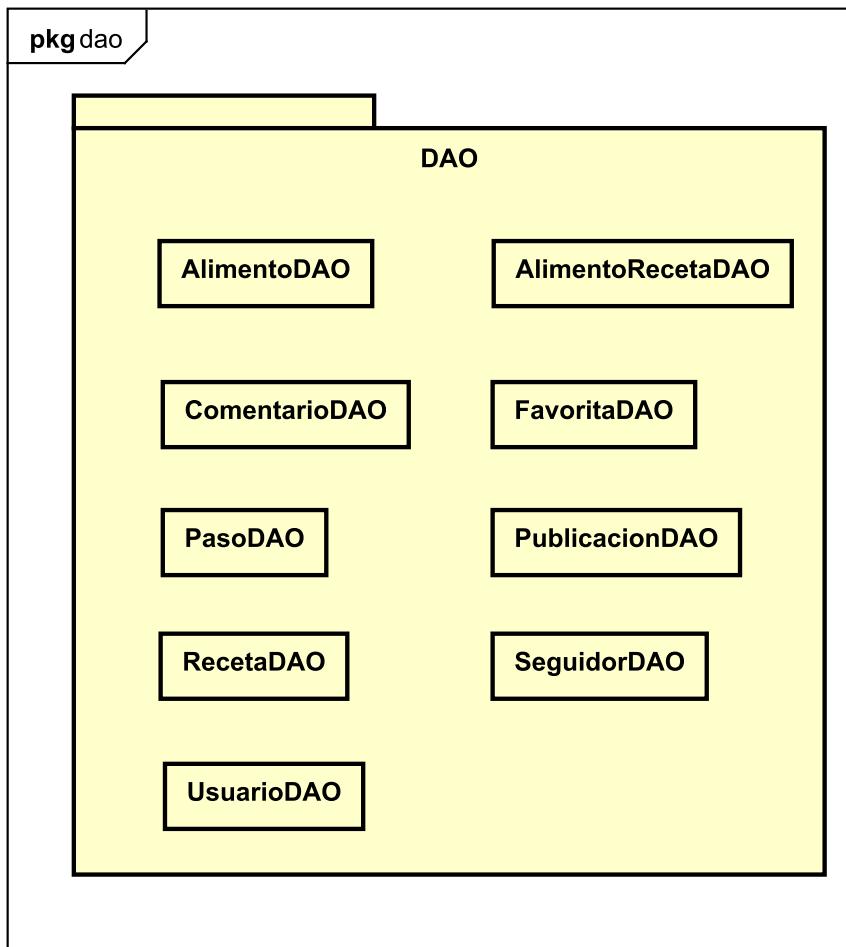


Figura 5.11: *Decomposition&Uses Style* del paquete *DAO*.

### 5.3. Patrones de Diseño

Para el desarrollo y refinamiento de los subsistemas y componentes software del proyecto se han utilizado diferentes Patrones de Diseño para resolver los problemas encontrados y realizar un sistema más robusto y mejor estructurado. A continuación se enumeran los Patrones de Diseño utilizados.

### 5.3.1. Observador

El patrón Observador [33] crea un mecanismo en el que se notifica a varios objetos sobre cualquier cambio que sufra un objeto que están observando. Este patrón es uno de los principales utilizados para interfaces gráficas. Los observadores empiezan a observar un determinado objeto suscribiéndose y cuando este objeto cambia se notifica a sus suscriptores los cambios para que realicen las acciones oportunas. Este patrón es el que utiliza para el enlace de datos definido en MVVM, para que los datos entre el Modelo y la Vista estén sincronizados.

En la Figura 5.12 se puede ver la estructura del patrón Observador [34].

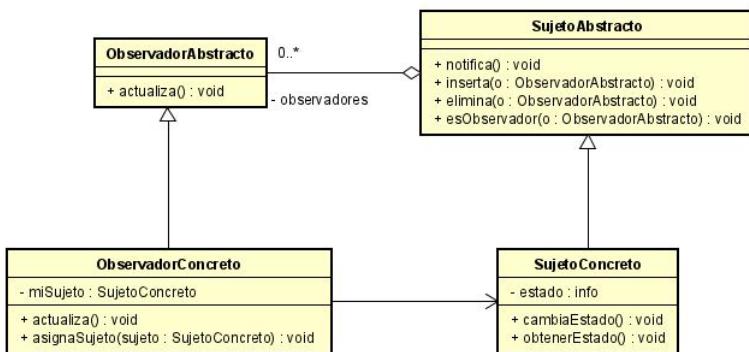


Figura 5.12: Patrón Observador

### 5.3.2. DAO/DTO

El objetivo del patrón DAO (*Data Access Object*) [35] es independizar la forma de acceder a una fuente de datos, de manera que todo el código que se utilice para acceder a la base de datos se agrupe en una clase DAO. Este patrón proporciona la ventaja de poder modificar la API (*Application Programming Interface*) de acceso a los datos, de modificar el repositorio de datos, o de facilidades para implementar controles de acceso a los datos como políticas de seguridad. Este patrón permite separar la lógica de negocio de la lógica de datos.

En este proyecto el patrón DAO se puede ver en el *backend* donde todas las clases DAO se agrupan en el paquete DAO. Cada clase DAO contiene funciones con las diferentes acciones que se pueden realizar sobre una determinada tabla de la base de datos, ya sean de búsqueda, borrado, actualización o creación.

El patrón DTO (*Data Transfer Object*) [36] se utiliza para crear objetos planos que guardan toda la información necesaria que se mostrará en la aplicación con unos atributos que pueden contener información de varias tablas o fuentes. Con el patrón DTO se pueden crear clases especiales para transmitir los datos, almacenando en ellas la información que

### 5.3. PATRONES DE DISEÑO

---

se desee y teniendo más flexibilidad de manera que se puedan crear estructuras de datos independientes del Modelo de Dominio.

El patrón DTO en Angular permite la creación de interfaces con los atributos deseados para agrupar la información de la manera más organizada posible. De esta manera se facilita la representación de las relaciones entre las tablas de la base de datos mediante la creación de clases DTO con atributos para recoger los valores deseados que representen estas relaciones.

En este proyecto el patrón DTO se utiliza en Angular en el *frontend* mediante la creación de interfaces, cada una correspondiendo con una clase del Modelo de Dominio. Cada clase DTO tiene los atributos de la clase del Modelo de Dominio y otros atributos que guarden otra información necesaria de esa clase debido a las relaciones que tiene con otras tablas. En NodeJS en el *backend* también se utiliza este patrón de la misma manera.

En la Figura 5.13 se puede ver la combinación de los patrones DAO y DTO [34].

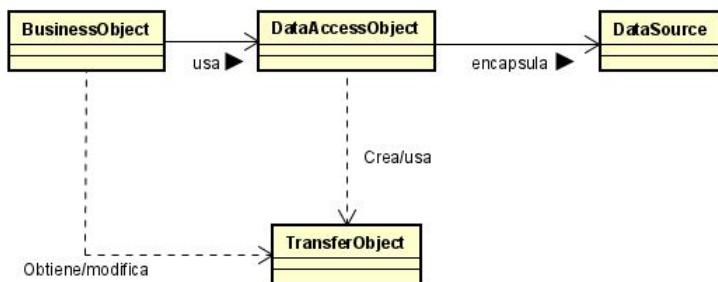


Figura 5.13: Patrón DAO/DTO

#### 5.3.3. *Singleton*

El patrón *Singleton* [37] se encarga de que solo exista una única instancia de una determinada clase, de manera que restringe la libre creación mediante diferentes técnicas. Por otro lado, establece un método para obtener esa instancia. De esta manera, solo existe una única instancia de la clase en toda la aplicación y es accesible de manera global.

Angular proporciona los *Singleton Services* que son servicios que tienen una sola instancia en toda la aplicación, de esta manera se tiene el control sobre cuándo y como se accede a este servicio [38].

Este patrón es utilizado en los servicios de Angular que realizan las peticiones al *backend* de manera que solo existe una instancia de cada servicio.

En la Figura 5.14 se puede ver la estructura del patrón *Singleton* [34].

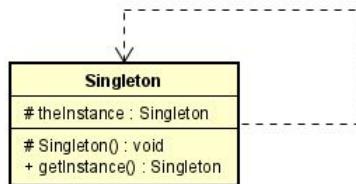


Figura 5.14: Patrón Singleton

## 5.4. Arquitectura Física del Sistema

El Diseño de Despliegue [39] se realiza para definir los dispositivos del sistema, sus enlaces de comunicación entre ellos y cómo se distribuyen los archivos del sistema entre ellos. Muestra la arquitectura en ejecución del sistema incluyendo su *hardware*, *software* y *middleware*. Para realizar el Diseño de Despliegue se definirá un Diagrama de Despliegue que contendrá los dispositivos dentro del sistema junto a los elementos *software* dentro de ellos y las relaciones entre estos dispositivos.

En la Figura 5.15 se muestra el Diagrama de Despliegue de la aplicación Web.

## 5.5. Diseño de la Base de Datos

El Diseño de la Base de Datos [40] es el proceso para definir la estructura adecuada de la base de datos que almacenará toda la información de la aplicación. Para ello se deben definir las diferentes tablas que se almacenarán en la base de datos, sus atributos junto a su tipo y las relaciones entre las diferentes tablas mediante las claves primarias y foráneas. Para el Diseño de la Base de Datos se va a utilizar un Diagrama de la Base de Datos que contendrá las tablas de la base de datos con sus atributos y su tipo y las relaciones entre ellas. Para diseñar la base de datos se ha utilizado como diagrama conceptual el Diagrama de Clases del Modelo de Dominio que se encuentra en la sección 4.2.1 a partir del cuál se han sacado las tablas de la base de datos.

En la Figura 5.16 se muestra el Diseño de la Base de Datos mediante su Diseño Lógico, para ello se utiliza una representación basada en UML con estereotipos. Los estereotipos «PK» y «FK» representan las claves primarias y foráneas. El estereotipo «table» sirve para representar una tabla en el esquema relacional. El estereotipo «entity» está en las tablas que representan un elemento del Dominio.

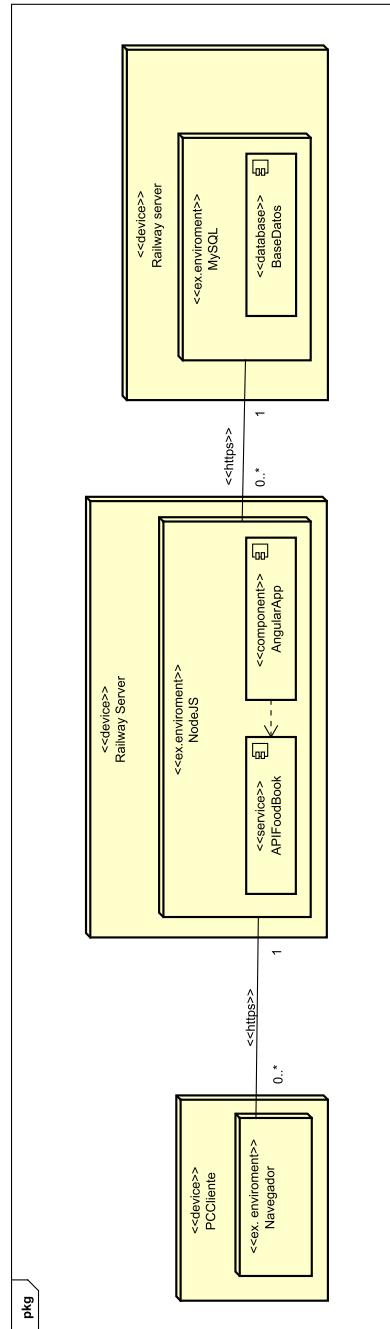


Figura 5.15: Diagrama de Despliegue

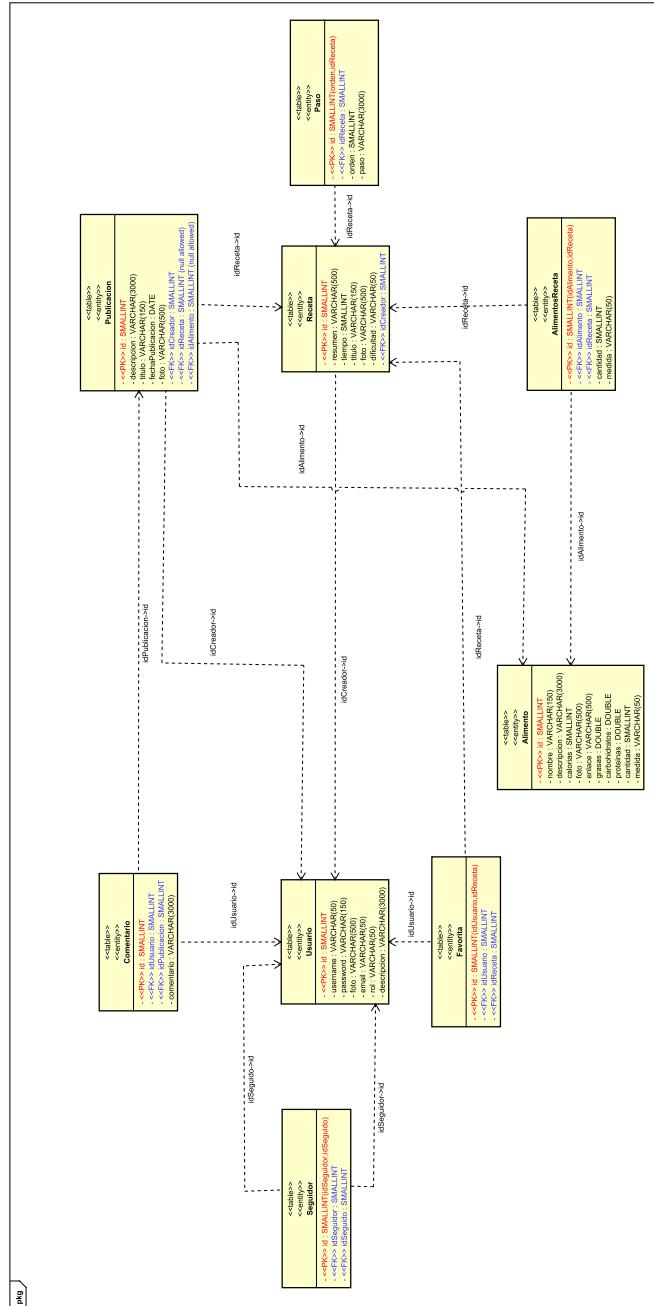


Figura 5.16: Diseño Lógico de la Base de Datos

## 5.6. Diseño de la Interfaz Gráfica

El *frontend* [41] de un sitio Web son todas las tecnologías de diseño y desarrollo Web que se visualizan en un navegador, es decir, la parte visible para los usuarios y tienen como fin interactuar con el Usuario que utilice el sitio Web. Determinará en gran medida el número de clientes de la aplicación.

La interfaz gráfica es fundamental para que los usuarios puedan navegar por la página Web, para ello el *frontend* se conecta con el *backend* para acceder a todos los contenidos que estén almacenados en la base de datos. Si la interfaz de una aplicación Web no está bien construida, el cliente podría abandonar el sitio Web porque no entiende cómo funciona.

Por lo tanto, los objetivos que debe cumplir el *frontend* son implementar los elementos visuales del sitio Web de manera que se facilite la interacción con el Usuario, mandar los datos recogidos en la interfaz al *backend* de manera adecuada para que pueda almacenarlos y saber interpretar los datos que envíe el *backend* para mostrárselos de manera adecuada al Usuario.

Otro aspecto importante, es cómo funciona la interfaz en los dispositivos móviles, ya que el uso de *smartphones* y *tablets* está completamente implementado en la sociedad actual. Por lo tanto para aumentar el número de usuarios es fundamental que la aplicación Web funcione desde cualquier dispositivo sea cual sea su tamaño de pantalla [42].

La experiencia de usuario [43] hace referencia a como se siente el Usuario al interactuar con el sistema. Aspectos sobre esto son la facilidad de uso, la utilidad, el valor percibido y la eficiencia a la hora de realizar una tarea. Una buena experiencia de usuario es fundamental para atraer usuarios. La experiencia del usuario ha tomado relevancia debido a las siguientes razones:

- **La complejidad de las páginas Web ha aumentado.** Actualmente las aplicaciones Web tienen muchas funcionalidades y contenidos.
- **Dispositivos diferentes.** Los usuarios acceden a las aplicaciones Web desde multitud de dispositivos diferentes.
- **Cada vez se valora más la accesibilidad.** Es importante que la aplicación Web se adapte a todo tipo de usuarios, tanto como los que tienen problemas visuales como los que tienen una conexión lenta o un dispositivo antiguo.
- **Exigencias de los usuarios.** La aplicación Web tiene que crear un producto único y de calidad, que haga que la experiencia de usuario sea agradable.

Los bocetos aquí creados sirven como ideas previas al desarrollo final de cada vista de la aplicación, por lo tanto la interfaz final desarrollada puede variar ligeramente respecto a estos bocetos. Por otro lado, estos bocetos son diseños simplificados de las interfaces que compondrán la aplicación Web, que principalmente se utilizarán como referencia para ver la estructura y organización de los elementos que compondrán cada pantalla de la aplicación.

Por ello, los detalles más específicos como los colores no se han incluido en muchos de estos bocetos.

Se ha creado un componente de Angular por cada interfaz que se pueda acceder en la aplicación Web. También se han definido componentes para representar a las entidades del Modelo de Dominio que se muestran en la aplicación, de manera que cada instancia de una entidad se muestra mediante una carta donde se muestra la información relevante.

En las Figuras 5.17 a 5.22 se muestran los bocetos de las cartas que se utilizarán para resumir los datos principales de cada elemento que se mostrará en la aplicación.

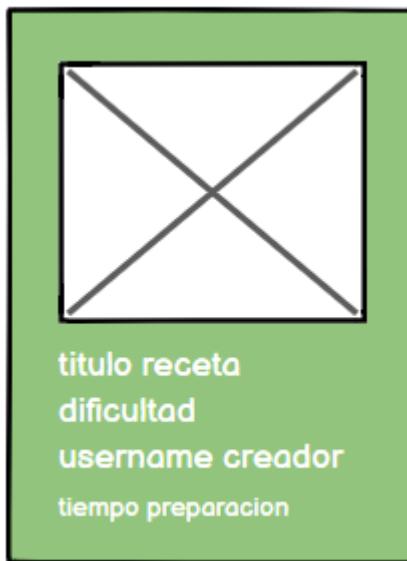


Figura 5.17: Boceto de la carta de una receta



Figura 5.18: Boceto de la carta de un Usuario



Figura 5.19: Boceto de la carta de una publicación



Figura 5.20: Boceto de la carta de un alimento

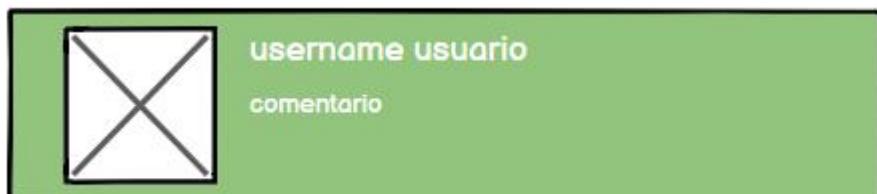


Figura 5.21: Boceto de la carta de un comentario

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---



Figura 5.22: Boceto de la carta de un alimento incluido en una receta

En las Figuras 5.23 a 5.43 se muestran los bocetos de las interfaces que permitirán la navegación por la aplicación y que utilizarán esas cartas.

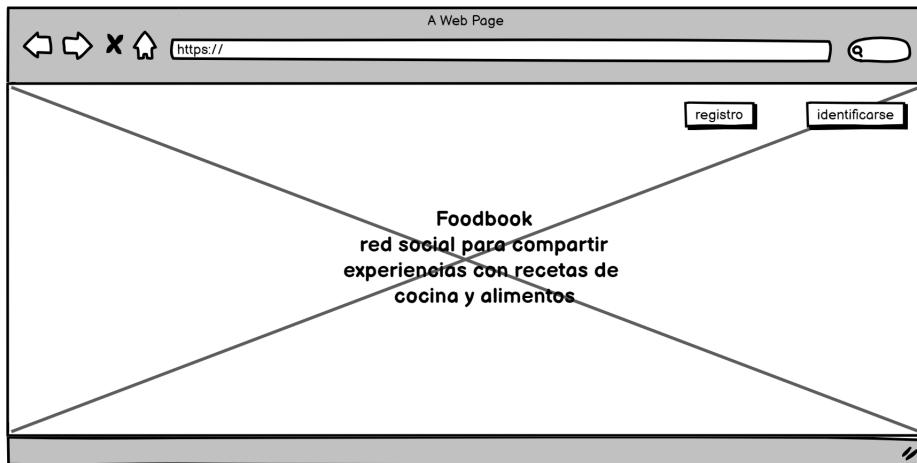


Figura 5.23: Parte superior del boceto de la interfaz de descripción de la aplicación.

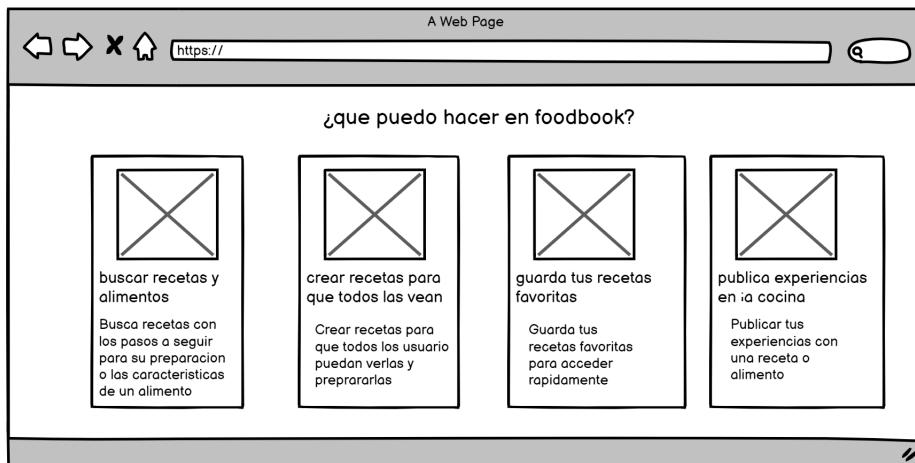


Figura 5.24: Parte inferior del boceto de la interfaz de descripción de la aplicación.

