

*Proyecto:*

# Cocinando

*Autora:*

Sandra M<sup>a</sup> Acha Nine

*Documento:*

Análisis técnico y funcional del proyecto fin de ciclo DAW (Desenvolvimiento de Aplicacións Web) para IES San Clemente

<b>INTRODUCCIÓN</b>	<b>3</b>
Presentación	3
Antecedentes	3
Objetivos	5
Alcance	6
Estructura de la documentación entregada	6
Información sobre el código del proyecto	7
<b>ANÁLISIS</b>	<b>8</b>
Identificación de implicados en el proyecto y ámbito	8
Definición del producto o requerimientos	8
Requerimientos funcionales y restricciones según tipos de usuarios	8
Requerimientos no funcionales	8
<b>DISEÑO</b>	<b>9</b>
Identificación de componentes	9
Diagrama de flujo(Flowchart)	10
Política de Código	10
Gestión de imágenes y fuentes	11
<b>IMPLEMENTACIÓN</b>	<b>12</b>
Tecnologías y herramientas empleadas	12
Despliegue	12
Nivel Programador	12
Ejecución en Local	12
Puesta en Producción	13
Nivel Usuario	13
Pruebas	13
<b>RESUMEN E INFORMACIÓN LEGAL</b>	<b>14</b>
Aspectos por determinar y posibles ampliaciones	14
Aspectos Realizados	15
Margo legal	16
Conclusiones	17
Bibliografía	17

# 1. INTRODUCCIÓN

## 1.1. Presentación

El proyecto consiste en el desarrollo de una aplicación web responsive para su uso en un restaurante mediante un dispositivo electrónico como puede ser un móvil, ordenador o tablet.

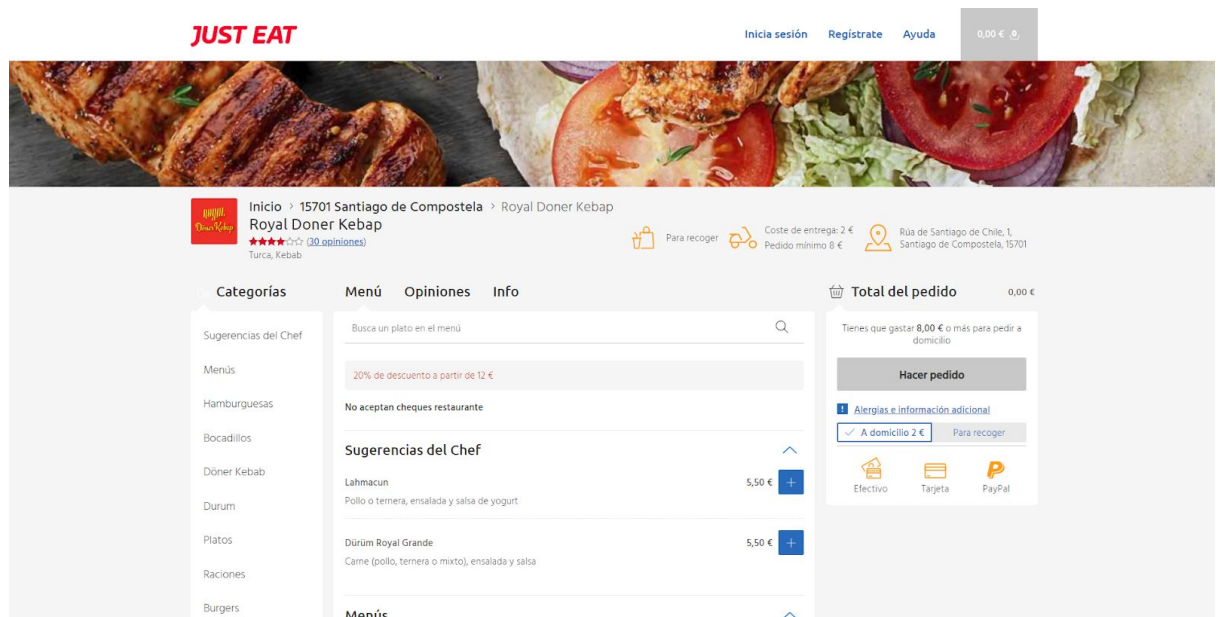
El propósito de la misma consiste en aportar información a los clientes de los productos disponibles, sus características, precios y la selección de los mismos. Suprimiendo así la necesidad de cartas y de comandas físicas. Los empleados del restaurante podrán consultar estas selecciones para su realización en cocina y su servicio en mesa.