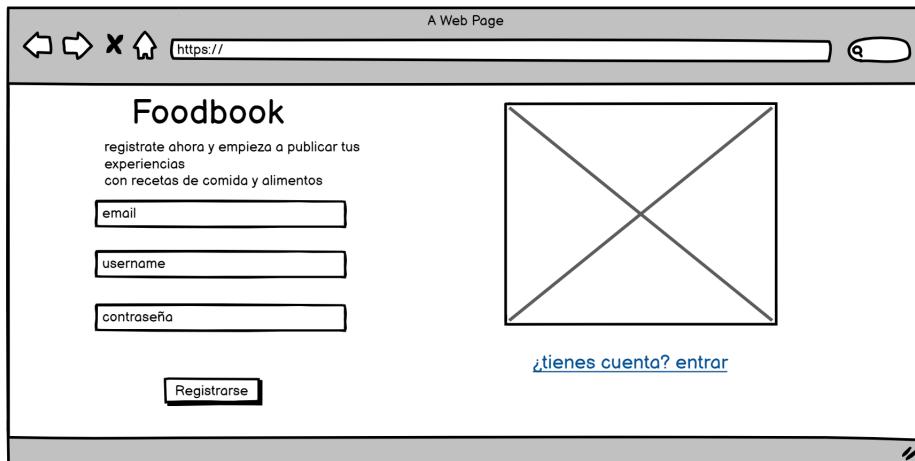


Figura 5.25: Boceto de la interfaz de registro de un Usuario.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---

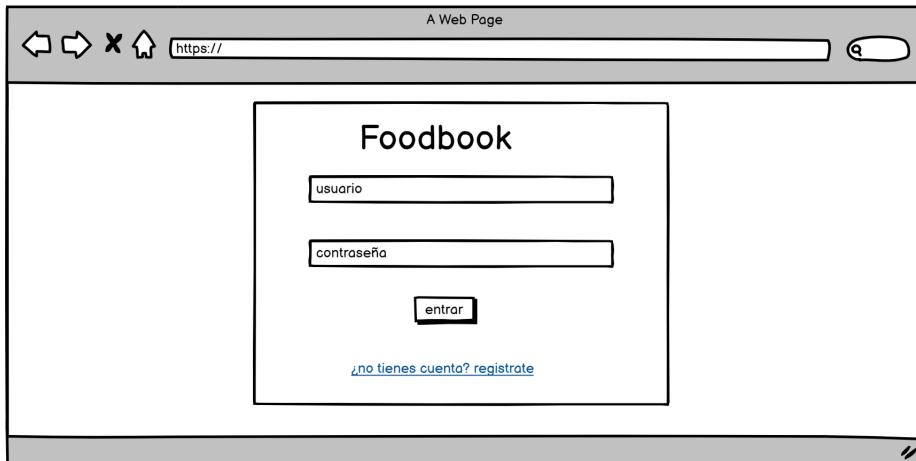


Figura 5.26: Boceto de la interfaz de identificación de un Usuario.

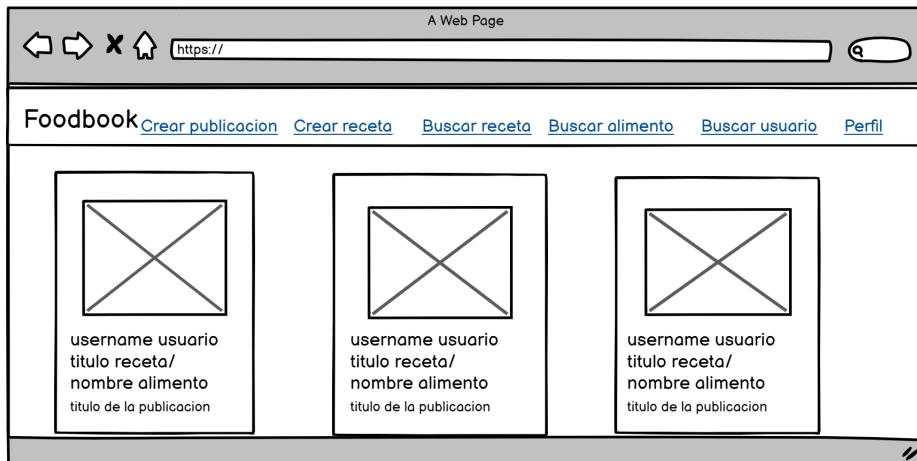


Figura 5.27: Boceto de la interfaz de muro de publicaciones.

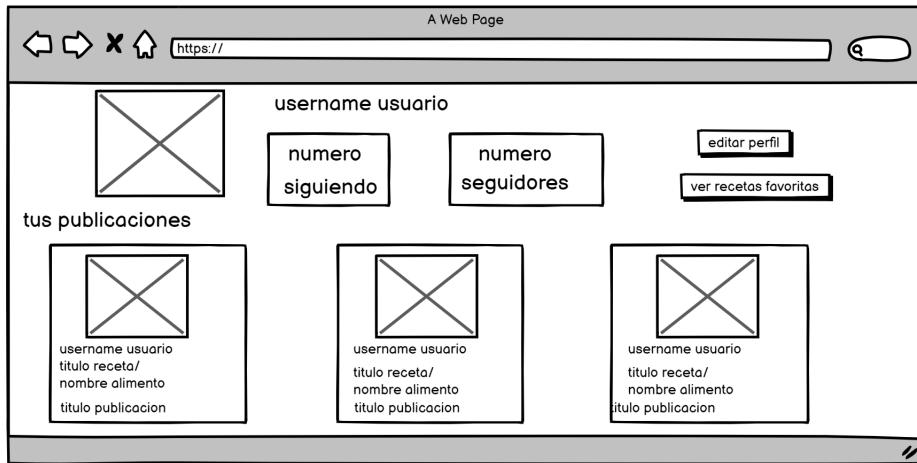


Figura 5.28: Boceto de la interfaz de perfil del Usuario.

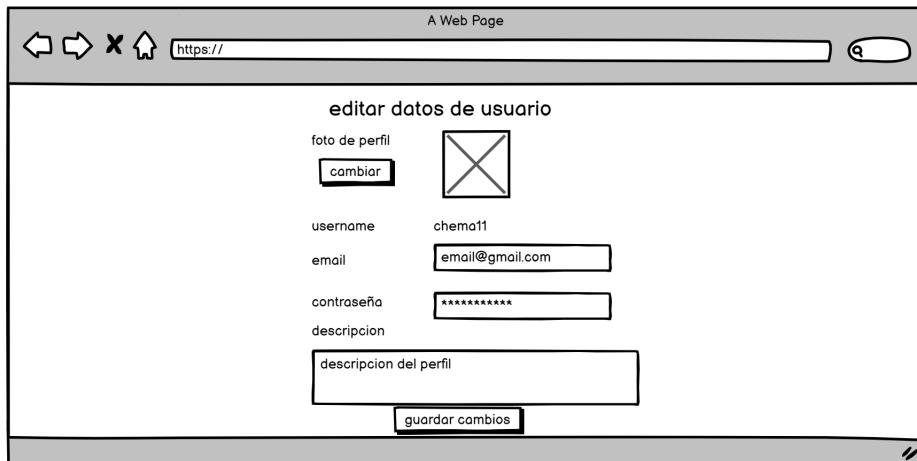


Figura 5.29: Boceto de la interfaz de editar perfil del Usuario.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---

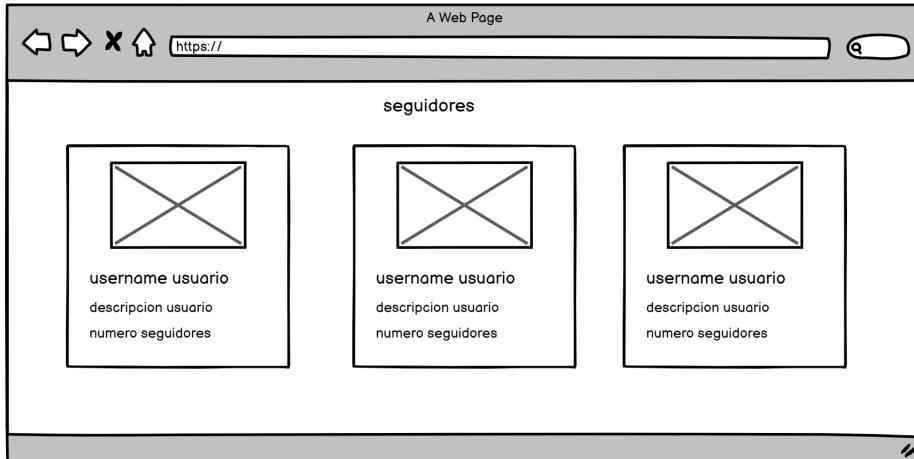


Figura 5.30: Boceto de la interfaz de seguidores del Usuario.

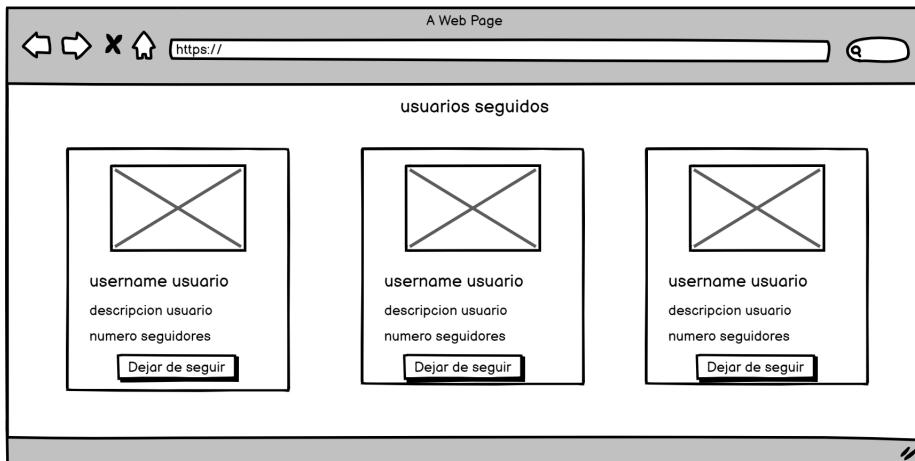


Figura 5.31: Boceto de la interfaz de seguidos del Usuario.

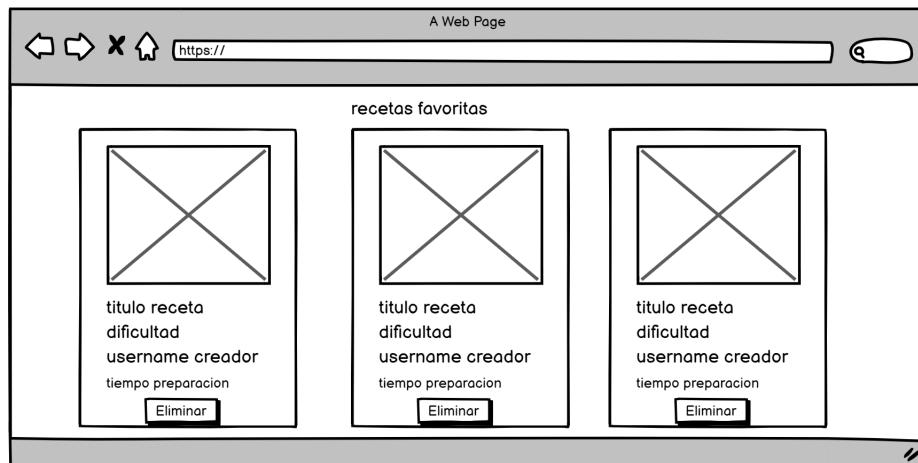


Figura 5.32: Boceto de la interfaz de recetas favoritas.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---

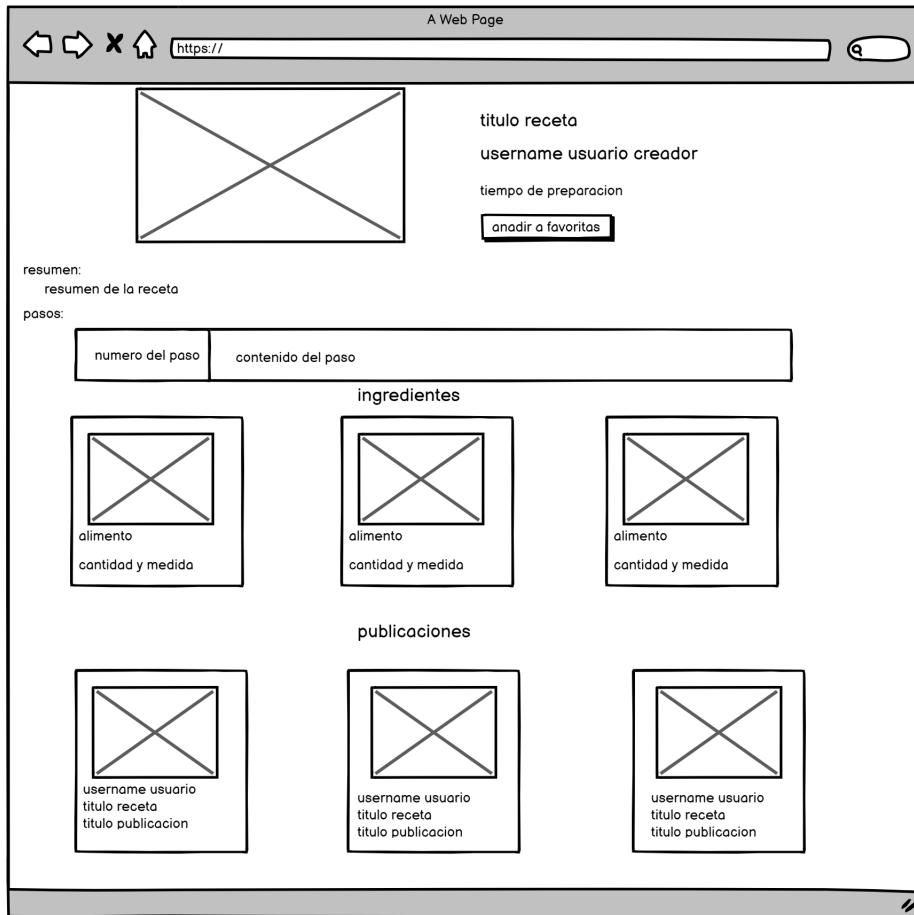


Figura 5.33: Boceto de la interfaz de detalles de la receta.

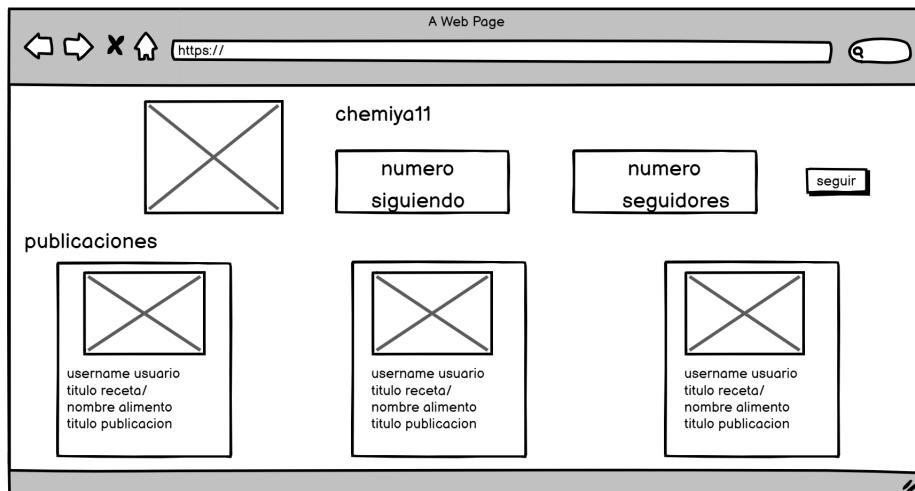


Figura 5.34: Boceto de la interfaz de detalles del Usuario.

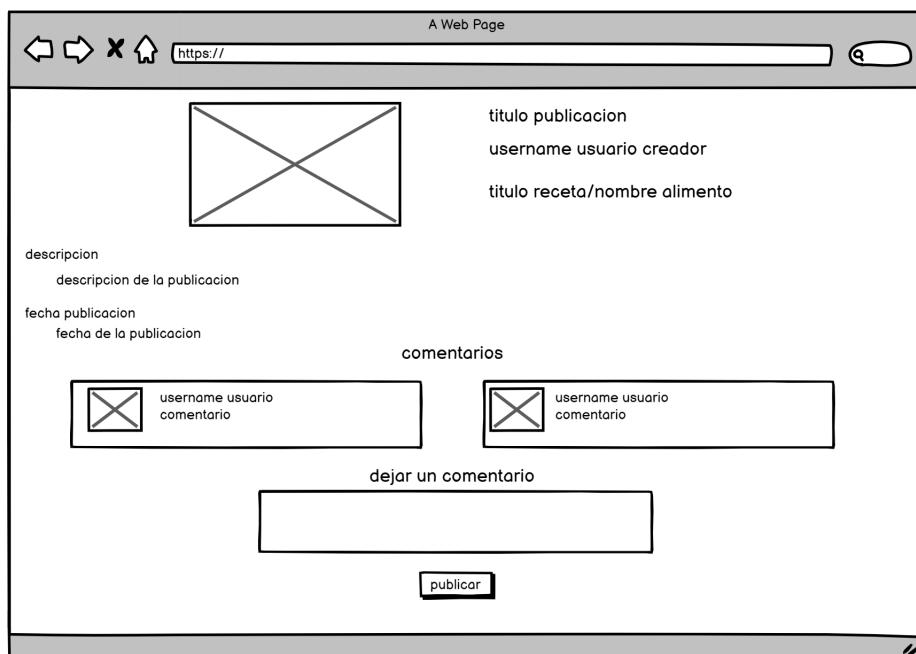


Figura 5.35: Boceto de la interfaz de detalles de la publicación.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---

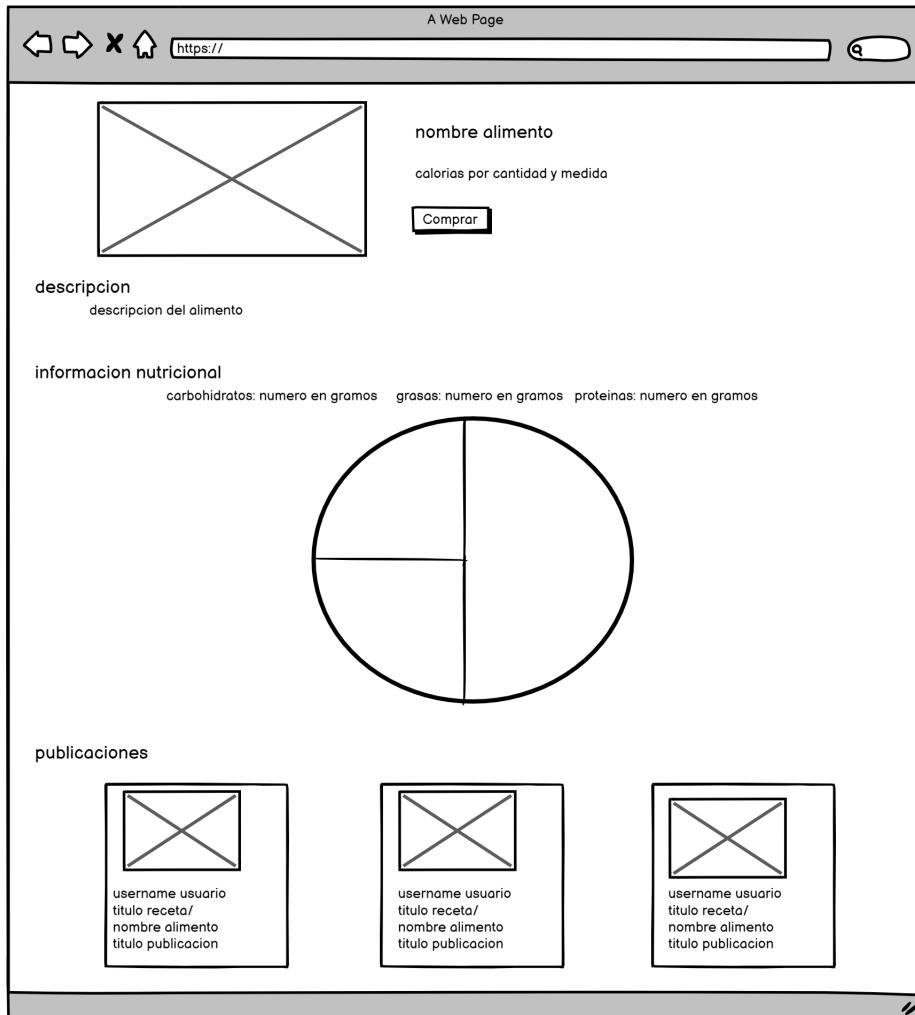


Figura 5.36: Boceto de la interfaz de detalles del alimento.