El resto del servicio seguiría el funcionamiento normal de este tipo de establecimientos, pudiendo los empleados, una vez los clientes se encuentren disfrutando de su pedido, añadir a petición de los mismos más productos o imprimir el ticket del pedido almacenado para ser pagado.

## 1.2. Antecedentes

En la actualidad existen aplicaciones para la realización de pedidos online donde a partir del acceso a la carta de productos se pueden consultar sus características, precios y otros datos. Añadiendo unidades del producto deseado a un “carrito” donde se debe realizar el pago del mismo para que dicho pedido sea enviado a los empleados del restaurante y así proceder a su elaboración y entrega en el local o a domicilio.


Por ejemplo [Just Eat](#).



 **Total del pedido**

13,50 €

Hacer pedido

 [Alergias e información adicional](#)

☒ A domicilio 2 €

Para recoger

Con 0.50€ más ahorras 20%

-

1 x Menú Kebab

6,00 €

Pollo

+ Agua

[Editar extras](#)

-

1 x Dürüm Royal Grande

5,50 €

Pollo

[Editar extras](#)

Subtotal

11,50 €


Gastos de envío

2,00 €


Total


13,50 €

Dejar comentarios al restaurante



 Efectivo

 Tarjeta

 PayPal

**JUST EAT**

## ¿Cómo quieres pagar?

### Pagar con tarjeta

Número de tarjeta


☒ Guardar mis datos de tarjeta

Caduca

MM

YYYY

CVV

 Los últimos tres números del número en la parte trasera de la tarjeta

Titular de la tarjeta

¿Tienes un cupón de descuento?

Hacer el pedido

PayPal

Pagar en efectivo

### Tu pedido

Menú Kebab	6,00 €
Agua	
Pollo	
Dürüm Royal Grande	5,50 €
Pollo	
Subtotal	11,50 €
Gastos de envío	2,00 €
Total	13,50 €

Royal Doner Kebab, Rúa de Santiago de Chile, 1, 15701

 Entrega solicitada para Lo antes posible

También existen aplicaciones en el caso de servicio en locales como por ejemplo [La Pepita](#). El camarero entrega la carta a los clientes, donde éstos consultan los productos con sus características y precios. Una vez los clientes se hayan decidido, el camarero plasma en su dispositivo los productos indicados y envía una comanda a la cocina para su elaboración.

El proyecto que propongo viene siendo una mezcla de ambos conceptos, donde es el cliente el que escoge los productos de la carta, puede leer su descripción y características, como ocurre en Just Eat, pero estando sentado en el restaurante. Una vez elegidos los productos los añade a su carrito y crea un pedido. Éste pedido crea una comanda para que la cocina los elabore, evitando, como ocurre en La Pepita, que debamos preguntar sobre los productos al camarero, esperar a que pueda tomarnos nota e indicarle lo que queremos.

### 1.3. Objetivos

La aplicación va dirigida al sector hostelero, restaurantes, taperías, bocaterías... establecimientos comerciales, donde se paga por la comida y por la bebida para ser consumidas en el mismo local, pero destinada a que tanto el cliente como el hostelero la utilicen, con diferentes funcionalidades.

La web será accesible desde Internet empleando un navegador web. Dicha web contendrá información sobre el establecimiento como productos, opciones de contacto, ubicación, etc.

Estará formada por varias pantallas, una principal donde el usuario puede acceder a la aplicación mediante varios tipos de login, usuario público, usuario con ticket disponible y usuario privado. Otra pantalla que dispondrá de la información de los productos y otra con el listado de pedidos.

Según el tipo de usuario podremos acceder a diferentes contenidos de la misma, aunque todos dispondrán de acceso a la información del establecimiento.

El usuario público dispondrá de acceso a la carta de productos disponibles, sus características y descripciones e imágenes, pero no a los precios de los mismos. Únicamente se le permitirá consultar la información disponible.

El usuario con ticket podrá tener acceso a los mismos datos que el usuario público con la diferencia de que el primero sí tendrá acceso a los precios de los productos y a la selección de los mismos. Para disponer de éstos privilegios, el usuario deberá introducir el número de ticket entregado por un empleado a su llegada al establecimiento en un formulario antes de acceder. Éste será validado permitiendo al usuario confirmar sus productos seleccionados y enviarlos en una comanda para su posterior elaboración.

El usuario privado tendrá acceso a la aplicación mediante un login formado por usuario y contraseña, podrá acceder a la lista de pedidos que los clientes van solicitando durante el servicio y a la gestión del estado de los mismos.