## CAPÍTULO 5. DISEÑO

A Web Page  
https://

crear receta

titulo

resumen

tiempo en minutos

dificultad

foto

pasos

numero de orden  contenido del paso

ingredientes

nombre alimento  
cantidad en medida  
numero

nombre alimento  
cantidad en medida  
numero

nombre alimento  
cantidad en medida  
numero

buscador

nombre del alimento  buscar

nombre alimento  
numero calorías

nombre alimento  
numero calorías

nombre alimento  
numero calorías

guardar receta

Figura 5.37: Boceto de la interfaz de crear receta.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

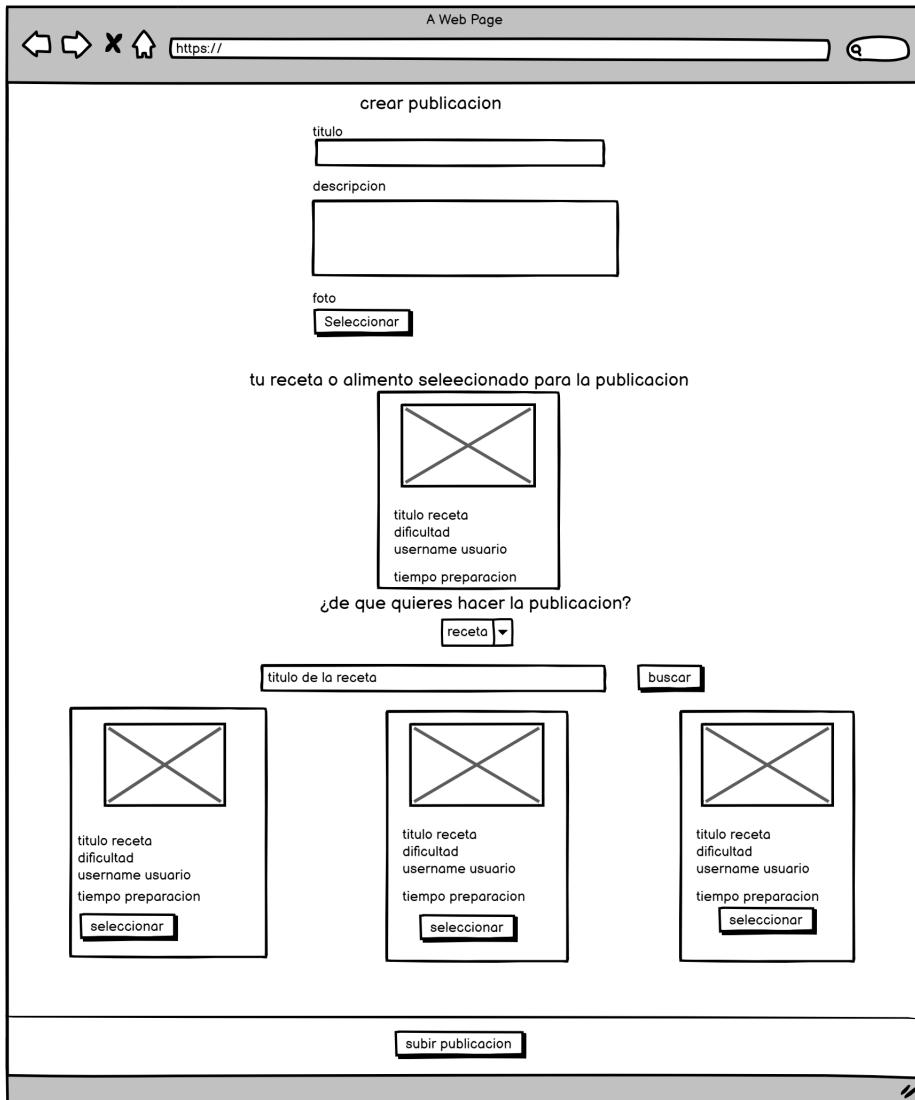
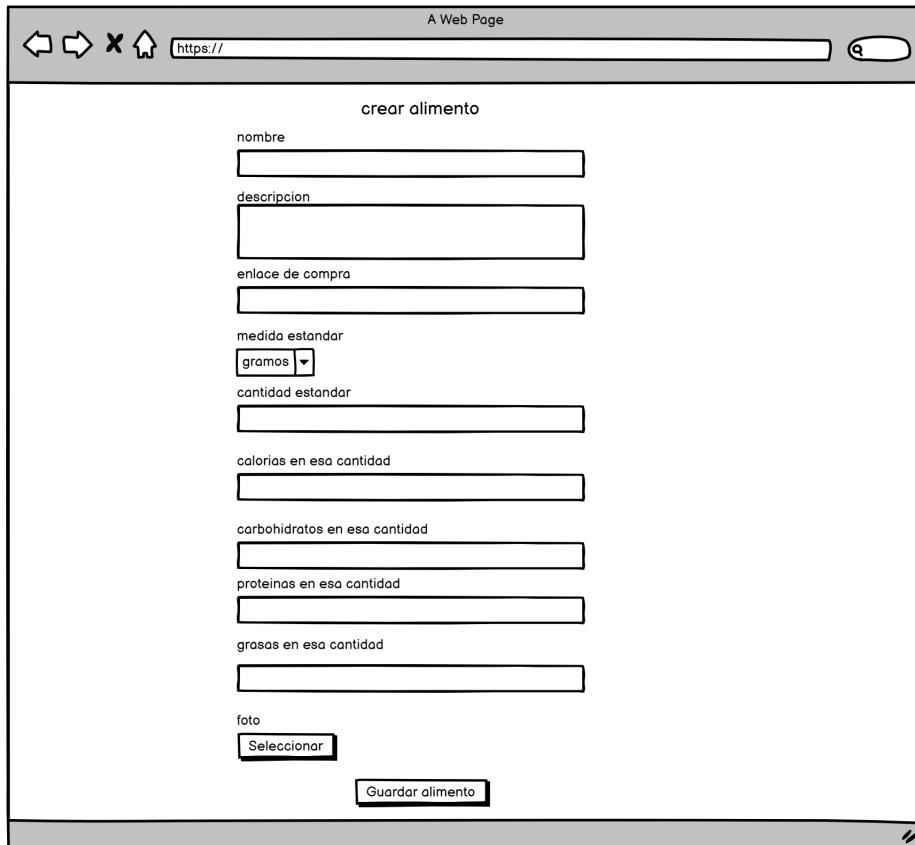


Figura 5.38: Boceto de la interfaz de crear publicación.

## CAPÍTULO 5. DISEÑO

A Web Page



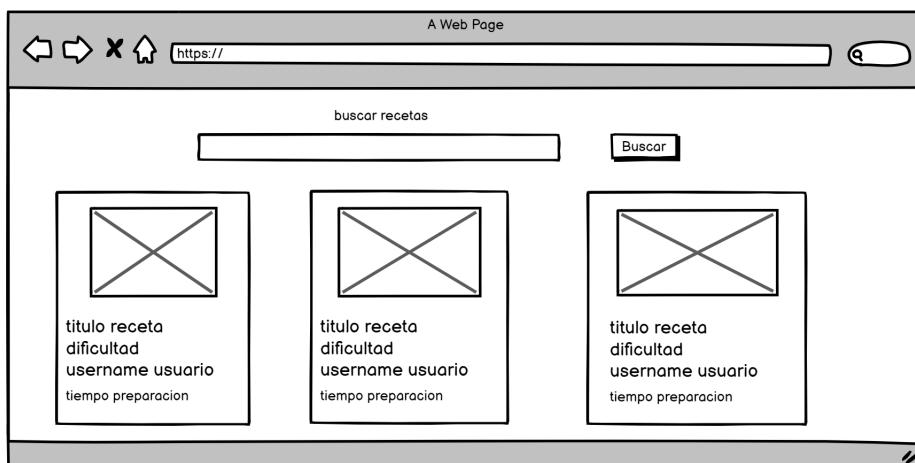
Este boceto muestra una interfaz web para crear un nuevo alimento. La barra superior incluye iconos para retroceder, avanzar, cerrar y volver, así como el URL "https://". El formulario principal se titula "crear alimento" y contiene los siguientes campos:

- campo de texto para "nombre"
- campo de texto para "descripcion"
- campo de texto para "enlace de compra"
- campo desplegable para "medida estandar" con la opción "gramos"
- campo de texto para "cantidad estandar"
- campo de texto para "calorías en esa cantidad"
- campo de texto para "carbohidratos en esa cantidad"
- campo de texto para "proteínas en esa cantidad"
- campo de texto para "grasas en esa cantidad"
- campo para "foto" con un botón "Seleccionar"

En la parte inferior del formulario hay un botón "Guardar alimento".

Figura 5.39: Boceto de la interfaz de crear alimento.

A Web Page



Este boceto muestra una interfaz web para buscar recetas. La barra superior incluye iconos para retroceder, avanzar, cerrar y volver, así como el URL "https://". El formulario principal se titula "buscar recetas" y contiene los siguientes elementos:

- campo de texto para "buscar recetas" con un botón "Buscar" adjunto
- tres resultados de búsqueda representados por cuadros que contienen un icono de receta (una caja con una X) y la siguiente información:
  - "titulo receta"
  - "dificultad"
  - "username usuario"
  - "tiempo preparacion"

Figura 5.40: Boceto de la interfaz de buscador de recetas.

## 5.6. DISEÑO DE LA INTERFAZ GRÁFICA

---

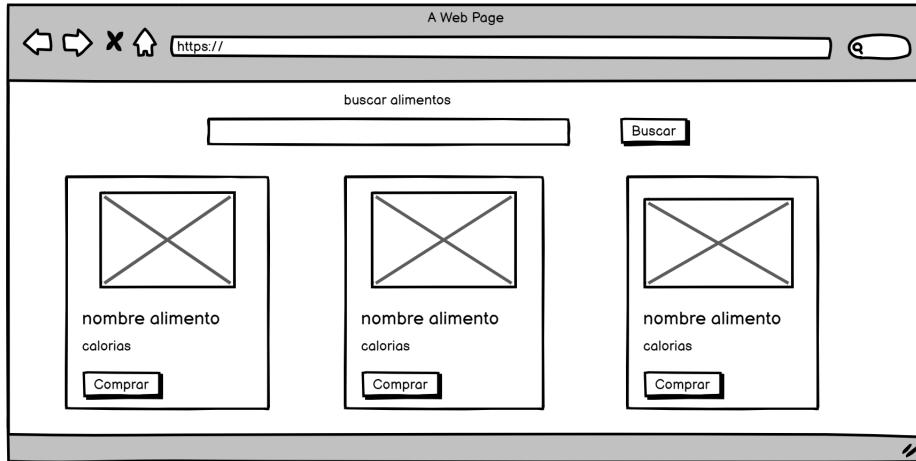


Figura 5.41: Boceto de la interfaz de buscador de alimentos.

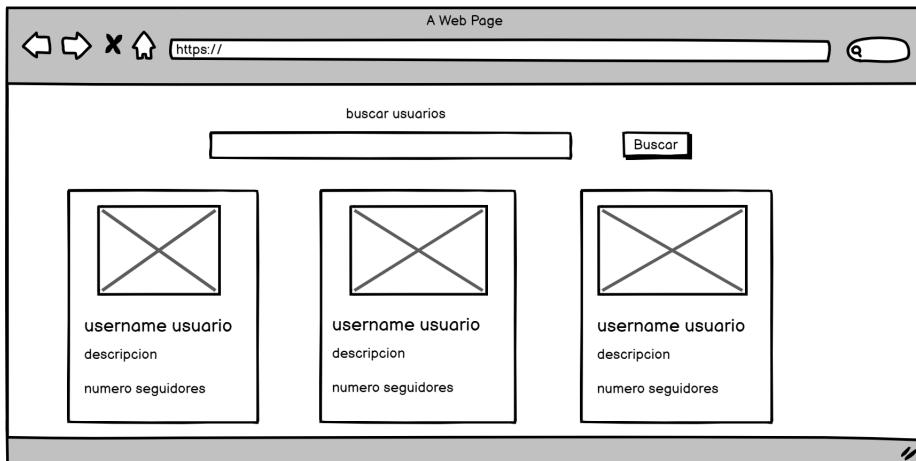


Figura 5.42: Boceto de la interfaz de buscador de usuarios.

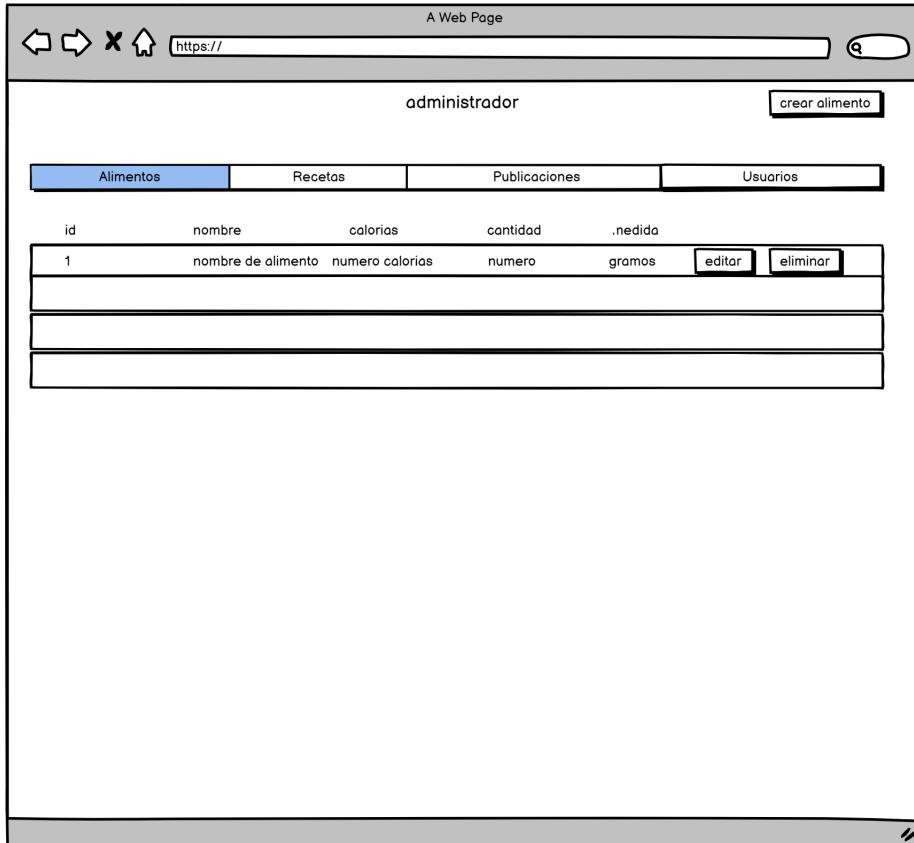


Figura 5.43: Boceto de la interfaz de vista de Administrador.

La selección de los colores para una aplicación Web es importante ya que determinará en gran medida lo que vea el Usuario. Para todo sitio Web cuantos menos colores contenga, será más fácil para el Usuario analizar el sitio Web y de esta forma sea más usable y accesible. Por lo tanto, una buena práctica para el diseño de sitios Web es utilizar tres colores, siendo uno para el fondo, otro para los textos y un color principal que represente la marca personal [44]. Para este proyecto se han escogido los siguientes colores:

- Color del fondo: blanco: #FFFFFF
- Color del texto: negro: #000000
- Color principal: verde. Se ha escogido el verde como el color principal para la aplicación Web ya que está estrechamente relacionado con la alimentación, en especial con la alimentación saludable. Por lo tanto, se ha escogido el verde claro #00B347 para las interfaces de la aplicación con la variación del verde oscuro #00722E para los efectos que haya que realizar cuando el Usuario interactúe con la aplicación Web.



# Capítulo 6

# Tecnologías utilizadas

## 6.1. Introducción

En este capítulo se explican las tecnologías utilizadas comentando brevemente sus principales ventajas y su funcionamiento. También se explica el motivo por el qué se han elegido esas tecnologías.

## 6.2. Base de datos

### 6.2.1. Elección del tipo de base de datos

Dentro de los diferentes tipos de bases de datos, existen dos principales grupos, las bases de datos relacionales y las no relacionales. La elección de uno de estos tipos para cualquier tipo de proyecto determina en gran medida el éxito del mismo. El contenido deberá estar estructurado y organizado para que se pueda acceder a él eficientemente. Una mala elección de la base de datos puede desembocar en una larga lista de problemas durante el desarrollo de la aplicación, lo que provocaría consecuencias fatales para el proyecto, por lo que es una decisión crítica [45].

Las bases de datos relacionales [46] consisten en una colección ordenada de registros que se organizan en diferentes tablas que se relacionan entre sí, de esta manera se pueden acceder a los contenidos de diferentes tablas y realizar cambios en los datos almacenados sin tener que reorganizar las tablas. Se utiliza SQL (*Structured Query Language*) para acceder a los datos y alterarlos. Estas bases de datos utilizan identificadores únicos para cada tabla, de esta manera se pueden establecer relaciones con otras tablas. Cada fila en una tabla es un registro que se identifica por este identificador único. Los principales sistemas gestores de bases de datos relacionales son MySQL, MariaDB o PostgreSQL entre otros.

## 6.2. BASE DE DATOS

---

Las bases de datos no relacionales [47] fueron diseñadas para modelos de datos específicos, y que al contrario que las bases de datos relacionales, no necesitan relacionarse con otros modelos. Este tipo de base de datos está cobrando actualmente mucha importancia debido a su acceso a los datos de manera muy sencilla. También utilizan un identificador único para cada registro de cada tabla, pero este identificador no se utiliza para relacionarlo con otro registro de otra tabla, como sucede con las bases de datos relacionales. El formato más habitual para este tipo de base de datos es el documento. Los sistemas gestores de bases de datos principales para las bases de datos no relacionales son MongoDB o Cassandra entre otros.

En este proyecto, donde hay diferentes tipos de datos que cada uno define una tabla, cada una con varias relaciones con las demás tablas, una base de datos relacional se ajustaría mejor. Esta es la principal razón por la que se ha escogido una base de datos relacional, ya que este proyecto contiene diferentes tipos de datos, con siempre los mismos atributos y cada tipo de dato que representa una tabla, se relaciona con varias tablas más.

Las principales ventajas de un base de datos relacional son:

- La sencillez del modelo relacional, lo que permite almacenar grandes cantidades de datos relacionados entre sí.
- Garantiza la uniformidad de los datos.
- No existe la duplicidad de registros.
- Evita conflictos cuando varios usuarios quieren acceder a los mismos datos en el mismo momento.
- Buen rendimiento debido a la gran cantidad de herramientas que existen para su uso.

Las bases de datos relacionales cumplen con los principios *ACID (Atomicity, Consistency, Isolation and Durability)* [48]. Estas son unas propiedades para garantizar la seguridad en las transacciones, es decir, cada operación que se realiza sobre la base de datos. La descripción de estos principios es:

- **Atomicidad:** esta propiedad define que para que una transacción se dé completada, se tienen que haber realizado todas las partes que la componen o ninguna de ellas. De esta forma, si se completan todas las partes de la transacción, se habrá modificado los datos de la manera deseada. Pero si una parte falla, tienen que fallar el resto de las partes.
- **Consistencia:** se refiere a que el sistema debe tener la capacidad de iniciar solo las operaciones que puede concluir, lo que significa que solo pueda ejecutar las partes de una transacción que cumplan las reglas de integridad definidas. Una transacción lleva al sistema de una condición válida a otra que también lo sea.
- **Aislamiento:** define el momento en el que los cambios realizados por una transacción se harán visibles al resto de transacciones concurrentes. Si se realiza una operación, no se debe afectar a otras, debido a que cada una debe ser ejecutada en un aislamiento total. El estado intermedio de cualquier transacción no debe ser visible a las demás.

- **Durabilidad:** una vez que se realiza una operación, tiene que persistir y no puede ser deshecha incluso si el sistema falla o se presenta un error. Los datos y cambios en una transacción que ha finalizado deben ser persistentes.

### 6.2.2. MySQL

MySQL [49] es el sistema gestor de base de datos relacional que más extendido está en la actualidad, principalmente por estar basado en código abierto. Fue desarrollado originalmente por MySQL AB. Como está basado en código abierto, es accesible por todos los desarrolladores, lo que hace que esté ampliamente extendido con una gran comunidad que ofrece soporte a otros usuarios. Sus principales ventajas son:

- **Arquitectura cliente y servidor:** cada cliente hace consultas para obtener los datos o realizar modificaciones y el servidor responde realizando estas peticiones.
- **Compatibilidad con SQL:** SQL está generalizado en la industria, y MySQL al ser un estándar tiene plena compatibilidad, lo que facilita las migraciones de base de datos.
- **Vistas:** se pueden configurar vistas personalizadas.
- **Procedimientos almacenados:** no procesa las tablas directamente, sino utilizando los procedimientos almacenados hace que la eficacia sea mayor.
- **Desencadenantes:** permite automatizar tareas en la base de datos, de forma que cuando se produce un evento, otro evento puede realizar otra funcionalidad.
- **Transacciones:** cada operación en la base de datos debe cumplir los principios ACID antes comentados.

Se ha utilizado MySQL Workbench como herramienta visual para el diseño, administración y gestión de la base de datos MySQL. Se ha utilizado la versión MySQL Workbench 8.0.32.

## 6.3. Frontend

### 6.3.1. Angular

Angular [50] [51] es un *framework* de código abierto que es utilizado para crear aplicaciones Web de una sola página llamadas *Single-Page Application (SPA)* [52]. Una *Single-Page Application* es una aplicación Web que ejecuta todo su contenido en una sola página, y que funciona mediante la carga de todo el contenido HTML, CSS, Javascript por completo al momento de abrir la Web. Si se pasa a otra sección solo se necesita cargar el nuevo contenido, pero no se tiene que cargar toda la página por completo. Esto provoca mejoras en

### 6.3. FRONTEND

---

los tiempos de respuesta y en la agilidad de la navegación que hace que la experiencia del Usuario sea mejor.

Angular fue creado por Google y su primera versión fue en 2012. Está basado en componentes para que las aplicaciones Web creadas sean escalables, tiene una colección de bibliotecas bien integradas que cubren muchas características, y un gran conjunto de herramientas.

Está construido sobre Typescript que es un lenguaje programación construido a un nivel superior de Javascript, por lo que lo dota de características adicionales. Todo el código que esté escrito en Javascript sirve para Typescript [53].

Como las aplicaciones Web en angular están basadas en componentes, cada componente contiene un archivo HTML, que será lo que vea el usuario, un archivo CSS para dar estilo a ese componente y un archivo Typescript con la lógica del componente.

Además de los componentes, los elementos principales en la arquitectura de Angular son los módulos, servicios y las directivas.

Las principales ventajas de Angular son:

- **Enlace bidireccional de datos:** la arquitectura de Angular enlaza Typescript y HTML, por lo tanto, el código de ambos está sincronizado.
- **Directivas:** se amplía la funcionalidad de los archivos HTML con el uso de directivas entre las que destacan ngModel, ngIf y ngFor.
- **Estructura de código:** contiene plantillas, lo que permite producir aplicaciones con código limpio.
- **Pruebas:** Angular permite pruebas unitarias y de integración.
- **Compatibilidad móvil y escritorio:** puede ejecutarse en la mayoría de los navegadores Web, y en tanto equipos de escritorio como dispositivos móviles.
- **Servicios:** permiten compartir información entre componentes y obtener datos del *backend* para mostrarlos en las vistas.
- **Rutas:** permite gestionar las rutas de la aplicación en el *frontend*.
- **Inyección de dependencias:** permite a los componentes consumir servicios mediante la inyección del servicio en el componente.

Debido a que se tiene experiencia previa con Angular por la realización de proyectos previos, y que los componentes y servicios de Angular se ajustaban correctamente a la estructura del proyecto, se eligió Angular como el *framework* para el *frontend*.

Se utiliza la versión 15.1.4 de Angular para el desarrollo del proyecto.

### 6.3.2. HTML5

*HyperText Markup Language 5* [54] es el componente más básico de la Web y se utiliza para definir el significado y la estructura del contenido que se presenta en la Web. El hipertexto se refiere a los enlaces que se utilizan para conectar páginas Web entre sí y son un aspecto fundamental en la Web. HTML utiliza marcas para etiquetar todo el contenido que se muestra en el navegador Web.

HTML es un estándar a cargo de W3C (*World Wide Web Consortium*) que es la principal organización dedicada a la estandarización de las tecnologías ligadas a la Web.

Se utiliza para este proyecto HTML en su versión 5.

### 6.3.3. CSS

*Cascading Style Sheets* [55] es un lenguaje de estilos que se utiliza para definir la presentación de los documentos HTML, lo que describe como se debe renderizar un elemento en la pantalla del Usuario al visitar una página Web. Tiene una especificación estandarizada por parte de W3C. Un gran avance en los últimos años para este lenguaje ha sido que el alcance de las especificaciones se ha incrementado enormemente, lo que hizo este lenguaje más efectivo para desarrollar.

El significado de *Cascading Style Sheets* hace referencia a que los estilos que se aplican a un elemento HTML son también propagados a los elementos que contiene en cascada. Lo que se crea con este lenguaje son estilos que se aplican a los elementos de la Web y los estilos declarados están en ficheros aparte de los archivos HTML.

Para este proyecto se utiliza CSS en su versión 3.

### 6.3.4. Bootstrap

Bootstrap [56] es un *framework* CSS desarrollado por Twitter en 2010 cuyo objetivo era estandarizar las herramientas de la compañía. Combina CSS y Javascript para dar estilos a los elementos que componen una página HTML, de manera que está compuesto por unos archivos CSS y Javascript que asignan las características deseadas a cada elemento. Proporciona una serie de componentes que facilitan el desarrollo con él. Estos componentes se almacenan en una biblioteca que además ayudan a una mejor interacción con el Usuario y, por lo tanto, que la comunicación con él sea mejor. El objetivo que persigue este *framework* es la construcción de sitios Web *responsive* de manera que se puedan ver en dispositivos móviles, además de en equipos de escritorio y otros dispositivos con diferentes tamaños de pantalla.

Se ha escogido Bootstrap para el desarrollo de este proyecto para dar estilos a las vistas de la aplicación de manera que se puedan adaptar a cualquier tamaño de pantalla. Se ha elegido esta tecnología debido a la experiencia acumulada con el uso de este *framework*.

Se ha utilizado para este proyecto la versión 5.0 de Bootstrap.

### 6.3.5. ChartJS

ChartJS [57] es una librería de Javascript de código abierto que permite crear gráficos en páginas Web. No requiere de dependencias externas y tiene la gran ventaja de que se puede integrar con cualquier *framework*. Tiene una amplia documentación y comunidad y al estar basada en Javascript no se requieren más conocimientos específicos para su uso, lo que permite realizar multitud de gráficos sin la necesidad de tener que realizar un gran esfuerzo de aprendizaje.

ChartJS se utiliza en el proyecto para la realización de un gráfico circular con la información nutricional de los alimentos y se ha escogido debido a su fácil implementación en la aplicación Web. Además al basarse en Javascript no requería aprendizaje, ya que se está familiarizado con este lenguaje de programación.

Se ha utilizado la versión 4.2.1 de ChartJS en este proyecto

### 6.3.6. Angular Material

Angular Material [58] es una librería utilizada para los proyectos desarrollados en Angular que acelera el desarrollo de las interfaces y hace que estas sean más elegantes. Proporciona componentes ya definidos que son reutilizables y que facilitan la interacción del Usuario con la página Web. Sus componentes están basados en las especificaciones de *Material Design* que es un lenguaje de diseño creado por Google para facilitar el diseño de sitios Web.

Se escogió Angular Material para realizar ciertos detalles de la aplicación Web debido a las facilidades que ofrece, de manera que Angular Material proporciona componentes ya definidos que requerirían mucho trabajo su creación desde cero simplemente con CSS. Por lo tanto, la utilización de Angular Material ahorra esfuerzo y proporciona una interfaz mucho más elegante.

Se ha utilizado la versión 15.2.0 de Angular Material en el proyecto.

## 6.4. Backend

### 6.4.1. NodeJS

NodeJS [59] es un entorno de ejecución de Javascript, por lo que incluye todo lo necesario para ejecutar cualquier programa que esté escrito en Javascript. Fue creado por los desarrolladores de Javascript, con el objetivo de transformar Javascript que solo se podía ejecutar en navegador a algo que se pudiese ejecutar en el PC. Se ejecuta en el motor de tiempo de ejecución Javascript V8, que es el motor de Javascript que alimenta Google Chrome. La función del motor es transformar el código en Javascript en código máquina rápidamente. NodeJS usa un modelo de entrada y salida sin bloqueo que es controlado por eventos, de esta

forma se mantiene liviano y eficiente. Su objetivo no es realizar operaciones intensivas con el procesador, sino para la creación de aplicaciones de red rápidas, ya que es capaz de manejar muchas conexiones simultáneas lo que hace que tenga gran escalabilidad.

Para la parte de *backend*, las dos opciones principales eran NodeJS y Spring Boot. Con Spring Boot sí que se tenía experiencia previa, pero NodeJS es actualmente la tecnología más demandada para la parte de *backend* de las aplicaciones, y aunque no se tuviese tanta experiencia como con Spring boot, al estar relacionados Javascript con NodeJS, y teniendo conocimientos sobre Javascript, la curva de aprendizaje de NodeJS no iba a ser muy elevada.

Se utiliza la versión 18.14.0 de NodeJS para el desarrollo del proyecto.

#### 6.4.2. ExpressJS

ExpressJS [60] es un *framework* de NodeJS que proporciona herramientas para desarrollar aplicaciones de *backend* que sean escalables. Su principal cualidad es que ofrece un sistema de enrutamiento y permite ampliar su funcionalidad con componentes dependiendo el uso que se le vaya a dar a una aplicación. Proporciona un conjunto de herramientas para aplicaciones Web, peticiones y respuestas HTTP, enrutamiento y *middleware* y también incorpora opciones para gestionar sesiones y *cookies*.

Se ha utilizado ExpressJS en el proyecto en la parte de *backend* para crear una API REST. De esta manera, la parte de *frontend* envía peticiones a esta API REST con las operaciones que se quieran realizar.

Se ha utilizado ExpressJS en su versión 4.18.2 para este proyecto.

### 6.5. Desarrollo del proyecto

#### 6.5.1. Git

Git [61] es el sistema de control de versiones más utilizado actualmente. Es de código abierto y con un mantenimiento activo. Fue creado por Linus Torvalds en 2005. Los desarrolladores globalmente están muy familiarizados con la ventaja de que funciona con una amplia variedad de sistemas operativos y entornos de desarrollo.

Un sistema de control de versiones [62] es una herramienta *software* que permite a los equipos de *software* gestionar todos los cambios en el código fuente y así trabajar de manera más rápida y eficiente. Realiza un seguimiento de todas las modificaciones del código en una especie de base de datos, de forma que si ocurre un error se puede volver a una versión anterior.

Git se basa en un repositorio y ramas. El repositorio es un espacio de alojamiento virtual donde se encuentra el código del proyecto. Una rama es una copia exacta del proyecto y se

## **6.5. DESARROLLO DEL PROYECTO**

---

crea a partir de otra rama. Cuando se quiera añadir una nueva funcionalidad o realizar una corrección se genera una nueva rama para encapsular los cambios realizados.

Las principales operaciones en Git son *pull* para descargar al entorno local los últimos cambios de una determinada rama del repositorio, *commit* para guardar los cambios realizados en local y *push* para subir al repositorio remoto los cambios realizados en local guardados en los *commits*. Previamente, *add* permite ver los cambios realizados sobre los archivos en local antes de hacer *commit* para guardar estos cambios. Una vez que se compruebe que una rama tiene la funcionalidad deseada, se realiza *merge* para añadir los cambios realizados en la rama a otra rama principal que recoja todos los avances.

Git cuenta con una arquitectura distribuida, de manera que no tienen un único espacio para toda la historia de versiones del *software*, sino que la copia de trabajo del código de cada desarrollador consiste también en un repositorio que alberga todos los cambios realizados.

Las principales ventajas de Git son el rendimiento, la seguridad y la flexibilidad. Git está diseñado para conservar la integridad del código fuente gestionado, por lo tanto, todos los contenidos almacenados en un repositorio de Git están protegidos con un algoritmo de *hash*. Git además es flexible ya que es compatible con muchos sistemas y proyectos, permite varios tipos de flujos de trabajo y es eficiente tanto para proyectos grandes como pequeños.

Se ha utilizado Git en su versión 2.35.1 para este proyecto.

### **6.5.2. Gitlab**

Gitlab [63] es un servicio web de control de versiones y *DevOps* de código abierto basado en Git. Permite el desarrollo colaborativo y permite gestionar, crear, administrar y conectar repositorios de Git. Además permite a los desarrolladores gestionar y realizar las diferentes tareas del proyecto mediante múltiples funcionalidades que aumentarán la eficiencia y velocidad de trabajo en el proyecto. Por lo tanto, sirve tanto como sistema de control de versiones como herramienta de gestión de proyectos.

Se ha utilizado Gitlab como herramienta de gestión del repositorio de Git y como herramienta de gestión del proyecto para controlar las tareas de desarrollo debido a la gran cantidad de herramientas que proporciona y que la Escuela de Ingeniería Informática proporciona licencia para su uso.

### **6.5.3. Astah**

Astah [64] es una herramienta de modelado UML que permite crear gran variedad de diagramas con las necesidades del sistema que se quiera construir. Contiene numerosas herramientas para que la creación de estos diagramas sea más eficiente. Ha sido utilizada para realizar los Diagramas UML y la Escuela de Ingeniería Informática proporciona licencia a los estudiantes en versión profesional lo que da numerosas ventajas.

En este proyecto se ha utilizado la versión de Astah Professional 9.0.

#### 6.5.4. Railway

Railway [65] es una plataforma como servicio que se utiliza para desplegar en la nube las aplicaciones desarrolladas. Una plataforma como servicio [66] consiste en un conjunto de servicios que están basados en la nube y que permite a los desarrolladores crear aplicaciones sin preocuparse por la configuración y el mantenimiento de los servidores, de manera que se tienen que centrar únicamente en crear la mejor experiencia para el Usuario posible.

Railway ha sido la tecnología utilizada para el despliegue de la aplicación ya que permite el despliegue tanto de la parte de *frontend* como la de *backend* y proporciona una versión gratuita con las capacidades suficientes como para desplegar la aplicación desarrollada.

#### 6.5.5. Cloudinary

Cloudinary [67] provee un servicio para almacenar imágenes y vídeos en la nube. Permite a sus usuarios almacenar, modificar y recuperar imágenes y vídeos para que se puedan utilizar en una aplicación. Este servicio en la nube almacena las imágenes en su servidor y permite el acceso a ellas mediante una URL personalizada. La cuenta gratuita permite almacenar fotos de hasta 10MB.

Se escogió Cloudinary para el almacenamiento de las fotos de la aplicación debido a que es una herramienta ampliamente utilizada por los desarrolladores, por lo que cuenta con una extensa comunidad y la amplia capacidad de almacenamiento que proporciona su plan gratuito.

#### 6.5.6. JWT

*Json Web Token* [68] es un estándar abierto RFC 7519 en el que se define un método para comunicar datos entre dos entidades de manera segura. De esta forma se puede transmitir información de manera segura que es verificada ya que se firma de manera virtual. Está compuesto por demandas en las que se realiza la transmisión de la información de una entidad a otra y la estructura de un *Json Web Token* es una cadena compuesta de tres partes separadas por un punto que se serializa. Las partes son las siguientes:

- *Header*: es el primer componente y está compuesto por el tipo del *token* que en este caso es JWT y el algoritmo utilizado que en este caso es SHA256.
- *Payload*: es donde se encuentran las demandas antes comentadas del *token*. Generalmente, son sobre una entidad y otra información asociada y pueden ser registradas, públicas o privadas. Las registradas no son obligatorias y las privadas sirven para compartir información.
- *Signature*: es donde se debe firmar el *header* y el *payload* codificado, el *secret* y el algoritmo que se definió en el *header*. Esta parte tiene como finalidad verificar que no haya ningún cambio en el contenido.

## **6.5. DESARROLLO DEL PROYECTO**

---

Las principales ventajas son su tamaño compacto y por lo tanto se puede utilizar para cualquier tipo de datos. Además es autocontenido, de manera que tiene toda la información necesaria en su interior y que se puede utilizar en varias plataformas.

Su uso principal es para el intercambio de información entre dos sitios, de manera que, al ser firmado digitalmente, se puede verificar la información. En este proyecto se ha utilizado para la autenticación de usuarios de manera que las contraseñas se encripten para aumentar la seguridad.

### **6.5.7. Trello**

Trello [69] es una aplicación para gestionar proyectos en línea que ayuda a equipos que pueden estar compuestos por grandes números de personas y en diferentes localizaciones, organizar y priorizar sus tareas para así colaborar en tiempo real hacia la consecución del proyecto. Su interfaz utiliza tableros y tarjetas para representar cada proyecto y tarea lo que ayuda a que se tenga una buena visibilidad de lo que falta por hacer y una mejor organización. Debido a estas ventajas es utilizado por pequeñas *startups* hasta grandes empresas del sector. Trello implementa un tablero Kanban.

Se ha escogido esta herramienta para la organización de las tareas a realizar en el proyecto debido a su interfaz gráfica que facilita la rápida visualización de qué tareas faltan por hacer. Además su fácil acceso desde cualquier sitio por cualquier persona relacionada con el proyecto y su experiencia previa con la utilización de esta tecnología han sido determinantes en su elección.

### **6.5.8. Visual Studio Code**

Visual Studio Code [70] es un editor de código fuente desarrollado por Microsoft para Windows, MacOS, Linux y Web. Facilita el desarrollo de código mediante múltiples herramientas como el soporte para la depuración, control integrado de Git, resaltado de sintaxis o refactorizaciones de código. Es altamente personalizable, lo que permite que se pueda adaptar y contiene gran cantidad de extensiones lo que facilitan la tarea de desarrollo. Es de código abierto y cuenta con una gran comunidad de usuarios, debido a que es uno de los editores más utilizado entre los desarrolladores.

Se ha utilizado este editor de código para todo el proyecto, tanto la parte de *backend* como *frontend* como la memoria. Para la parte de *frontend* en Angular, Visual Studio Code proporciona numerosas extensiones que facilitan el desarrollo en este *framework*. Para la parte de *backend* en NodeJS, Visual Studio Code al contar con línea de comandos integrada facilita la depuración y corrección de los errores. Finalmente para la memoria, Visual Studio Code permite la edición de archivos en Latex.

### 6.5.9. Balsamiq

Balsamiq Wireframes [71] es una herramienta *online* que se utiliza para crear *wireframes*, es decir, esquemas de las páginas que contendrá la aplicación Web y que servirán como guía del esqueleto que tendrán las interfaces de la aplicación Web.

Se ha utilizado para crear los bocetos de las interfaces que contendrá la aplicación Web debido a que se ha trabajado previamente con esta herramienta y que es de gran popularidad entre los desarrolladores para crear bocetos de interfaces que luego se plasmen en una aplicación. Además contiene gran cantidad de herramientas que pueden ayudar a la creación de interfaces que luego hagan que la experiencia del Usuario sea mejor al utilizar la aplicación Web.

## *6.5. DESARROLLO DEL PROYECTO*

---

## Capítulo 7

# Implementación y pruebas

### 7.1. Introducción

En este capítulo se explica la estructura del código del proyecto, tanto en la parte del *frontend* en Angular como en la parte del *backend* con NodeJS. Además, se explican los principales problemas encontrados durante el desarrollo del proyecto y los diferentes tipos de pruebas realizadas para comprobar el correcto funcionamiento de la aplicación Web.

### 7.2. Estructura del código de *frontend*

En esta sección se detalla la estructura de ficheros en la parte realizada en Angular para el *frontend*. Hay ciertas carpetas en las que no se muestra su estructura debido a su similitud con otras. En la Figura 7.1 se muestra esta estructura.

A continuación se detalla el contenido de las carpetas mostradas en esta estructura:

- **node\_modules**: contiene los paquetes instalados que se usan en Angular.
- **app**: contiene todo el código del proyecto. Dentro de esta carpeta destacan los archivos:
  - **app-routing.module.ts**: contiene las rutas de acceso para cada componente gráfico de la interfaz de la aplicación, es decir, la ruta que aparecerá en la barra del navegador para acceder a cada componente de la interfaz.
  - **app.module.ts**: donde se importan todos los componentes de la aplicación y los módulos que se van a utilizar.

Además la carpeta **app** contiene tres carpetas y el contenido que almacena cada una de ellas es el siguiente:

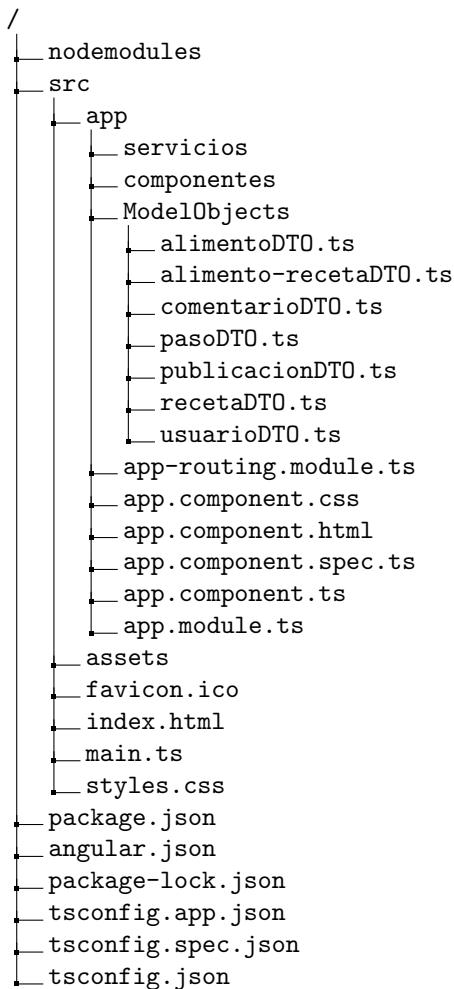


Figura 7.1: Estructura de ficheros en la parte del *frontend* en Angular.

- **ModelObjects:** que contiene los DTO utilizados para gestionar los datos introducidos por el Usuario y obtenidos del *backend* en el *frontend*.
- **servicios:** contiene carpetas que almacenan los servicios que se comunicarán con el *backend* para obtener la información de la base de datos y guardar cambios sobre la información almacenada en ella. Hay varios servicios, cada uno asociado a una tabla de la base de datos, y almacenarán las peticiones que se realicen al *backend* y que afecten a la tabla de la base de datos asociada. Cada una de estas carpetas almacena dos archivos:
  - Un archivo terminado en `service.spec.ts` que sirve para realizar test unitarios.
  - Un archivo terminado en `service.ts` en el que se almacenan los métodos que realizan las peticiones al *backend*.
- **componentes:** almacena los componentes gráficos de la interfaz de la aplicación Web. Al ser un número elevado de componentes de la interfaz, se han agrupado según su funcionalidad. La descripción de estas carpetas que se han utilizado para agrupar los componentes es la siguiente:
  - **admin-vistas:** en esta carpeta se encuentran todos los componentes que podrá ver el Administrador cuando se identifique en la aplicación.
  - **buscadores:** contiene dos carpetas:
    - **componentes-buscar:** donde se encuentran las vistas en las que se puede realizar una búsqueda de un determinado elemento. Estas interfaces son reutilizables ya que utilizadas por otros componentes.
    - **vistas-buscar:** contiene los componentes que solo se utilizan para realizar búsquedas y por lo tanto estos componentes utilizan el componente de la carpeta **componentes-buscar** que corresponda.
  - **cartas:** en esta carpeta se agrupan los componentes que sirven como cartas para representar la información más relevante de una instancia de un determinado tipo de datos. Estas cartas son utilizadas por varias interfaces en la aplicación.
  - **creadores:** en esta carpeta se pueden encontrar a las interfaces que servirán para crear un contenido por parte del Usuario y que se almacenará en la base de datos.
  - **detalles:** en esta carpeta se encuentran las vistas que sirven para mostrar toda la información de un elemento concreto de entre todos los tipos de datos diferentes que se manejan en la aplicación.
  - **gestiones-usuario:** en esta carpeta se encuentran los componentes que muestran los datos almacenados del Usuario y la modificación de estos.
  - **sin-identificar:** en esta carpeta se encuentran las vistas que serán utilizadas por los usuarios antes de que se identifiquen en la aplicación.