La aplicación se puede comercializar a cualquier local donde se sirva comida y bebida. Para ello, sólo es necesario modificar los datos de la aplicación, no hay necesidad de modificar el código de la misma si no se desean funcionalidades adicionales o modificaciones en las mismas.

## 1.4. Alcance

Se pretende conseguir un flujo básico desde la parte de login donde accederemos mediante los diferentes roles de la web, desde el listado e información de los diferentes productos, hasta la pantalla de carrito donde se finaliza la solicitud de los productos o servicios seleccionados. Y mediante el rol privado, el acceso al listado de pedidos y gestión de los mismos.

## 1.5. Estructura de la documentación entregada

Este documento está dividido en una serie de capítulos que corresponden, básicamente, a las distintas etapas que conforman el proceso de desarrollo del proyecto. Estas etapas han sido:

- **Introducción:** Se redactó de una manera global una primera visión del proyecto donde señalamos su finalidad y una breve descripción de las funcionalidades y características. La finalidad de esta etapa es aportar información general sobre el proyecto y sus funcionalidades.

- **Análisis:** Se realizó el modelado conceptual de la futura solución mediante el uso de diagramas (diagrama de clases y diagramas de casos de uso). Los modelos ayudan a visualizar como es el sistema, proporcionando plantillas que sirven de guía en la construcción de la aplicación.

En esta etapa se especifica qué debe hacer la aplicación pero no cómo debe hacerlo.

- **Diseño:** Se utilizaron los elementos y modelos obtenidos durante el análisis para transformarlos en mecanismos que puedan ser utilizados en un entorno web con las características y condiciones que establecen este tipo de entornos. Se diseñaron todos los niveles de los que consta la aplicación (nivel de presentación, nivel lógico y nivel de persistencia).

Tanto la etapa del análisis como la del diseño están desprovistas de código.

- **Implementación:** Se utilizaron los elementos obtenidos en el diseño para permitir la elaboración del producto o prototipo funcional, es decir, que puede ser puesto en marcha y sometido a pruebas.

Para ello se consideraron las diversas tecnologías que han intervenido en la elaboración de dicho producto. Todo lo desarrollado en las etapas del análisis y del diseño, se traduce a código.

Esta fase incluye también comprobación del correcto funcionamiento de las funcionalidades desarrolladas mediante una serie de pruebas.

- **Resumen e información legal:** En este apartado se detalla la información legal del proyecto, un breve resumen de los pasos realizados, las posibles ampliaciones a realizar en el proyecto, para su mejora o finalización, las conclusiones y la bibliografía.

## 1.6. Información sobre el código del proyecto

El repositorio se puede encontrar en bitbucket, en la siguiente url <https://bitbucket.org/agrasar/pfc-sandraacha-cocinando>.

La información detallada a continuación también está disponible en el fichero [README.md](#) del proyecto.

Dentro de la carpeta principal del proyecto “**pfc-sandraacha-cocinando**” disponemos de varios ficheros y carpetas de configuraciones y nuestro código que se encuentra dentro de “src”.

En esta, además de ficheros de configuraciones, podemos distinguir el archivo de html principal de la aplicación “index.html” y tres carpetas.

-app: Este directorio va a almacenar la web en sí misma, las vistas y las funcionalidades.

-assets: En ésta añadiremos los recursos extra y que no forman parte de angular, por ejemplo: fuentes, imágenes, ficheros mock de datos, css...

-environments: Aquí podemos ver los ficheros de configuración de los diferentes entornos de ejecución, angular te los crea por defecto, pero en nuestro caso los personalizamos.

Dentro del directorio “app” crearemos una estructura para separar los diferentes ficheros que crearemos, según las funcionalidades que desempeñan y las pantallas a las que pertenecen, y los ficheros principales de la aplicación encargados del flujo entre pantallas, idioma y comunicación entre todos los elementos de la estructura. Angular nos crea el primer componente(app.component) enlazado con las rutas configuradas.

-components: Controlan una pantalla o parte de una y su vista, contienen funcionalidades específicas. Los estructuramos por pantallas y cada pantalla en partes de la misma.

-core: Clases con unos propósitos específicos abstraídas de los componentes para fomentar la reutilización y modularidad, separándolos de las funcionalidades principales de los componentes, storages, contants...

-directives: Comportamientos definidos que podemos utilizar en todos los elementos DOM que queramos de la aplicación.

-model: Modelos de los objetos para mostrar y utilizar los datos.