En la Figura 7.2 se muestra la estructura de ficheros dentro de la carpeta **servicios**.

En la Figura 7.3 se muestra la estructura de ficheros dentro de la carpeta **componentes**.

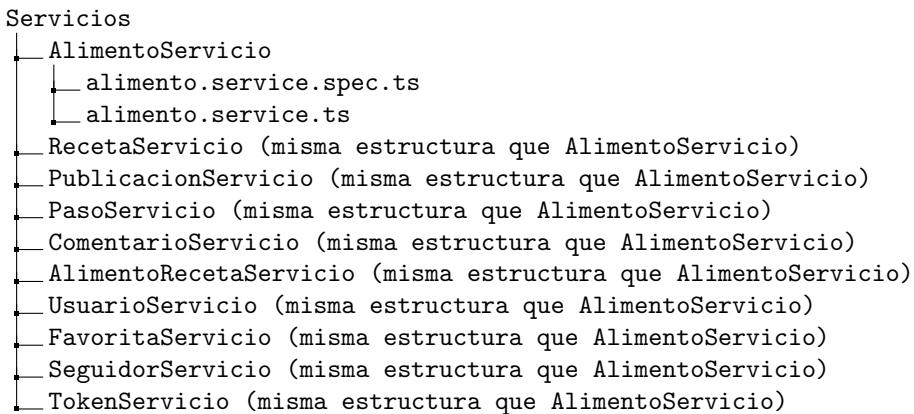


Figura 7.2: Estructura de ficheros en la carpeta `servicios`.

### 7.3. Estructura del código de *backend*

En la Figura 7.4 se muestra la estructura de la parte realizada en NodeJS para el *backend*.

A continuación se detalla el contenido de las carpetas y archivos mostrados en esta estructura:

- Carpeta `config` que contiene el siguiente archivo:
  - `config.js`: que se usa para la conexión con la base de datos SQL.
- Carpeta `images`: se utiliza como carpeta auxiliar para guardar las imágenes temporalmente que suba el usuario.
- Carpeta `node_modules`: contiene los paquetes instalados que se usan en NodeJS.
- Archivo `index.js`: es el archivo que se ejecuta para poner en funcionamiento esta parte del *backend* de manera que empiece a escuchar peticiones en el puerto definido.
- Carpeta `src`: contiene el código principal de la parte de *backend* y la descripción de las carpetas en su interior es la siguiente:
  - `controlador`: esta carpeta contiene el siguiente archivo:
    - `controlador.js`: en él se definen las rutas del *backend* de la API REST a las que el *frontend* puede realizar peticiones. Estas rutas se definen mediante el uso de ExpressJS y cada ruta llamará un método de un DAO de los almacenados en la carpeta DAO dependiendo de la operación solicitada y sobre qué tipo de datos se debe realizar.
  - `middleware`: en esta carpeta se encuentran los archivos que ayudan a almacenar las fotos que adjunte el Usuario a cada determinado tipo de datos. Estos archivos son los siguientes:

```
Componentes
└ admin-vista
    └ tabla-alimento
        └ tabla-alimento.component.css
        └ tabla-alimento.component.html
        └ tabla-alimento.component.spec.ts
        └ tabla-alimento.component.ts
    └ tabla-usuario (misma estructura que tabla-alimento)
    └ tabla-publicacion (misma estructura que tabla-alimento)
    └ tabla-receta (misma estructura que tabla-alimento)
    └ vista-admin (misma estructura que tabla-alimento)
└ buscadores
    └ componentes-buscar
        └ buscador-alimento (misma estructura que tabla-alimento)
        └ buscador-receta (misma estructura que tabla-alimento)
        └ buscador-usuario (misma estructura que tabla-alimento)
    └ vistas-buscar
        └ buscar-alimento (misma estructura que tabla-alimento)
        └ buscar-receta (misma estructura que tabla-alimento)
        └ buscar-usuario (misma estructura que tabla-alimento)
└ cartas
    └ carta-alimento (misma estructura que tabla-alimento)
    └ carta-alimento-receta (misma estructura que tabla-alimento)
    └ carta-publicacion (misma estructura que tabla-alimento)
    └ carta-usuario (misma estructura que tabla-alimento)
    └ carta-receta (misma estructura que tabla-alimento)
    └ carta-comentario (misma estructura que tabla-alimento)
└ creadores
    └ crear-alimento (misma estructura que tabla-alimento)
    └ crear-publicacion (misma estructura que tabla-alimento)
    └ crear-receta (misma estructura que tabla-alimento)
└ detalles
    └ detalle-alimento (misma estructura que tabla-alimento)
    └ detalle-publicacion (misma estructura que tabla-alimento)
    └ detalle-receta (misma estructura que tabla-alimento)
    └ detalle-usuario (misma estructura que tabla-alimento)
└ gestiones-usuario
    └ editar-perfil (misma estructura que tabla-alimento)
    └ seguidores-usuario (misma estructura que tabla-alimento)
    └ seguidos-usuario (misma estructura que tabla-alimento)
    └ favoritas-recetas (misma estructura que tabla-alimento)
    └ muro-publicaciones (misma estructura que tabla-alimento)
└ sin-identificar
    └ identificacion-usuario (misma estructura que tabla-alimento)
    └ registro-usuario (misma estructura que tabla-alimento)
    └ descripcion-aplicacion (misma estructura que tabla-alimento)
```

Figura 7.3: Estructura de los ficheros en la carpeta componentes.

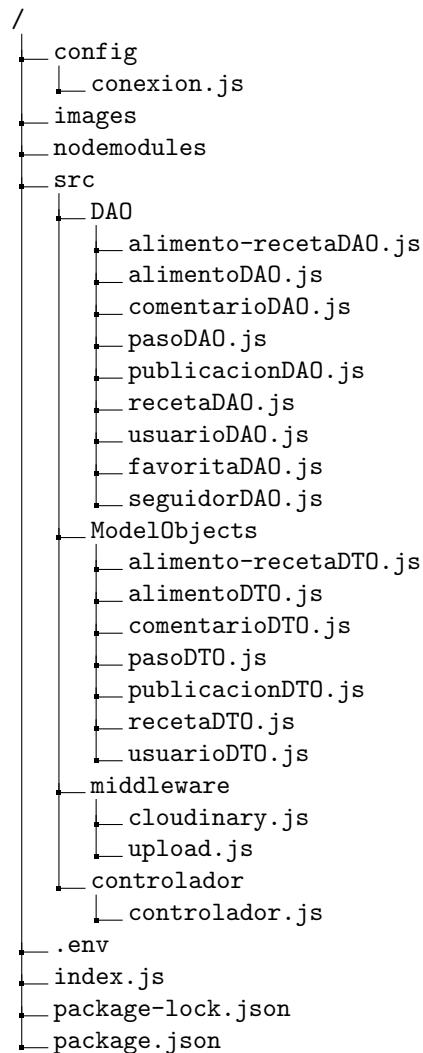


Figura 7.4: Estructura de los ficheros en la parte del *backend* en NodeJS.

- `upload.js`: que almacena las fotos que adjunte el Usuario de manera temporal en la carpeta `images`.
- `cloudinary.js`: que se encarga de subir las fotos a Cloudinary para que queden almacenadas en la nube y se pueda acceder a ellas.
- **DAO**: esta carpeta contiene los DAO para cada tipo de datos que se maneja en la aplicación. Cada DAO almacena las operaciones relacionadas sobre la base de datos para cada tipo de datos, de manera que en cada DAO se pueden encontrar agrupadas todas las operaciones sobre una determinada tabla de la base de datos.
- **ModelObjects**: esta carpeta contiene los DTO que se usarán para manejar los datos en el *backend* que se obtengan de la base de datos y para enviárselos al *frontend* en el formato correcto.

## 7.4. Pruebas

Antes de que una aplicación Web sea utilizada por los clientes finales, es importante que sea verificada y validada a fondo para asegurarse de que todo funciona correctamente y de acuerdo a los objetivos de usuario. A continuación se definen los tipos de pruebas que se van a realizar:

- **Prueba de funcionalidad**: estas pruebas comprueban que la aplicación Web se comporta según lo esperado sin problemas y la aplicación cumple con las especificaciones. Estas pruebas se encargan de comprobar la conexión entre el *backend* y *frontend*, los enlaces dentro de la aplicación Web y la información enviada u obtenida por el usuario. Son las principales para una aplicación Web porque verificarán si puede ejecutarse correctamente. Pueden realizarse manualmente o por un *software* de prueba. La aplicación de estas pruebas a este proyecto se van a centrar en verificar que los enlaces funcionen correctamente y dirijan a la página deseada, que los formularios funcionen correctamente, verificando la validez de los campos introducidos por el usuario. Además, comprobarán que las *cookies* almacenan los datos adecuados, que los archivos HTML y CSS del proyecto funcionan correctamente, que las interfaces se comportan como se espera y que el flujo de trabajo con la aplicación Web, es decir, como usaría la aplicación Web un Usuario final, funciona correctamente. Aquí es importante incluir escenarios inesperados de manera que la aplicación muestre los mensajes adecuados cuando el Usuario realice una acción que no es válida en la aplicación Web.
- **Pruebas de usabilidad**: estas pruebas sitúan la experiencia del Usuario en el centro de diseño de la aplicación. El objetivo principal de cualquier aplicación Web es atraer clientes, por lo tanto, estas pruebas se centran en comprobar como de fácil de utilizar es la aplicación Web. Estas pruebas se deben centrar en ver la facilidad con la que un Usuario puede navegar por la aplicación Web, si las funcionalidades que se ofrecen son fáciles de utilizar y si el contenido de la aplicación se muestra de una manera adecuada al Usuario para su uso. Para estas pruebas, la mejor opción es reclutar usuarios que puedan probar la aplicación definiendo ciertas acciones a realizar para comprobar si los usuarios pueden realizar lo que se pide.

- **Pruebas de compatibilidad:** estas pruebas verifican que la aplicación sea compatible con todo tipo de dispositivos y navegadores. Para ello es importante verificar que la aplicación Web sea compatible con los principales navegadores como Google Chrome o Firefox, de manera que los usuarios puedan utilizar la aplicación en estos navegadores sin retrasos ni errores. Por otro lado, también es importante comprobar que la aplicación web funcione en todo tipo de sistemas operativos debido a la variedad de sistemas operativos que existen. Por último, pero un aspecto fundamental, es verificar que la aplicación sea compatible con todo tipo de dispositivos, de manera que funcione bien con todo tipo de tamaño de pantallas, debido a que cada Usuario utilizará un dispositivo diferente con un tipo tamaño de pantalla único.

Para las pruebas de funcionalidad, el propio desarrollador verificará que la aplicación se comporta como debe y cumple con las especificaciones. Para las pruebas de usabilidad se utilizarán usuarios cuyos perfiles sean similares a los perfiles de los usuarios finales de la aplicación para comprobar cómo se desenvuelven al utilizar la aplicación y si les resulta fácil utilizarla. Para las pruebas de compatibilidad, la aplicación se utilizará en diferentes navegadores, sistemas operativos y dispositivos para comprobar que en todos los casos funciona correctamente.

## 7.5. Problemas y dificultades encontradas

A lo largo del desarrollo del proyecto se han ido encontrando diferentes problemas que han supuesto desafíos para la consecución con éxito del proyecto, pero finalmente se han podido encontrar solución a cada uno de ellos, aunque en cierta medida pueden haber retrasado el avance del proyecto. Los principales problemas afrontados han sido los siguientes:

- **Almacenamiento de las fotos:** uno de los aspectos fuertes de cualquier aplicación es la capacidad que tiene el Usuario de adjuntar fotos para que el contenido que cree el Usuario sea más descriptivo. Para este proyecto este era también un aspecto importante y desde un primer momento, que se pudiesen adjuntar imágenes era un requisito indispensable para esta aplicación. Esta funcionalidad ha sido difícil de implementar debido a que hay variedad de opciones disponibles, cada una con sus ventajas y contras. La opción que se ha escogido es almacenar las fotos en la nube y guardar el enlace de acceso a cada foto en el objeto que tiene asociada esa foto en la base de datos, de manera que solo se necesite este enlace para acceder a la foto.
- **Flujo de datos entre el *frontend* y *backend*:** la comunicación entre el *backend* y el *frontend* no ha sido sencilla de implementar debido a que las tablas almacenadas en la base de datos contenían numerosas relaciones lo que hacía que cada clase del Modelo de Dominio no contuviese toda la información relevante y se necesitasen de clases DTO con atributos añadidos para representar esta información. Este problema de comunicación entre el *frontend* y *backend* se ha resuelto utilizando clases DTO con atributos añadidos respecto a las clases del Modelo de Dominio para almacenar información relevante de las relaciones entre las tablas de la base de datos. Además se han utilizado consultas SQL complejas para obtener los datos necesarios.

- **Crecimiento de la aplicación y organización de su contenido:** este proyecto se divide en la parte de *frontend* con Angular y la parte de *backend* en NodeJS y ambas partes contienen gran cantidad de archivos de varios tipos agrupados en numerosas carpetas. En el momento que esta aplicación empezó a crecer en funcionalidad y el número de archivos que contenía empezó a aumentar, se volvió un aspecto fundamental organizar todos estos archivos en carpetas para que la aplicación pudiese seguir creciendo y se pudiese realizar la depuración de errores más fácilmente.

## 7.6. Casos de Prueba

Las Tablas 7.1 a 7.23 muestran los casos de prueba.

CP-01	Iniciar sesión
Descripción	El Usuario se identifica en la aplicación.
Entrada	<ul style="list-style-type: none"><li>■ <i>username: chema11</i></li><li>■ contraseña: <i>123456789</i></li></ul>
Escenario	Introduce el <i>username</i> y contraseña definido en la entrada y pulsa el botón <i>identificarse</i> .
Resultado esperado	El Usuario se ha identificado en la aplicación y se redirige a la vista <i>muro-publicaciones</i> .
Trazabilidad	CU-01 Iniciar sesión.

Tabla 7.1: CP-01. Iniciar sesión.

## 7.6. CASOS DE PRUEBA

---

<b>CP-02</b>	<b>Registrarse</b>
Descripción	El Usuario se registra en la aplicación.
Entrada	<ul style="list-style-type: none"> <li>■ <i>username: chema99</i></li> <li>■ contraseña: <i>123456789</i></li> <li>■ <i>email: chema99@gmail.com</i></li> </ul>
Escenario	Introduce el <i>username</i> , contraseña y <i>email</i> definido en la entrada y pulsa el botón <i>registrarse</i> .
Resultado esperado	El Usuario se ha registrado en la aplicación y se redirige a la vista <i>identificacion-usuario</i> .
Trazabilidad	CU-02 Registrarse.

Tabla 7.2: CP-02. Registrarse.

<b>CP-03</b>	<b>Cerrar sesión</b>
Descripción	El Usuario cierra su sesión en la aplicación.
Precondición	El Usuario está identificado en el sistema.
Escenario	El Usuario pulsa el botón <i>cerrar sesión</i> .
Resultado esperado	El Usuario ha cerrado su sesión y se redirige a la vista <i>identificacion-usuario</i> .
Trazabilidad	CU-03 Cerrar sesión.

Tabla 7.3: CP-03. Cerrar sesión.

CP-04	Editar información del usuario
Descripción	El Usuario edita la información de su cuenta.
Entrada	<ul style="list-style-type: none"><li>■ foto: <i>pruebas.jpg</i></li><li>■ contraseña: <i>abcdefghijklm</i></li><li>■ email: <i>chema77@gmail.com</i></li><li>■ descripción: <i>perfil de pruebas para la aplicación</i></li></ul>
Escenario	Introduce la foto, contraseña, <i>email</i> y descripción definidas en la entrada y pulsa el botón <i>editar perfil</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario ha cambiado los datos de su cuenta y se redirige a la vista <i>perfil-usuario</i> .
Trazabilidad	CU-04 Editar información del Usuario.

Tabla 7.4: CP-04. Editar información del Usuario.

## 7.6. CASOS DE PRUEBA

---

<b>CP-05</b>	<b>Crear receta</b>
Descripción	El Usuario crea una receta.
Entrada	<ul style="list-style-type: none"> <li>■ título: <i>receta de prueba</i></li> <li>■ resumen: <i>esta receta saludable sirve para probar la aplicación</i></li> <li>■ tiempo: <i>20</i></li> <li>■ foto: <i>pruebas.png</i></li> <li>■ dificultad: <i>fácil</i></li> <li>■ pasos: <ul style="list-style-type: none"> <li>• <i>Primero se ponen a cocer los macarrones</i></li> <li>• <i>Después, se mezcla tomate con carne picada en una sartén</i></li> <li>• <i>Finalmente se juntan los macarrones con el tomate y la carne picada</i></li> </ul> </li> <li>■ alimentos en la receta: <ul style="list-style-type: none"> <li>• <i>Macarrones, 300 gramos</i></li> <li>• <i>Tomate, 200 gramos</i></li> <li>• <i>Carne picada, 100 gramos</i></li> </ul> </li> </ul>
Escenario	Introduce el título, resumen, tiempo, foto, dificultad, pasos y alimentos en la receta y pulsa el botón <i>crear receta</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario ha creado una receta y se redirige a la vista <i>muro-publicaciones</i> .
Trazabilidad	CU-06 Crear receta.

Tabla 7.5: CP-05. Crear receta.

<b>CP-06</b>	<b>Ver publicaciones usuarios seguidos</b>
Descripción	El Usuario ve las publicaciones de los usuarios a los que sigue.
Entrada	<ul style="list-style-type: none"> <li>■ <i>username: chema11</i></li> <li>■ contraseña: <i>123456789</i></li> </ul>
Escenario	Introduce el <i>username</i> y contraseña definidos en la entrada y pulsa <i>identificarse</i> para iniciar sesión en la aplicación.
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario se ha identificado en la aplicación y se redirige a la vista <i>muro-publicaciones</i> donde se muestran las publicaciones de los usuarios a los que sigue.
Trazabilidad	CU-05 Ver publicaciones usuarios seguidos.

Tabla 7.6: CP-06. Ver publicaciones usuarios seguidos.

<b>CP-07</b>	<b>Buscar alimento</b>
Descripción	El Usuario busca un alimento introduciendo su nombre.
Entrada	<ul style="list-style-type: none"> <li>■ nombre: <i>brócoli</i></li> </ul>
Escenario	Introduce el nombre del alimento definido en la entrada y pulsa el botón <i>buscar</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se muestran todos los alimentos con un nombre similar al introducido por el usuario.
Trazabilidad	CU-07 Buscar alimento.

Tabla 7.7: CP-07. Buscar alimento.

## 7.6. CASOS DE PRUEBA

---

<b>CP-08</b>	<b>Buscar receta</b>
Descripción	El Usuario busca una receta introduciendo su título.
Entrada	<ul style="list-style-type: none"> <li>■ título: <i>macarrones</i></li> </ul>
Escenario	Introduce el título de la receta definido en la entrada y pulsa el botón <i>buscar</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se muestran todas las recetas con un título similar al introducido por el usuario.
Trazabilidad	CU-08 Buscar receta.

Tabla 7.8: CP-08. Buscar receta.

<b>CP-09</b>	<b>Crear publicación</b>
Descripción	El Usuario crea una publicación.
Entrada	<ul style="list-style-type: none"> <li>■ título: <i>publicación de prueba</i></li> <li>■ descripción: <i>esta publicación servirá para probar la aplicación</i></li> <li>■ foto: <i>pruebas.png</i></li> <li>■ receta: <i>macarrones con tomate</i></li> </ul>
Escenario	Introduce el título, descripción, foto y receta definidos en la entrada y pulsa el botón <i>crear publicación</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario ha creado una publicación y se redirige a la vista <i>muro-publicaciones</i> .
Trazabilidad	CU-09 Crear publicación.

Tabla 7.9: CP-09. Crear publicación.

<b>CP-10</b>	<b>Buscar Usuario</b>
Descripción	El Usuario busca un Usuario introduciendo su <i>username</i> .
Entrada	■ <i>username: javier89</i>
Escenario	Introduce el <i>username</i> definido en la entrada y pulsa el botón <i>buscar</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se muestran todos los usuarios con un <i>username</i> similar al introducido por el usuario.
Trazabilidad	CU-10 Buscar Usuario.

Tabla 7.10: CP-10. Buscar usuario.

<b>CP-11</b>	<b>Consultar recetas favoritas</b>
Descripción	El Usuario consulta sus recetas marcadas como favoritas.
Escenario	El Usuario pulsa el botón <i>ver recetas favoritas</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se muestran las recetas marcadas como favoritas por el Usuario.
Trazabilidad	CU-12 Consultar recetas favoritas.

Tabla 7.11: CP-11. Consultar recetas favoritas.

<b>CP-12</b>	<b>Borrar receta de favoritas</b>
Descripción	El Usuario elimina una receta de sus favoritas.
Escenario	El Usuario pulsa el botón <i>eliminar receta de favoritas</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se elimina la receta de las favoritas del usuario.
Trazabilidad	CU-13 Borrar receta de favoritas.

Tabla 7.12: CP-12. Borrar receta de favoritas.

## 7.6. CASOS DE PRUEBA

---

<b>CP-13</b>	<b>Añadir receta a favoritas</b>
Descripción	El Usuario añade la receta a sus favoritas.
Escenario	El Usuario pulsa el botón <i>añadir receta a favoritas</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se añade la receta a las favoritas del usuario.
Trazabilidad	CU-11 Añadir receta a favoritas.

Tabla 7.13: CP-13. Añadir receta a favoritas.

<b>CP-14</b>	<b>Consultar usuarios seguidos</b>
Descripción	El Usuario consulta sus usuarios seguidos.
Escenario	El Usuario pulsa el botón <i>ver usuarios seguidos</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	Se muestran los usuarios seguidos por el usuario.
Trazabilidad	CU-14 Consultar usuarios seguidos.