-pipes: Funcionalidades específicas y muy concretas para utilizarse en toda la aplicación.

-routes: Fichero de rutas configuradas donde podemos especificar y controlar los accesos mediante los diferentes niveles de acceso o rutas parciales donde solo alteramos una parte de la ruta y la pantalla...

-services: Clases para el acceso a los datos de la aplicación, desde el servicio hasta el modelo pasando por las conversiones de los datos.

[Aquí](#) podemos ver las instrucciones de despliegue en local de la aplicación.

## 2. ANÁLISIS

### 2.1. Identificación de implicados en el proyecto y ámbito

El desarrollo de la aplicación va destinado al sector hostelero, más concretamente a restaurantes, taperías, bocaterías...establecimientos que elaboran comidas de algún tipo.

La web ofrece diversos contenidos y funcionalidades que aportan información sobre los productos o servicios de los que dispone el establecimiento y facilita la elección de los mismos por los clientes.

Podemos diferenciar dos tipos de clientes, clientes potenciales(públicos) y clientes actuales(con ticket).

Existe otro apartado de la aplicación web únicamente destinado a los empleados donde se permite la visualización y gestión de los productos o servicios seleccionados por cada cliente.

### 2.2. Definición del producto o requerimientos

#### 2.2.1. Requerimientos funcionales y restricciones según tipos de usuarios

-**Usuario público:** No tiene la necesidad de aportar datos a la aplicación web, accederá a la web públicamente y tendrá acceso a toda la información sobre los platos o productos exceptuando el precio de los mismos, únicamente para la consulta de la misma, en ningún momento dispondrá de la posibilidad de realizar pedidos.

-**Usuario con ticket:** Debe informar el número de ticket para su acceso a la aplicación web y tendrá acceso a toda la información sobre los platos o productos y a la posibilidad de seleccionar elementos y realizar el pedido.

-**Usuario privado:** Debe introducir sus credenciales, usuario y contraseña y tendrá acceso al listado de pedidos y los productos seleccionados en cada uno de estos, no a la información de los productos o platos.

#### 2.2.2. Requerimientos no funcionales

Para poder acceder a la aplicación será necesario disponer de algún dispositivo electrónico que disponga de conexión a Internet mediante un navegador web con la url de la aplicación.

En resoluciones fuera de lo habitual, versiones de navegadores web antiguas o conexión a internet lenta o deficiente es posible que la aplicación no funcione o se visualice todo lo correctamente que debería.