Tabla 7.14: CP-14. Consultar usuarios seguidos.

<b>CP-15</b>	<b>Dejar de seguir Usuario</b>
Descripción	El Usuario deja de seguir a un Usuario.
Escenario	El Usuario pulsa el botón <i>dejar de seguir usuario</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario ya no sigue al Usuario.
Trazabilidad	CU-15 Dejar de seguir Usuario.

Tabla 7.15: CP-15. Dejar de seguir Usuario.

<b>CP-16</b>	<b>Seguir Usuario</b>
Descripción	El Usuario empieza a seguir a un Usuario.
Escenario	El Usuario pulsa el botón <i>seguir usuario</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El Usuario seleccionado se añade a la lista de usuarios seguidos por el Usuario.
Trazabilidad	CU-16 Seguir Usuario.

Tabla 7.16: CP-16. Seguir Usuario.

<b>CP-17</b>	<b>Añadir comentario a publicación</b>
Descripción	El Usuario añade un comentario a una publicación.
Entrada	<ul style="list-style-type: none"> <li>■ comentario: <i>buenas recetas, tengo que probarla</i></li> </ul>
Escenario	Introduce el comentario definido en la entrada y pulsa el botón <i>guardar comentario</i> .
Precondición	El Usuario está identificado en el sistema.
Resultado esperado	El comentario se añade a la lista de comentarios de la publicación.
Trazabilidad	CU-17 Añadir comentario a publicación.

Tabla 7.17: CP-17. Añadir comentario a publicación.

<b>CP-18</b>	<b>Eliminar Usuario</b>
Descripción	El Administrador elimina un Usuario.
Escenario	El Administrador pulsa el botón <i>eliminar usuario</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se elimina el Usuario y todas sus publicaciones, recetas y comentarios creados.
Trazabilidad	CU-18 Eliminar Usuario.

Tabla 7.18: CP-18. Eliminar Usuario.

## 7.6. CASOS DE PRUEBA

---

<b>CP-19</b>	<b>Eliminar publicación</b>
Descripción	El Administrador elimina una publicación.
Escenario	El Administrador pulsa el botón <i>eliminar publicación</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se elimina la publicación y todos sus comentarios.
Trazabilidad	CU-19 Eliminar publicación.

Tabla 7.19: CP-19. Eliminar publicación.

<b>CP-20</b>	<b>Eliminar alimento</b>
Descripción	El Administrador elimina un alimento.
Escenario	El Administrador pulsa el botón <i>eliminar alimento</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se elimina el alimento, todas sus publicaciones asociadas y todas las apariciones del alimento como ingrediente de una receta.
Trazabilidad	CU-20 Eliminar alimento.

Tabla 7.20: CP-20. Eliminar alimento.

<b>CP-21</b>	<b>Eliminar receta</b>
Descripción	El Administrador elimina una receta.
Escenario	El Administrador pulsa el botón <i>eliminar receta</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se elimina la receta, todas sus publicaciones asociadas y los alimentos definidos como ingredientes para su preparación.
Trazabilidad	CU-22 Eliminar receta.

Tabla 7.21: CP-21. Eliminar receta.

CP-22	Crear alimento
Descripción	El Administrador crea un alimento.
Entrada	<ul style="list-style-type: none"> <li>■ nombre: <i>macarrones</i></li> <li>■ descripción: <i>pasta con muchos carbohidratos</i></li> <li>■ calorías: <i>109</i></li> <li>■ foto: <i>macarrones.jpg</i></li> <li>■ enlace: <i>https://carrefour.es/macarrones1kg</i></li> <li>■ grasas: <i>23</i></li> <li>■ carbohidratos: <i>60</i></li> <li>■ proteínas: <i>17</i></li> <li>■ cantidad: <i>100</i></li> <li>■ medida: <i>gramos</i></li> </ul>
Escenario	Introduce el nombre, descripción, calorías, foto, enlace, grasas, carbohidratos, proteínas, cantidad y medida y pulsa el botón <i>crear alimento</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se crea el alimento en la aplicación y el Usuario se redirige a la vista <i>vista-admin</i> .
Trazabilidad	CU-23 Crear alimento.

Tabla 7.22: CP-22. Crear alimento.

## 7.6. CASOS DE PRUEBA

---

<b>CP-23</b>	<b>Editar información de alimento</b>
Descripción	El Administrador edita la información de un alimento.
Entrada	<ul style="list-style-type: none"> <li>■ carbohidratos: 70</li> <li>■ grasas: 13</li> <li>■ foto: <i>macarrones1.jpg</i></li> </ul>
Escenario	Introduce los carbohidratos, grasas y foto y pulsa el botón <i>editar alimento</i> .
Precondición	El Administrador está identificado en el sistema.
Resultado esperado	Se actualizan los datos del alimento y el Administrador se redirige a la vista <i>vista-admin</i> .
Trazabilidad	CU-21 Editar información de alimento.

Tabla 7.23: CP-23. Editar información de alimento.

## Capítulo 8

# Seguimiento del proyecto

### 8.1. Introducción

En este capítulo se explica cómo se ha desarrollado el proyecto con respecto a la planificación inicial. Se define su evolución con las tareas que se han realizado en cada *sprint* ya que se ha aplicado el marco de trabajo ágil Scrum. Además se exponen los problemas encontrados en los *sprints*, los riesgos que han aparecido y qué mejoras se han propuesto en las *Sprint Review*.

### 8.2. Seguimiento del proyecto

Para cada tarea realizada en el proyecto se ha definido un código para clasificarla según el tipo del que sea:

- **DOC.** Tareas relacionadas con la documentación del proyecto.
- **DEV.** Tareas relacionadas con el desarrollo de la aplicación del proyecto.
- **TEST.** Tareas relacionadas con la realización de pruebas sobre la aplicación del proyecto.
- **BUG.** Tareas relacionadas con la corrección de errores en la aplicación del proyecto.
- **REU.** Tareas relacionadas con reuniones con los tutores.

Para cada tarea dentro de cada *sprint* se define su identificador único, su tipo, una descripción de en qué consiste la tarea, el tiempo empleado para completarla y su estado al finalizar el *sprint*. Los estados que puede tener una tarea son:

## 8.2. SEGUIMIENTO DEL PROYECTO

---

- **No iniciada.** Al finalizar el *sprint*, esta tarea no se ha comenzado y por lo tanto no se ha realizado ningún trabajo de ella.
- **En progreso.** Al finalizar el *sprint*, esta tarea se ha comenzado y se ha realizado trabajo de ella pero no se ha completado.
- **Completada.** Al finalizar el *sprint*, esta tarea se ha finalizado.

### 8.2.1. Sprint 1: 30/1/2023 - 16/2/2023

Al comenzar el *sprint* 1 se ha realizado un análisis de las posibles opciones que hay para este proyecto. Se han valorado las diferentes ideas propuestas y se ha estudiado su viabilidad y posibles problemas a enfrentar. Se ha llevado a cabo una reunión con los tutores para comentar estas ideas y escoger la idea que se desarrollaría en el proyecto. Una vez escogida la idea, se han clarificado los detalles del proyecto y se han concretado los objetivos que se pretenden lograr con este proyecto. A continuación se han empezado a realizar las primeras partes de la documentación. Se ha realizado una primera versión de los capítulos de introducción, planificación, requisitos, análisis y diseño.

Se han priorizado las secciones de la documentación sobre los Casos de Uso, Modelo de Dominio, Diseño de la Base de Datos y Diseño de la Interfaz Gráfica debido a que con estas secciones realizadas se podría empezar en el siguiente *sprint* a desarrollar la aplicación.

Este *sprint* ha tenido una duración de 80 horas.

En las Tablas 8.1 y 8.2 se pueden ver las tareas que se han realizado en este *sprint*.

### 8.2.2. Sprint 2: 16/2/2023 - 5/3/2023

Durante el *sprint* 2 se ha dedicado más tiempo a avanzar en el desarrollo de las interfaces de la aplicación. Se han desarrollado las principales funcionalidades de la aplicación en base a la primera versión de los capítulos realizados de la documentación en el *sprint* previo. Con estas secciones iniciales de la documentación donde se explicaba de manera general la aplicación, se ha podido empezar a desarrollar muchas de las interfaces de la aplicación.

Para el desarrollo de identificación y registro han aparecido numerosos problemas con la utilización de JWT para que las contraseñas de los usuarios se almacensen de manera segura. Entre estos problemas cabe destacar que cuando un Usuario se registraba en la aplicación, la contraseña se encriptaba de una manera distinta cada vez. Esto se ha resuelto definiendo una clave secreta para que la contraseña se encriptase siempre siguiendo un patrón establecido para que después se pudiese recuperar.

Para las interfaces donde se debían cargar imágenes de un elemento como un alimento, una receta, un Usuario o una publicación se han encontrado varios problemas para almacenar estas imágenes en Cloudinary. Para conseguir esta funcionalidad se han realizado varias pruebas y se han probado numerosas opciones. Finalmente se ha conseguido almacenar la imagen

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T001	DOC	Búsqueda y análisis de ideas para el proyecto.	6h	Completada
T002	REU	Reunión de arranque con los tutores.	2h	Completada
T003	DOC	Documentar resumen del proyecto.	2h	Completada
T004	DOC	Documentar contexto en el capítulo de introducción.	2h	Completada
T005	DOC	Documentar objetivos en el capítulo de introducción.	2h	Completada
T006	DOC	Documentar motivación en el capítulo de introducción.	1h	Completada
T007	DOC	Documentar aplicaciones similares en el capítulo de introducción.	1h	Completada
T007	DOC	Documentar estructura de la memoria en el capítulo de introducción.	1h	Completada
T008	DOC	Documentar introducción en el capítulo de planificación.	1h	Completada
T009	DOC	Documentar Scrum en el capítulo de planificación.	2h	Completada
T010	DOC	Documentar adaptación de Scrum al proyecto en el capítulo de planificación.	1h	Completada
T011	DOC	Documentar análisis de riesgos en el capítulo de planificación.	2h	Completada
T012	DOC	Documentar presupuesto en el capítulo de planificación.	2h	Completada
T013	DOC	Documentar <i>Product Backlog</i> inicial en el capítulo de planificación.	2h	Completada
T014	DOC	Documentar introducción en el capítulo de requisitos.	1h	Completada
T015	DOC	Desarrollar y documentar Requisitos Funcionales, No Funcionales y de Información en el capítulo de requisitos.	4h	Completada
T016	DOC	Desarrollar y documentar Actores Principales en el capítulo de requisitos.	2h	Completada
T017	DOC	Desarrollar y documentar Diagrama de Casos de Uso en el capítulo de requisitos.	3h	Completada

Tabla 8.1: Tareas de *sprint* 1.

## 8.2. SEGUIMIENTO DEL PROYECTO

---

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T018	DOC	Desarrollar y documentar Descripción de los Casos de Uso en el capítulo de requisitos.	6h	Completada
T019	DOC	Documentar introducción en el capítulo de análisis.	1h	Completada
T020	DOC	Desarrollar y documentar Modelo de Dominio en el capítulo de análisis.	5h	Completada
T021	DOC	Desarrollar y documentar Modelo de Análisis en el capítulo de análisis.	5h	Completada
T022	DOC	Desarrollar y documentar Realización de Casos de Uso de Análisis en el capítulo de análisis.	5h	Completada
T023	DOC	Documentar introducción en el capítulo de diseño.	1h	Completada
T024	DOC	Desarrollar y documentar Arquitectura Lógica del Sistema en el capítulo de diseño.	2h	Completada
T025	DOC	Desarrollar y documentar Arquitectura del Cliente y del Servidor en el capítulo de diseño.	2h	Completada
T026	DOC	Desarrollar y documentar Patrones de Diseño en el capítulo de diseño.	3h	Completada
T027	DOC	Desarrollar y documentar Arquitectura Física del Sistema en el capítulo de diseño.	2h	Completada
T028	DOC	Desarrollar y documentar Diseño de la Base de Datos en el capítulo de diseño.	4h	Completada
T029	DOC	Desarrollar y documentar Diseño de la Interfaz Gráfica en el capítulo de diseño.	7h	Completada

Tabla 8.2: Tareas de *sprint* 1.

en Cloudinary enviando la imagen desde el *frontend* al *backend* y desde aquí enviando una petición POST para almacenar la imagen.

En la *Sprint Review* se ha realizado una demostración del incremento realizado durante el *sprint* y se han propuesto numerosas mejoras sobre la aplicación como mostrar todos los usuarios, alimentos o recetas en los buscadores al inicio cuando el Usuario no haya introducido ningún valor para buscar.

En este *sprint* ha aparecido el riesgo *R-01 Falta de cualificación con las tecnologías utilizadas* ya que se desconocía como se iban a almacenar las fotos cargadas en la aplicación en la nube mediante Cloudinary. Se ha aplicado un plan de contingencia que ha consistido en consultar en internet mediante vídeos y foros como almacenar y recuperar fotos en Cloudinary.

Este *sprint* ha tenido una duración de 80 horas.

En la Tabla 8.3 se pueden ver las tareas que se han realizado en este *sprint*.

### 8.2.3. Sprint 3: 5/3/2023 - 22/3/2023

En este *sprint* se ha corregido el resumen, el capítulo de introducción y el capítulo de requisitos que son de los más importantes en la memoria. Respecto a las correcciones del capítulo de requisitos, se han modificado los Requisitos Funcionales, No Funcionales y de Información definidos en el *sprint 1* para definirlos de una manera más clara y que se ajusten a lo que se va a desarrollar en la aplicación. De manera similar, el Diagrama de Casos de Uso y la Descripción de los Casos de Uso también se han corregido para exponer más claramente la funcionalidad de la aplicación. El hecho de haber desarrollado en el *sprint 2* las interfaces ha ayudado a refinar el capítulo de requisitos.

Se ha documentado el capítulo de tecnologías utilizadas debido a que ya se tienen claras las tecnologías que se van a utilizar en el proyecto. Se han corregido las interfaces de la aplicación de identificación, registro y descripción de la aplicación.

Para la interfaz de descripción de la aplicación se han corregido los textos que se mostraban, ya que está vista funciona como una *landing page* para atraer a los usuarios. Se han definido textos que describen más claramente las ventajas de la aplicación y se han organizado las secciones de esta interfaz por orden de importancia.

Para la interfaz de registro, se ha corregido la validación del formulario, ya que no pueden existir dos usuarios con el mismo *email* o con el mismo *username*. Para conseguirlo, cada vez que se pulsa una tecla en el campo correspondiente se realiza una comprobación para ver si existe un Usuario con ese *email* o *username* introducido.

Este *sprint* ha tenido una duración de 30 horas.

En la Tabla 8.4 se pueden ver las tareas que se han realizado en este *sprint*.

## 8.2. SEGUIMIENTO DEL PROYECTO

---

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T030	DEV	Desarrollo de la interfaz de identificación.	3h	Completada
T031	DEV	Desarrollo de la interfaz de registro.	3h	Completada
T032	DEV	Desarrollo de la interfaz de descripción de la aplicación.	4h	Completada
T033	DEV	Desarrollo de la interfaz de muro de publicaciones.	3h	Completada
T034	DEV	Desarrollo de la interfaz de perfil de usuario.	3h	Completada
T035	DEV	Desarrollo de la interfaz de editar usuario.	4h	Completada
T036	DEV	Desarrollo de la interfaz de seguidores del usuario.	1h	Completada
T037	DEV	Desarrollo de la interfaz de seguidos del usuario.	1h	Completada
T038	DEV	Desarrollo de la interfaz de recetas favoritas.	1h	Completada
T039	DEV	Desarrollo de la interfaz de detalles de una receta.	4h	Completada
T040	DEV	Desarrollo de la interfaz de detalles de un usuario.	4h	Completada
T041	DEV	Desarrollo de la interfaz de detalles de una publicación.	4h	Completada
T042	DEV	Desarrollo de la interfaz de detalles de un alimento.	4h	Completada
T043	DEV	Desarrollo de la interfaz de crear receta.	6h	Completada
T044	DEV	Desarrollo de la interfaz de crear publicación.	5h	Completada
T045	DEV	Desarrollo de la interfaz de crear alimento.	5h	Completada
T046	DEV	Desarrollo de la interfaz de buscar receta.	5h	Completada
T047	DEV	Desarrollo de la interfaz de buscar usuario.	5h	Completada
T048	DEV	Desarrollo de la interfaz de buscar alimento.	5h	Completada
T049	DEV	Desarrollo de la interfaz de vista de administrador.	10h	Completada

Tabla 8.3: Tareas de *sprint 2*.

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T050	REU	Reunión de seguimiento con los tutores.	1h	Completada
T051	DOC	Corregir errores del resumen.	2h	Completada
T052	DOC	Corregir errores contexto, objetivos, motivación, aplicaciones similares y estructura de la memoria del capítulo introducción.	4h	Completada
T053	DOC	Corregir errores introducción, Requisitos Funcionales, No Funcionales y de Información, Actores Principales, Diagrama de Casos de Uso y Descripción de los Casos de Uso del capítulo de requisitos.	5h	Completada
T054	DOC	Documentar introducción en el capítulo de tecnologías utilizadas.	1h	Completada
T055	DOC	Documentar base de datos en el capítulo de tecnologías utilizadas.	3h	Completada
T056	DOC	Documentar <i>frontend</i> en el capítulo de tecnologías utilizadas.	2h	Completada
T057	DOC	Documentar <i>backend</i> en el capítulo de tecnologías utilizadas.	2h	Completada
T058	DOC	Documentar tecnologías utilizadas en el desarrollo del proyecto en el capítulo de tecnologías utilizadas.	3h	Completada
T059	BUG	Correcciones de la interfaz de identificación.	1.75h	Completada
T060	BUG	Correcciones de la interfaz de registro.	1.75h	Completada
T061	BUG	Correcciones de la interfaz de descripción de la aplicación.	2.75h	Completada
T062	TEST	Pruebas de la interfaz de identificación.	0.25h	Completada
T063	TEST	Pruebas de la interfaz de registro.	0.25h	Completada
T064	TEST	Pruebas de la interfaz de descripción de la aplicación.	0.25h	Completada

Tabla 8.4: Tareas de *sprint* 3.

## **8.2. SEGUIMIENTO DEL PROYECTO**

---

### **8.2.4. Sprint 4: 22/3/2023 - 8/4/2023**

Este *sprint* se ha dedicado a corregir los primeros capítulos de la memoria entre los que se encuentran la planificación y análisis. En el capítulo de análisis se han realizado correcciones sobre el Modelo de Dominio, principalmente añadiendo nuevos atributos a las diferentes clases y creando ciertas relaciones entre clases que no existían y que al desarrollar las interfaces se han descubierto.

Además se ha documentado el capítulo de implementación y pruebas debido a que se han realizado grandes correcciones en la interfaces de la aplicación y por lo tanto se deben ir realizando pruebas de las interfaces completas.

Para la interfaz de editar los datos del usuario, se ha encontrado el problema de que si no se editaba ningún campo el Usuario también podía pulsar de editar y por lo tanto no se actualizaba ningún campo. Para solucionar esto se ha realizado una validación de qué campos del formulario había editado el Usuario de manera que cuando el Usuario editase algún campo se activase el botón de guardar los cambios.

En la *Sprint Review* se ha realizado una demostración del incremento realizado durante el *sprint* y se han propuesto numerosas mejoras sobre la aplicación como dejar la barra de navegación fija, hacer más visibles los botones para crear una publicación o una receta y dejar márgenes mayores en las interfaces para que las vistas se ajusten mejor a los dispositivos móviles.

En este *sprint* ha aparecido el riesgo *R-03 Desarrollo de la interfaz de la aplicación incorrecta* debido a que al realizar pruebas con potenciales usuarios de la aplicación, se ha detectado que no comprendían correctamente las vistas de la aplicación. Se ha aplicado un plan de contingencia que ha consistido en modificar las interfaces para que los usuarios las comprendiesen mejor por lo que se ha simplificado y se ha mejorado la visibilidad de los elementos más importantes.

Este *sprint* ha tenido una duración de 30 horas.

En las Tablas 8.5 y 8.6 se pueden ver las tareas que se han realizado en este *sprint*.

### **8.2.5. Sprint 5: 8/4/2023 - 25/4/2023**

Este *sprint* se han corregido los capítulos de diseño y tecnologías utilizadas. En el capítulo de diseño principalmente se han realizado correcciones sobre el Diseño de la Base de Datos para que el Diseño Lógico de la Base de Datos se ajuste a las correcciones introducidas en Modelo de Dominio. La Arquitectura Lógica del Sistema y los Patrones de Diseño se han corregido en este capítulo para que reflejen el estado actual de la aplicación desarrollada.

Además se han corregido las vistas de la aplicación que más funcionalidad tienen y por tanto requerían más trabajo. Estas vistas son las que muestran los detalles de un elemento como una receta, publicación, alimento o Usuario y las vistas que se usan para crear elementos como los alimentos, recetas o publicaciones.

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T065	REU	Reunión de seguimiento con los tutores.	2h	Completada
T066	DOC	Corregir errores introducción, Scrum, adaptación de Scrum al proyecto, análisis de riesgos, presupuesto y <i>Product Backlog</i> inicial del capítulo de planificación.	3h	Completada
T067	DOC	Corregir errores introducción, Modelo de Dominio, Modelo de Análisis y Realización de Casos de Uso de Análisis del capítulo de análisis.	5h	Completada
T068	DOC	Documentar introducción en el capítulo de implementación y pruebas.	1h	Completada
T069	DOC	Documentar estructura del código de <i>frontend</i> en el capítulo de implementación y pruebas.	2h	Completada
T070	DOC	Documentar estructura del código de <i>backend</i> en el capítulo de implementación y pruebas.	2h	Completada
T071	DOC	Documentar pruebas en el capítulo de implementación y pruebas.	2h	Completada
T072	DOC	Documentar problemas y dificultades encontradas en el capítulo de implementación y pruebas.	2h	Completada
T073	DOC	Desarrollar y documentar Casos de Prueba en el capítulo de implementación y pruebas.	3h	Completada
T074	BUG	Correcciones de la interfaz de muro de publicaciones.	0.75h	Completada
T075	BUG	Correcciones de la interfaz de perfil de usuario.	1.75h	Completada
T076	BUG	Correcciones de la interfaz de editar usuario.	1.75h	Completada
T077	BUG	Correcciones de la interfaz de seguidores del usuario.	0.75h	Completada
T078	BUG	Correcciones de la interfaz de seguidos del usuario.	0.75h	Completada
T079	BUG	Correcciones de la interfaz de recetas favoritas.	0.75h	Completada

Tabla 8.5: Tareas de *sprint* 4.

## 8.2. SEGUIMIENTO DEL PROYECTO

---

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T080	TEST	Pruebas de la interfaz de muro de publicaciones.	0.25h	Completada
T081	TEST	Pruebas de la interfaz de perfil de usuario.	0.25h	Completada
T082	TEST	Pruebas de la interfaz de editar usuario.	0.25h	Completada
T083	TEST	Pruebas de la interfaz de seguidores del usuario.	0.25h	Completada
T084	TEST	Pruebas de la interfaz de seguidos del usuario.	0.25h	Completada
T085	TEST	Pruebas de la interfaz de recetas favoritas.	0.25h	Completada

Tabla 8.6: Tareas de *sprint* 4.

Para la vista de creación de una receta se han corregido numerosos problemas como la validación de que no se añadiese el mismo alimento dos veces a los alimentos necesarios para su preparación. Esto se pudo solucionar recorriendo los alimentos ya seleccionados y comprobando que el alimento nuevo a añadir no se encontraba entre ellos. También se han encontrado problemas en cómo se mostraban los pasos registrados en la receta en pantallas de dispositivos móviles. Esto supuso un gran contratiempo ya que las correcciones realizadas no arreglaban el problema. Se ha optado por cambiar ligeramente la interfaz para que los pasos registrados se mostrasen de manera ordenada también en dispositivos móviles.

En este *sprint* ha aparecido el riesgo *R-12 Aparición de bugs con complicada solución* debido que como se ha comentado se han encontrado problemas a la hora de mostrar los pasos registrados de una receta en pantallas de dispositivos móviles. Se ha aplicado un plan de contingencia que ha consistido en buscar soluciones alternativas para este problema, de manera que se ha cambiado ligeramente la interfaz para que los pasos se mostraran correctamente.

Este *sprint* ha tenido una duración de 30 horas.

En la Tabla 8.7 se pueden ver las tareas que se han realizado en este *sprint*.

### 8.2.6. Sprint 6: 25/4/2023 - 12/5/2023

En este *sprint* se han realizado los últimos capítulos de la documentación que son el de seguimiento y conclusiones. Además se ha realizado el apéndice de manuales y se han realizado las últimas correcciones en las interfaces de la aplicación dedicadas a la búsqueda de elementos como recetas, alimentos o usuarios. Además, se ha corregido la vista de Administrador.

Para la interfaz de vista de Administrador se han corregido varios detalles en las tablas

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T086	REU	Reunión de seguimiento con los tutores.	1h	Completada
T087	DOC	Corregir introducción, Arquitectura Lógica del Sistema, patrones de diseño, Arquitectura Física del Sistema, Diseño de la Base de Datos y Diseño de la Interfaz Gráfica del capítulo de diseño.	4h	Completada
T088	DOC	Corregir errores de la introducción, base de datos, <i>frontend</i> , <i>backend</i> y desarrollo del proyecto del capítulo tecnologías utilizadas.	3h	Completada
T089	BUG	Correcciones de la interfaz de detalles de la receta.	2.75h	Completada
T090	BUG	Correcciones de la interfaz de detalles del usuario.	2.75h	Completada
T091	BUG	Correcciones de la interfaz de detalles de la publicación.	1.75h	Completada
T092	BUG	Correcciones de la interfaz de detalles del alimento.	1.75h	Completada
T093	BUG	Correcciones de la interfaz de crear receta.	3.75h	Completada
T094	BUG	Correcciones de la interfaz de crear publicación.	3.75h	Completada
T095	BUG	Correcciones de la interfaz de crear alimento.	3.75h	Completada
T096	TEST	Pruebas de la interfaz de detalles de la receta.	0.25h	Completada
T097	TEST	Pruebas de la interfaz de detalles del usuario.	0.25h	Completada
T098	TEST	Pruebas de la interfaz de detalles de la publicación.	0.25h	Completada
T099	TEST	Pruebas de la interfaz de detalles del alimento.	0.25h	Completada
T100	TEST	Pruebas de la interfaz de crear receta.	0.25h	Completada
T101	TEST	Pruebas de la interfaz de crear publicación.	0.25h	Completada
T102	TEST	Pruebas de la interfaz de crear alimento.	0.25h	Completada

Tabla 8.7: Tareas de *sprint* 5.

## **8.2. SEGUIMIENTO DEL PROYECTO**

---

que muestran los usuarios, alimentos, recetas y publicaciones almacenadas en la aplicación. Cabe destacar que para la tabla que mostraba las publicaciones almacenadas en la aplicación no se mostraba el nombre del Usuario que la había realizado. Este error se debía a que el *frontend* no sabía interpretar correctamente los datos que le mandaba el *backend* y por lo tanto había campos que no se almacenaban. Esto se ha corregido estableciendo que tanto el *frontend* como el *backend* llamasen cada dato que se comunicaban del mismo modo.

Este *sprint* ha tenido una duración de 30 horas.

En la Tabla 8.8 se pueden ver las tareas que se han realizado en este *sprint*.

### **8.2.7. Sprint 7: 12/5/2023 - 29/5/2023**

En este *sprint* se han revisado los capítulos de seguimiento y conclusiones y el apéndice de manuales. Se han corregido los errores encontrados y se han actualizado estos capítulos según las últimas correcciones en la aplicación.

En este *sprint* ha aparecido el riesgo *R-10 Falta de tiempo para trabajar en el proyecto* debido que en estas fechas se realizaban los exámenes de las asignaturas. Se ha aplicado un plan de contingencia que ha consistido en posponer al siguiente *sprint* las tareas que no se han podido realizar en este *sprint*.

Este *sprint* tenía una duración planificada de 10 horas pero finalmente ha tenido una duración de 6 horas ya que no se han podido realizar todas las tareas.

En la Tabla 8.9 se pueden ver las tareas que se han realizado en este *sprint*.

### **8.2.8. Sprint 8: 29/5/2023 - 15/6/2023**

En este último *sprint* se ha comprobado que todo el trabajo final sea correcto. Para ello se ha revisado la memoria para comprobar que no contenga errores. Además se ha probado la aplicación a fondo para comprobar que toda la funcionalidad cumple con lo requerido, que todas las vistas funcionan correctamente y que se almacenan los datos de la manera deseada.

Este *sprint* tenía una duración planificada de 10 horas pero finalmente ha tenido una duración de 14 horas ya que se han tenido que completar las tareas pendientes del *sprint* anterior.

En la Tabla 8.10 se pueden ver las tareas que se han realizado en este *sprint*.

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T103	DOC	Corregir errores de la introducción, estructura del código de <i>frontend</i> , estructura del código de <i>backend</i> , pruebas, problemas y dificultades encontradas y Casos de Prueba del capítulo implementación y pruebas.	3h	Completada
T104	REU	Reunión de seguimiento con los tutores.	1h	Completada
T105	DOC	Documentar el trabajo realizado en los <i>sprints</i> en el capítulo de seguimiento.	7h	Completada
T106	DOC	Documentar conclusiones en el capítulo de conclusiones.	3h	Completada
T107	DOC	Documentar líneas de trabajo futuras en el capítulo de conclusiones.	1h	Completada
T108	DOC	Documentar el manual de despliegue en el apéndice de manuales.	2h	Completada
T109	DOC	Documentar el manual de Usuario en el apéndice de manuales.	3h	Completada
T110	BUG	Correcciones de la interfaz de buscar receta.	1.75h	Completada
T111	BUG	Correcciones de la interfaz de buscar usuario.	1.75h	Completada
T112	BUG	Correcciones de la interfaz de buscar alimento.	1.75h	Completada
T113	BUG	Correcciones de la interfaz de vista de administrador	2.75h	Completada
T114	TEST	Pruebas de la interfaz de buscar receta.	0.25h	Completada
T115	TEST	Pruebas de la interfaz de buscar usuario.	0.25h	Completada
T116	TEST	Pruebas de la interfaz de buscar alimento.	0.25h	Completada
T117	TEST	Pruebas de la interfaz de vista de administrador.	0.25h	Completada

Tabla 8.8: Tareas de *sprint* 6.

## 8.2. SEGUIMIENTO DEL PROYECTO

---

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T118	DOC	Corregir el trabajo realizado en los <i>sprints</i> en el capítulo de seguimiento.	4h	Completada
T119	DOC	Corregir conclusiones y líneas de trabajo futuras en el capítulo de conclusiones.	0h	No Iniciada
T120	DOC	Corregir el manual de Usuario y de despliegue en el apéndice de manuales.	2h	En Progreso

Tabla 8.9: Tareas de *sprint 7*.

Identificador	Tipo	Descripción	Tiempo Empleado	Estado
T119	DOC	Corregir conclusiones y líneas de trabajo futuras en el capítulo de conclusiones.	2h	Completada
T120	DOC	Corregir el manual de Usuario y de despliegue en el apéndice de manuales.	2h	Completada
T121	DOC	Revisión final de la memoria.	7h	Completada
T122	TEST	Prueba final de la aplicación.	3h	Completada

Tabla 8.10: Tareas de *sprint 8*.

# Capítulo 9

## Conclusiones

### 9.1. Introducción

En este capítulo se explican las conclusiones del presente Trabajo de Fin de Grado. También se exponen que líneas de trabajo futuras se pueden realizar para completar la aplicación desarrollada.

### 9.2. Conclusiones

Una vez finalizado el proyecto, es el momento de valorar que resultado se ha obtenido y como ha evolucionado. Desde el punto de vista del alumno, la realización de esta aplicación para el Trabajo de Fin de Grado ha sido un proceso muy gratificante y satisfactorio, ya que se han podido aplicar los conocimientos aprendidos en estos últimos cuatro años en la Universidad. Además, se ha podido obtener una aplicación de gran utilidad que puede servir como muestra de los conocimientos adquiridos en cuanto al desarrollo Web y a la Ingeniería de *Software*. Este proyecto ha servido para aclarar todos los conocimientos adquiridos en la Universidad y aplicarlos a un proyecto completo, lo que también ha supuesto un gran aprendizaje.

Por otro lado, se han cumplido los objetivos de desarrollo y académicos, no exentos de dificultades, pero un desglose de los objetivos cumplidos se describe a continuación.

En cuanto a los objetivos de desarrollo descritos en la sección 1.2.1:

- Se ha desarrollado una aplicación Web que permite a los usuarios compartir sus recetas y publicaciones. Además, cuenta con mucha más funcionalidad y es fácil de utilizar.

## 9.2. CONCLUSIONES

---

- Se ha seguido el proceso de desarrollo de *software* para crear una aplicación Web. Se han realizado todos los pasos en el proceso de desarrollo de *software* lo que ha hecho que el resultado final sea una aplicación robusta que cumple los requisitos del cliente.
- Se ha realizado la documentación del proyecto justificando las decisiones tomadas para así facilitar su mantenimiento y escalabilidad en el futuro.

En cuanto a los objetivos académicos descritos en la sección 1.2.2:

- Se ha desarrollado una aplicación Web con Angular y NodeJS que se encuentran entre las principales tecnologías del mercado. Además, la aplicación desarrollada puede ser de gran utilidad ya que puede fomentar la alimentación saludable mediante las publicaciones de los usuarios en una red social.
- Se han aplicado los conocimientos aprendidos de Ingeniería de *Software* para crear una aplicación Web. Este proyecto ha servido para asentar estos conocimientos poniéndolos en práctica y además ha supuesto un gran aprendizaje.
- Se ha realizado una interfaz fácil de utilizar y con un diseño atractivo. Cumplir este objetivo no ha sido sencillo, ya que se han consultado numerosos usuarios y se han realizado pruebas para ver qué visión tenían los usuarios finales de la interfaz de la aplicación.
- Se han mejorado las habilidades en el desarrollo del *frontend* en Angular, HTML y CSS ya que se han realizado interfaces con alto grado de dificultad debido a la cantidad de funcionalidad que ofrece la aplicación y la capacidad para adaptarse a dispositivos móviles.
- Se ha mejorado en el diseño de aplicaciones Web con Bootstrap ya que se ha conseguido que la aplicación se adapte a cualquier tamaño de pantalla pese a las dificultades encontradas debido a la complejidad de algunas interfaces.
- Se ha creado una base de datos MySQL para almacenar los datos de la aplicación de manera organizada. Para ello, se ha puesto especial énfasis en el Diseño de la Base de Datos y se han realizado varias correcciones sobre las tablas, atributos y relaciones.
- Se han ampliado los conocimientos sobre NodeJS en el desarrollo del *backend* para gestionar las peticiones del *frontend* y gestionar los datos a almacenar y obtener de la base de datos.
- Se ha desplegado la aplicación mediante Railway para que sea accesible por cualquier Usuario y así hacer que pueda obtener popularidad y un gran número de usuarios.
- Se ha aplicado el marco de trabajo ágil Scrum al proyecto. Esto ha ayudado a comprender en profundidad los pilares de esta forma de trabajo poniéndolos en práctica para el desarrollo de una aplicación.

### 9.3. Líneas de trabajo futuras

Se han encontrado varias propuestas que podrían mejorar y completar la aplicación. Entre estas propuestas caben destacar:

- **Proporcionar más recetas y alimentos en la aplicación para que los usuarios tengan más opciones entre las que elegir y encuentren lo que busquen.** Esto se podría realizar obteniendo las recetas y alimentos de una API. Esta API proporcionaría a la aplicación varias recetas y alimentos para que los usuarios los seleccionasen para crear recetas o realizar publicaciones.
- **Incluir varias fotos en una receta o en una publicación.** Actualmente solo se permite incluir una foto al crear una receta o una publicación. Se podría permitir que los usuarios adjuntasen varias fotos para hacer que la publicación o receta fuese más descriptiva.
- **Añadir *hashtags* a las publicaciones y permitir la búsqueda en base a ellos.** Al crear una publicación se podrían añadir *hashtags* para así agrupar las publicaciones similares. De esta manera, los usuarios podrían buscar publicaciones según el *hashtag* que deseen y así encontrar publicaciones similares relacionadas con un determinado tema.
- **Clasificar las recetas y alimentos según su tipo.** Se podrían agrupar las recetas según su tipo para facilitar la búsqueda a los usuarios. Entre estos tipos, se podrían incluir postres, carnes, veganas, rápidas, etc. De esta forma, el Usuario podría encontrar recetas según el tipo de receta que desee buscar. Lo mismo sucede con los alimentos, entre los tipos de alimentos se podrían encontrar verduras, frutas, carnes y pescados. El Usuario podría buscar los alimentos de un tipo determinado y encontrar todos los alimentos que pertenecen a ese determinado tipo.

### *9.3. LÍNEAS DE TRABAJO FUTURAS*

---

# Apéndice A

## Manuales

### A.1. Manual de despliegue e instalación

Para utilizar la aplicación se puede acceder al enlace de la aplicación Web que se encuentra en el Apéndice B.

#### A.1.1. Requisitos para la instalación

En caso de que se quiera desplegar la aplicación en local, se debe tener instalado el siguiente *software*:

- Angular: versión 15.1.4 o superior.
- NodeJS: versión 18.14.0 o superior.
- Git: versión 2.35.1 o superior en caso de que se quiera clonar el repositorio.
- MySQL Workbench: versión 8.0.32 o superior.

#### A.1.2. Instalación

Los pasos a seguir para desplegar el proyecto son los siguientes:

- Descargar el código del proyecto del repositorio de Gitlab que se encuentra en el Apéndice B. Para ello se puede descargar el .zip con el código del proyecto o clonar el repositorio mediante el comando:

```
$ git clone https://gitlab.inf.uva.es/josloza/tfg-josloza
```

- En la carpeta **frontend** que contiene el código de la parte de Angular, se debe ejecutar en un terminal el siguiente comando para instalar las dependencias necesarias para que la parte de Angular funcione:

```
$ npm install
```

- A continuación en la misma carpeta se debe ejecutar el siguiente comando para desplegar la parte de Angular:

```
$ ng serve
```

- En la carpeta **backend** se debe ejecutar en un terminal el siguiente comando para instalar las dependencias necesarias para que la parte de NodeJS funcione:

```
$ npm install
```

- A continuación, en la misma carpeta se debe ejecutar el siguiente comando para desplegar la parte de NodeJS:

```
$ node index.js
```

- Una vez hecho esto, se puede acceder desde un navegador a la url `http://localhost:4200/` que mostrará la aplicación Web en funcionamiento.

## A.2. Manual de Usuario

Se han tratado de realizar interfaces intuitivas y fáciles de utilizar para que el Usuario aprenda rápidamente su manejo. Sin embargo, a continuación se explican los pasos que puede realizar un Usuario al utilizar la aplicación y cómo puede interactuar con las pantallas.

### A.2.1. Registro e inicio de sesión

Al entrar al enlace de la aplicación definido en Apéndice B, se muestra la **pantalla descripción-aplicacion** (Figura A.1) donde se describe la funcionalidad de la aplicación. En esta pantalla si se pulsa sobre el botón *registro*, se dirigirá al Usuario a la **pantalla registro-usuario** (Figura A.2) para registrarse en la aplicación. Una vez registrado, desde la pantalla descripción-aplicacion si se pulsa el botón *identificarse* se le dirigirá a la **pantalla identificación-usuario** (Figura A.3) para identificarse donde se puede identificar en la aplicación con el *username* y contraseña que se ha introducido en el registro.

### A.2.2. Muro publicaciones, datos del Usuario y su edición

Una vez identificado, se mostrará la **pantalla muro-publicaciones** (Figura A.4) que no mostrará ninguna publicación ya que todavía no se sigue a ningún usuario. En la barra de navegación de la parte superior de esta pantalla si se pulsa sobre *mi perfil* se le llevará a la **pantalla perfil-usuario** (Figura A.5) donde se muestran los datos de su perfil. Una vez

ahí, si se pulsa sobre el botón *editar perfil* se accederá a la **pantalla editar-usuario** (Figura A.6) para editar los datos del perfil donde se puede cambiar la foto de perfil o contraseña entre otros datos. Una vez hechos los cambios si se pulsa el botón *guardar* se le dirigirá de vuelta a la pantalla perfil-usuario donde se muestran los datos de su perfil actualizados.

### A.2.3. Buscador de usuarios, detalles del Usuario y seguidos

En la barra de navegación superior, si se pulsa sobre *buscar* aparecerá un menú despegable. Si se pulsa sobre *buscar usuario* se abrirá la **pantalla buscar-usuario** (Figura A.7) para buscar usuarios donde se podrá introducir un *username* para buscar. Si se clica sobre la carta de un Usuario se mostrarán los detalles del Usuario en la **pantalla detalles-usuario** (Figura A.8). Si se pulsa el botón *seguir usuario* se comenzará a seguir al Usuario y en la pantalla perfil-usuario pulsando sobre el número de usuarios seguidos se le dirigirá a la **pantalla seguidos-usuario** (Figura A.9) que muestra los usuarios a los que sigue. Si en cambio en la pantalla perfil-usuario se pulsa sobre el número de seguidores, se le dirigirá a la **pantalla seguidores-usuario** (Figura A.10) donde se mostrarán los usuarios que le siguen.

### A.2.4. Buscador de recetas, detalles de la receta y favoritas

En el menú despegable al pulsar *buscar* si se pulsa sobre *buscar receta* se le abrirá la **pantalla buscar-receta** (Figura A.11). En ella se puede introducir el título de la receta a buscar y al clicar sobre una carta de una receta se mostrará la **pantalla detalles-receta** (Figura A.12) con los detalles de la receta. En esta pantalla se puede pulsar el botón *añadir receta a favoritas* para añadir la receta a favoritas y en la pantalla perfil-usuario pulsando sobre el botón *mis recetas favoritas*, se le dirigirá a la **pantalla favoritas-recetas** (Figura A.13) que muestra las recetas favoritas del Usuario donde se incluirá esta receta.

### A.2.5. Buscador de alimentos y detalles del alimento

En el menú despegable al pulsar *buscar* si se pulsa sobre *buscar alimento*, se abrirá la **pantalla buscar-alimento** (Figura A.14). Aquí se podrá introducir el nombre del alimento que desea buscar y clicando sobre una carta de un alimento, se mostrará la **pantalla detalles-alimento** (Figura A.15) con los detalles del alimento.

### A.2.6. Crear receta y crear publicación

En la barra de navegación superior, si se pulsa sobre *crear* aparecerá un menú despegable. Si se pulsa sobre *crear receta* se dirigirá al Usuario a la **pantalla crear-receta** (Figura A.16). En esta pantalla, si se llenan todos los campos se podrá crear una receta. En cambio si en el menú despegable al pulsar *crear* se pulsa en *crear publicación* se le dirigirá a la **pantalla crear-publicacion** (Figura A.17). De la misma forma, introduciendo los valores requeridos, se podrá crear una publicación.

### A.2.7. Administrador

Si el Usuario al identificarse tiene el rol de Administrador, se le dirigirá a la **pantalla vista-admin** (Figura A.18). En ella, aparecen por defecto todos los usuarios registrados en la aplicación. Si se clica en la pestaña *alimentos*, se mostrarán todos los alimentos almacenados en la aplicación (Figura A.19). Si se clica en la pestaña *recetas*, se mostrarán todas las recetas creadas en la aplicación (Figura A.20). Si se clica en la pestaña *publicaciones*, se mostrarán todas las publicaciones hechas en la aplicación (Figura A.21). Estas pantallas se muestran como una tabla donde las filas son los elementos de cada tipo. Se puede eliminar cualquiera de estos elementos, pulsando sobre el botón *eliminar* en la fila que corresponda al elemento.

### A.2.8. Crear alimento y editar

Por otro lado, desde la pantalla vista-admin, si se pulsa el botón *crear alimento*, se mostrará la **pantalla crear-alimento** (Figura A.22). En ella, rellenando los campos se podrá crear un alimento. Sin embargo, si quiere editar un alimento, clicando sobre la pestaña de alimentos en la pantalla vista-admin, se puede pulsar el botón *editar* sobre el alimento que deseé. A continuación, se abrirá la pantalla crear-alimento pero cargando los datos del alimento a editar (Figura A.23).



Figura A.1: Pantalla descripcion-aplicacion.



Figura A.2: Pantalla registro-usuario.

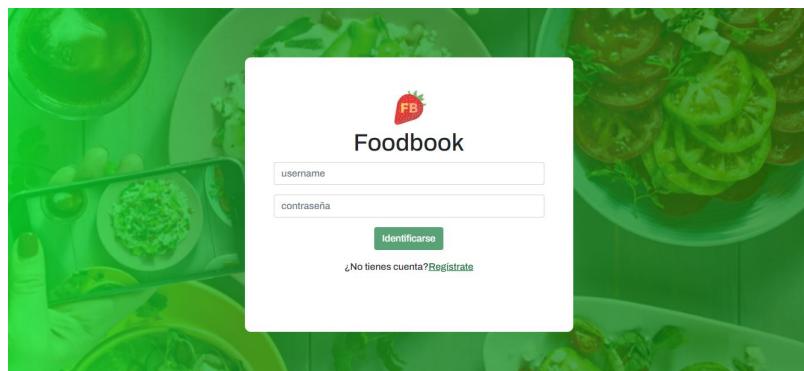


Figura A.3: Pantalla identificacion-usuario.



Figura A.4: Pantalla muro-publicaciones.

## A.2. MANUAL DE USUARIO



Figura A.5: Pantalla perfil-usuario.



Figura A.6: Pantalla editar-usuario.



Figura A.7: Pantalla buscar-usuario.

## APÉNDICE A. MANUALES

---



Figura A.8: Pantalla detalles-usuario.



Figura A.9: Pantalla seguidos-usuario.



Figura A.10: Pantalla seguidores-usuario.



Figura A.11: Pantalla buscar-receta.

Crear Q Buscar Mi Perfil Cerrar Sesión

crema de brocoli

Añadir receta a mis favoritas

chema11

25 minutos

Figura A.12: Pantalla detalles-receta.

RECETAS FAVORITAS

crema de brocoli  
facil  
25 minutos  
chema11

Eliminar

Figura A.13: Pantalla favoritas-recetas.

## APÉNDICE A. MANUALES



Figura A.14: Pantalla buscar-alimento.



Figura A.15: Pantalla detalles-alimento.

This screenshot shows a form for creating a new recipe. The title of the form is "CREAR RECETA". Below the title is a section titled "Información de la receta" with the sub-instruction "En primer lugar, introduce la información básica de la receta para que los demás usuarios sepan en qué consiste". There are four input fields: "Título" (Title), "Resumen" (Summary), "Tiempo de preparación en minutos" (Preparation time in minutes), and "Dificultad" (Difficulty). At the bottom of the form is a green "Guardar Receta" (Save Recipe) button.

Figura A.16: Pantalla crear-receta.

## A.2. MANUAL DE USUARIO

---

Figura A.17: Pantalla crear-publicacion.

Recetas		Alimentos		Usuarios		Publicaciones	
id	username	email		rol	eliminar		
1	chema11	chema112@gmail.com		user	<button>eliminar</button>		
2	javier89	javier8989@gmail.com		user	<button>eliminar</button>		
3	fernandoVall	fernValladolid@gmail.com		user	<button>eliminar</button>		
4	rodrigo21	rod21go@gmail.com		user	<button>eliminar</button>		
5	roberto_fer	ron34fer@gmail.com		user	<button>eliminar</button>		
6	esteban_ja_22	esteban@gmail.com		user	<button>eliminar</button>		
7	ruben893-manz	ruben@gmail.com		user	<button>eliminar</button>		
8	ana_89_gonzal	anagonz@gmail.com		user	<button>eliminar</button>		

Figura A.18: Pantalla vista-admin en la pestaña usuarios.

Recetas		Alimentos		Usuarios		Publicaciones	
id	nombre	calorías	cantidad	medida	acciones		
1	brocoli	34	100	gramos	<button>editar</button> <button>eliminar</button>		
2	queso	350	100	gramos	<button>editar</button> <button>eliminar</button>		
3	calabacín	16	100	gramos	<button>editar</button> <button>eliminar</button>		
4	patata	104	100	gramos	<button>editar</button> <button>eliminar</button>		
5	arroz	129	100	gramos	<button>editar</button> <button>eliminar</button>		
6	zanahorias	41	100	gramos	<button>editar</button> <button>eliminar</button>		
7	manzanas	72	1	unidades	<button>editar</button> <button>eliminar</button>		
8	naranja	62	1	unidades	<button>editar</button> <button>eliminar</button>		

Figura A.19: Pantalla vista-admin en la pestaña alimentos.

## APÉNDICE A. MANUALES

---

PANEL DE ADMINISTRADOR				
Recetas		Alimentos	Usuarios	Publicaciones
id	título	tiempo	creador	eliminar
1	crema de brocoli	25	chemall	<button>eliminar</button>
2	paella	160	chemall	<button>eliminar</button>
3	macarrones con tomate	20	javier89	<button>eliminar</button>
4	prueba	12	chemall	<button>eliminar</button>
5	prueba	12	chemall	<button>eliminar</button>
6	pruebafinal	12	chemall	<button>eliminar</button>
7	adsdsall	11	chemall	<button>eliminar</button>

Figura A.20: Pantalla vista-admin en la pestaña recetas.

PANEL DE ADMINISTRADOR					
Recetas		Alimentos	Usuarios		Publicaciones
id	título	fecha publicacion	creador	alimento/receta	eliminar
1	Domingo paella para todos	2023-02-01T18:00:00.000Z	rodrigo21	paella	<button>eliminar</button>
2	Paella XXL para todo el vecindario	2023-02-01T18:00:00.000Z	fernandoVall	paella	<button>eliminar</button>
3	Me ha faltado echar mas colorante	2023-02-01T18:00:00.000Z	roberto_fer	paella	<button>eliminar</button>
4	Comida sana para los excesos de Navidad	2023-02-01T18:00:00.000Z	ruben893-manz	crema de brocoli	<button>eliminar</button>
5	El brocoli esta infravalorado	2023-02-01T18:00:00.000Z	chemall	crema de brocoli	<button>eliminar</button>
6	Quieres verde? pues toma 3 tazas	2023-02-01T18:00:00.000Z	javier89	crema de brocoli	<button>eliminar</button>
7	Macarrones rapidos para salir del paso	2023-02-01T18:00:00.000Z	rodrigo21	macarrones con tomate	<button>eliminar</button>
8	Que buenos los macarrones!!	2023-02-01T18:00:00.000Z	ana_89_gonzal	macarrones con tomate	<button>eliminar</button>

Figura A.21: Pantalla vista-admin en la pestaña publicaciones.

+

### CREAR ALIMENTO

#### Información general

En primer lugar introduce la información básica del alimento

Nombre

Descripción

Enlace donde comprarlo

Foto

Categoría

Subcategoría

Etiquetas

Imagen

Figura A.22: Pantalla crear-alimento.

The screenshot shows a web-based application for creating a food item. The main title is 'CREAR ALIMENTO' (Create Food) in bold capital letters. Below it is the section 'Información general' (General Information). A sub-instruction says 'En primer lugar introduce la información básica del alimento' (First, enter the basic information about the food). The form fields are as follows:

- Nombre**: A text input field containing the value 'brocoli'.
- Descripción**: A text area containing the following text: 'El brócoli es una verdura que tiene su origen en los países del Mediterráneo Oriental, donde los romanos lo expandieron a otras regiones donde lo cultivaban y consumían, razón por la cual es una verdura muy popular en Italia siendo su nombre también originario de allí, la palabra brócoli o'.
- Enlace donde comprarlo**: A text input field containing the URL 'https://www.carrefour.es/supermercado/brocoli-floretes-carrefour-400-g/R-VC4AECOMM-444613/p?ic\_source=portal-y-corporativo&ic\_medium=t'.
- Foto**: A small thumbnail image of a broccoli floret.

At the bottom right of the form is a green button labeled 'Guardar Alimento' (Save Food).

Figura A.23: Pantalla editar-alimento.

## Apéndice B

# Resumen de enlaces adicionales

Los enlaces útiles de interés en este Trabajo Fin de Grado son:

- Repositorio del código: <https://gitlab.inf.uva.es/josloza/tfg-josloza>.
- Enlace a la aplicación Web: <https://foodbook-production.up.railway.app/>



# Bibliografía

- [1] Simon Kemp. ¡La audiencia publicitaria de TikTok llega a 1,002 millones! (Y otras estadísticas impactantes). <https://blog.hootsuite.com/es/informe-digital-estadisticas-de-redes-sociales/>. Accessed: 2023-2-15.
- [2] Susana Galeano. Cuáles son las redes sociales con más usuarios del mundo (2023). <https://marketing4ecommerce.net/cuales-redes-sociales-con-mas-usuarios-mundo-ranking/>. Accessed: 2023-2-15.
- [3] PuroMarketing. TikTok llega ya a las cifras récord: ya tiene 1.000 millones de usuarios activos en todo el mundo. <https://www.puromarketing.com/149/35721/tiktok-llega-cifras-record-tiene-millones-usuarios-activos-todo-mundo>. Accessed: 2023-2-15.
- [4] financialfood. La tendencia hacia una alimentación saludable se aceleró en 2020 por la pandemia. <https://financialfood.es/la-tendencia-hacia-una-alimentacion-saludable-se-acelero-en-2020>. Accessed: 2023-2-15.
- [5] VelSid. Es probable que las redes sociales influyan en los hábitos alimentarios de los usuarios. <https://gastronomiaycia.republica.com/2020/02/09/es-probable-que-las-redes-sociales-influyan-en-los-habitos-alimentarios>. Accessed: 2023-2-15.
- [6] Diego Santos. Ventajas y desventajas de Instagram: todo lo que puede hacer por tu marketing (y lo que no). <https://blog.hubspot.es/marketing/ventajas-desventajas-instagram>. Accessed: 2023-2-15.
- [7] Tecnocible. Ventajas y desventajas de los Canales en Youtube. <https://www.tecnocible.com/ventajas-desventajas-canales-en-youtube/>. Accessed: 2023-2-15.
- [8] Yelp. Conecta con pequeños comercios fantásticos. <https://www.yelp.es/barcelona>. Accessed: 2023-2-15.
- [9] Recetas de rechupete. Recetas de rechupete. <https://www.recetasderechupete.com/>. Accessed: 2023-2-15.

## BIBLIOGRAFÍA

---

- [10] Directo al paladar. Directo al paladar. <https://www.directoalpaladar.com/>. Accessed: 2023-2-15.
- [11] Ken Schwaber y Jeff Sutherland. The 2020 Scrum Guide. <https://scrumguides.org/scrum-guide.html#scrum-definition>. Accessed: 2023-3-9.
- [12] Manifesto for Agile Software Development. Manifesto for Agile Software Development. <https://agilemanifesto.org/>. Accessed: 2023-3-9.
- [13] Ken Schwaber y Jeff Sutherland. La guía de Scrum. <https://scrumguides.org/docs/scrumguide/v2016/2016-Scrum-Guide-Spanish.pdf>. Accessed: 2023-3-9.
- [14] proyectosagiles.org. Qué es Scrum. <https://proyectosagiles.org/que-es-scrum/>. Accessed: 2023-3-9.
- [15] Bob Hughes y Mike Cottere. Software Project Management. McGraw-Hill Education, 5th edition, 2009.
- [16] Team Asana. Matriz de riesgos: cómo evaluar los riesgos para lograr el éxito del proyecto. <https://asana.com/es/resources/risk-matrix-template>. Accessed: 2023-6-12.
- [17] Indeed. ¿Cuánto se gana como uno Programador/a junior en España? <https://es.indeed.com/career/programador-junior/salaries>. Accessed: 2023-3-9.
- [18] Seguridad Social. Bases y tipos de cotización 2022. <https://www.seg-social.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/36537>. Accessed: 2023-4-18.
- [19] asesorias.com. Horas de trabajo anuales. ¿Cómo calcularlas? <https://asesorias.com/empresas/normativas/laboral/jornada/horas-trabajo-anuales/>. Accessed: 2023-4-18.
- [20] BBVA. ¿Qué es el fondo de maniobra de una empresa? <https://www.bbva.es/finanzas-vistazo/ef/empresas/que-es-el-fondo-de-maniobra-de-una-empresa.html>. Accessed: 2023-4-14.
- [21] Max Rehkopf. Historias de usuario con ejemplos y plantilla. <https://www.atlassian.com/es/agile/project-management/user-stories>. Accessed: 2023-5-21.
- [22] Visuresolutions. Qué son los requisitos funcionales: ejemplos, definición, guía completa. <https://visuresolutions.com/es/blog/functional-requirements/>. Accessed: 2023-4-14.
- [23] Junta de Andalucía. Procedimiento para desarrollar los requisitos de un sistema software que satisfaga las necesidades de negocio. <https://www.juntadeandalucia.es/servicios/madeja/contenido/procedimiento/20>. Accessed: 2023-3-29.
- [24] IBM. Definición de casos de uso. <https://www.ibm.com/docs/es/elm/6.0.3?topic=requirements-defining-use-cases>. Accessed: 2023-3-29.
- [25] EcuRed. Modelo de dominio. [https://www.ecured.cu/Modelo\\_de\\_dominio](https://www.ecured.cu/Modelo_de_dominio). Accessed: 2023-4-14.

## BIBLIOGRAFÍA

---

- [26] IBM. El modelo de análisis. <https://www.ibm.com/docs/es/rsm/7.5.0?topic=model-analysis>. Accessed: 2023-4-18.
- [27] Ionos. Diagramas de secuencia: mostrar interacciones con UML. <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/diagramas-de-secuencia/>. Accessed: 2023-4-14.
- [28] IBM. ¿Qué es la arquitectura de tres niveles? <https://www.ibm.com/es-es/topics/three-tier-architecture>. Accessed: 2023-5-27.
- [29] Martín Durán. Qué es la arquitectura en capas, ventajas y ejemplos. <https://blog.hubspot.es/website/que-es-arquitectura-en-capas>. Accessed: 2023-6-15.
- [30] Leomaris Reyes. Aplicando el patrón de diseño MVVM. <https://medium.com/@reyes.leomaris/aplicando-el-patr%C3%B3n-de-dise%C3%BAo-mvvm-d4156e51bbe5>. Accessed: 2023-5-6.
- [31] Fullstack.pe. Introducción al Data Binding. <https://www.fullstack.pe/blog/angular-data-binding>. Accessed: 2023-5-6.
- [32] Microsoft. Model-View-ViewModel (MVVM). <https://learn.microsoft.com/es-es/dotnet/architecture/maui/mvvm>. Accessed: 2023-6-12.
- [33] Oscar Blancarte. Observer. <https://reactiveprogramming.io/blog/es-patrones-de-diseno/observer>. Accessed: 2023-5-6.
- [34] Yania Crespo González-Carvajal. Apuntes de la asignatura de Diseño de Software.
- [35] Oscar Blancarte. Data Access Object (DAO) Pattern. <https://www.oscarblancarteblog.com/2018/12/10/data-access-object-dao-pattern/>. Accessed: 2023-5-6.
- [36] Oscar Blancarte. Data Transfer Object (DTO). <https://reactiveprogramming.io/blog/es-patrones-geologicos/dto>. Accessed: 2023-5-6.
- [37] Oscar Blancarte. Singleton. <https://reactiveprogramming.io/blog/es-patrones-de-diseno/singleton>. Accessed: 2023-5-6.
- [38] Angular. Singleton services. <https://angular.io/guide/singleton-services>. Accessed: 2023-6-12.
- [39] Creately. La Guía Fácil de los Diagramas de Despliegue UML. <https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/>. Accessed: 2023-5-6.
- [40] MediaCloud. Qué es el diseño de base de datos y cómo planificarlo. <https://blog.mdccloud.es/que-es-el-diseno-de-base-de-datos-y-como-planificarlo/>. Accessed: 2023-4-12.
- [41] Naxer. ¿Qué es el frontend? <https://www.naxer.es/noticias/que-es-el-frontend/>. Accessed: 2023-4-12.
- [42] Webit. ¿Por qué es importante el Frontend? <https://www.webit.cl/por-que-es-importante-el-frontend/>. Accessed: 2023-4-12.

## BIBLIOGRAFÍA

---

- [43] Nerea Boada. ¿Por qué es tan importante el User Experience o Experiencia del Usuario? <https://www.cyberclick.es/numerical-blog/por-que-user-experience-o-experiencia-del-usuario>. Accessed: 2023-4-12.
- [44] Joel Cantero. ¿Cuántos y cuáles colores debe usar tu página web? <https://www.joelcantero.com/tutorial/cuantos-cuales-colores-debe-usar-tu-web/>. Accessed: 2023-4-12.
- [45] Jorge Rodriguez. Diferencias entre Bases de Datos relacionales y no relacionales. <https://codenotch.com/blog/diferencias-entre-bases-de-datos-relacionales-y-no-relacionales/>. Accessed: 2023-4-14.
- [46] Ayudaley. Bases de datos relacional ¿Qué es y sus características? <https://ayudaleyproteccondatos.es/bases-de-datos/relacional/>. Accessed: 2023-4-14.
- [47] Ayudaley. Base de datos no relacional. ¿Qué es? Características y ejemplos. <https://ayudaleyproteccondatos.es/bases-de-datos/no-relacional/>. Accessed: 2023-5-7.
- [48] Keepcoding. ¿Qué es el modelo ACID en bases de datos? <https://keepcoding.io/blog/que-es-acid-bases-datos/>. Accessed: 2023-5-7.
- [49] Angel Robledano. Qué es MySQL: Características y ventajas. <https://openwebinars.net/blog/que-es-mysql/>. Accessed: 2023-5-7.
- [50] Deyimar A. ¿Qué es Angular y cuáles son sus ventajas? <https://www.hostinger.es/tutoriales/que-es-angular>. Accessed: 2023-5-7.
- [51] Manuel José Gonçalves. ¿Qué es Angular y para qué sirve? <https://www.hiberus.com/crecemos-contigo/que-es-angular-y-para-que-sirve/>. Accessed: 2023-5-7.
- [52] Sara López Mora. ¿Qué son las Single-page application (SPA)? El desarrollo elegido por Gmail y LinkedIn? <https://digital55.com/blog/que-son-single-page-application-spa-desarrollo-elegido-por-gmail-linkedin/>. Accessed: 2023-5-7.
- [53] José Luis Chacón. TypeScript: qué es, diferencias con JavaScript y por qué aprenderlo. <https://profile.es/blog/que-es-typescript-vs-javascript/>. Accessed: 2023-5-7.
- [54] MDN Web Docs. HTML: Lenguaje de etiquetas de hipertexto. <https://developer.mozilla.org/es/docs/Web/HTML>. Accessed: 2023-4-22.
- [55] MDN Web Docs. CSS. <https://developer.mozilla.org/es/docs/Web/CSS>. Accessed: 2023-4-22.
- [56] Guest Author. Bootstrap: guía para principiantes de qué es, por qué y cómo usarlo. <https://rockcontent.com/es/blog/bootstrap/>. Accessed: 2023-4-22.
- [57] Laura Cercas Ramos. Cómo añadir gráficos en tu web con Chart.js. <https://www.adictosaltrabajo.com/2022/07/01/como-anadir-graficos-en-tu-web-con-chart-js/>. Accessed: 2023-4-22.

## BIBLIOGRAFÍA

---

- [58] José Carlos Fatjó. Introducción a Angular Material. <https://tech.tribalyte.eu/blog-introduccion-angular-material>. Accessed: 2023-4-22.
- [59] Jesus Lucas. Qué es NodeJS y para qué sirve. <https://openwebinars.net/blog/que-es-nodejs/>. Accessed: 2023-4-22.
- [60] Kinsta. ¿Qué es Express.js? Todo lo que Debes Saber. <https://kinsta.com/es/base-de-conocimiento/que-es-express/>. Accessed: 2023-4-22.
- [61] Atlassian. Qué es Git. <https://www.atlassian.com/es/git/tutorials/what-is-git>. Accessed: 2023-4-22.
- [62] Atlassian. ¿Qué es el control de versiones? <https://www.atlassian.com/es/git/tutorials/what-is-version-control>. Accessed: 2023-4-22.
- [63] Turingears. ¿Qué es GitLab? <https://turingears.com/que-es-gitlab/>. Accessed: 2023-5-15.
- [64] Astah. Astah. <https://astah.net/>. Accessed: 2023-4-22.
- [65] Railway. Railway. <https://railway.app/>. Accessed: 2023-5-15.
- [66] Salesforce. ¿Qué es PaaS? - Descripción de plataforma como servicio. <https://www.salesforce.com/es/learning-centre/tech/paas/>. Accessed: 2023-5-15.
- [67] Getapp. Cloudinary. <https://www.getapp.es/software/101121/cloudinary>. Accessed: 2023-5-15.
- [68] Keepcoding. ¿Qué es JWT (JSON Web Tokens)? <https://keepcoding.io/blog/que-es-jwt/>. Accessed: 2023-5-15.
- [69] Trello. Trello. <https://trello.com>. Accessed: 2023-5-15.
- [70] OpenWebinars. Qué es Visual Studio Code y qué ventajas ofrece. <https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/>. Accessed: 2023-4-12.
- [71] Balsamiq. Balsamiq. <https://balsamiq.com/>. Accessed: 2023-4-12.