### 3. DISEÑO

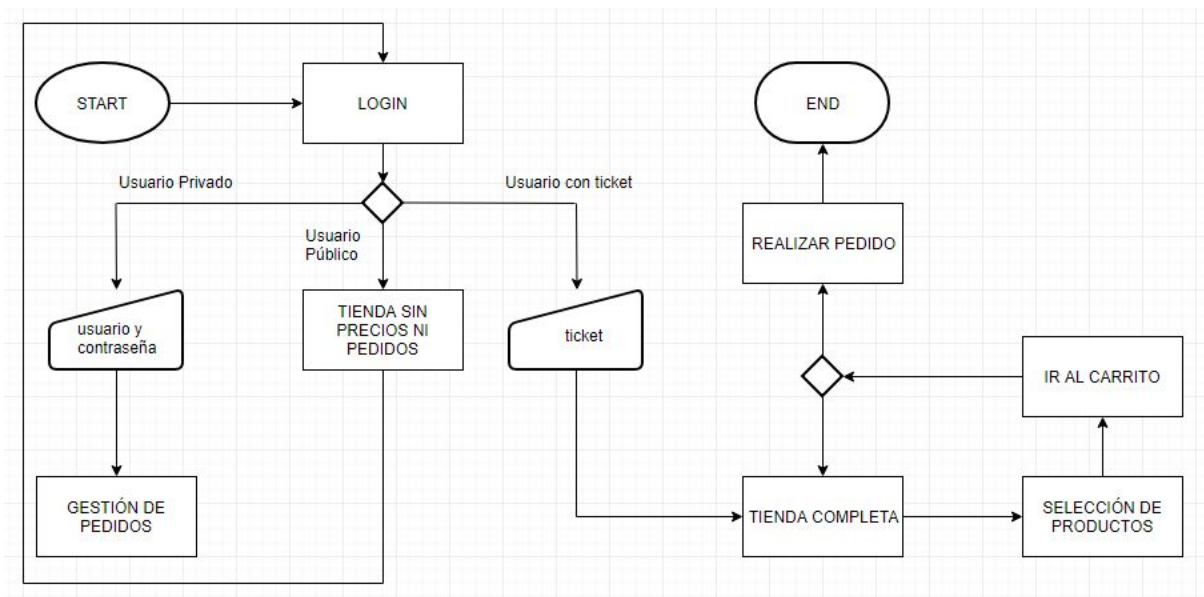
#### 3.1. Identificación de componentes

COMPONENTE	VERSIÓN	LICENCIA	OTROS DATOS
angular	5.0.0	MIT	Framework
ngx-translate/core	9.0.2	MIT	Librería biblioteca Internacionalización
ngx-translate/http-loader	2.0.1	MIT	Librería carga de traducciones Internacionalización
ngx-bootstrap	2.0.3	MIT	Bootstrap para Angular
ngx-logger	1.1.2	MIT	Impresión en Consola
ngx-responsive	6.0.0	MIT	Directivas para Responsive
ngx-cookie	2.0.1	MIT	Servicio de Cookies
font-awesome	4.7.0	OFL-1.1 AND MIT	Paquete de Iconos de Tipo Fuentes
jquery	3.2.1	MIT	Biblioteca de JavaScript
rxjs	5.5.2	Apache 2.0	Extensiones para JavaScript
node	8.9.4	<a href="https://raw.githubusercontent.com/nodejs/node/master/LICENSE">https://raw.githubusercontent.com/nodejs/node/master/LICENSE</a>	JavaScript Runtime
tslint	5.11.0	Apache 2.0	Análisis Estático Extensible
codelyzer	4.5.0	MIT	Conjunto de Reglas Tslint
typescript	3.1.6	Apache 2.0	Lenguaje JavaScript y otros Elementos Combinados
karma	3.0.0	MIT	Test Runner para Javascript

jasmine	2.8.8	MIT	JavaScript Testing Framework
protractor	5.4.0	MIT	end-to-end Test Framework

### 3.2. Diagrama de flujo(Flowchart)

Fichero [“Cocinando Flowchart.html”](https://draw.io) creado con draw.io.



### 3.3. Política de Código

Se utiliza un archivo “tslint.json” para configurar qué reglas de código que se ejecutan y cada una de sus opciones, estas reglas pueden ser preestablecidos y/o personalizadas.

Estas reglas se aplican a los ficheros de typescript, .ts, donde podemos especificar el nivel de gravedad y configuraciones concretas de cada una.

Las reglas disponibles, su descripción y configuraciones se pueden encontrar en <https://palantir.github.io/tslint/rules/>.

Algunas que hemos configurado:

- curly: Controla las llaves de las sentencias if, for...
- deprecation: Avisa cuando se está utilizando un API deprecada.
- eofline: Asegura que el archivo termina con una nueva línea.
- forin: Requiere que un for-in sea controlado con una sentencia if.
- prefer-const: Asegura que se utilicen declaraciones de variables como const en lugar de let y var cuando es posible.

### 3.4. Gestión de imágenes y fuentes

Las imágenes principales de la web se gestionan directamente como elementos en el DOM de tipo "img" y las almacenamos en "/assets/images", en este grupo reducido se encuentran el logo, la imagen de fondo del login y las imágenes de los productos que se gestionan.

El resto de imágenes se van a tratar como fuentes, ya que su control y estilos resulta más sencillo, la mayoría de ellas las obtendremos y gestionaremos con la web fontastic.

Las fuentes que no hemos podido conseguir de fontastic, como los alérgenos, ha sido necesario buscarlas como svgs vectoriales para poder ser subidas y añadidas a nuestro paquete de fuentes de fontastic. En este caso ha sido necesario modificar dichos svgs ya que se encontraban mal formados y al ser añadidos a fontastic, no se visualizaban y/o transformaban correctamente, se han realizado las siguientes modificaciones mediante el programa "Adobe Illustrator":

- No pueden contener bordes
- Se deben formar los grupos de trazos correctamente
- Las secciones del icono que deben ser un único trazado se unifican
- Las partes sin color deben ser transparentes por lo que deben ser bocados sobre las zonas de color.
- El fondo del svg debe ser transparente
- El tamaño de la imagen debe contener una distancia al borde del svg y tanto el espacio(mesa de trabajo) como la imagen interior deben ser siempre del mismo tamaño

## 4. IMPLEMENTACIÓN

El proyecto se realizará implementando un acceso a datos doble, utilizando mocks con datos estáticos para la presentación de las vistas y la posibilidad de enlazar el proyecto con un backend programado con rests o microservicios para el acceso a la base de datos, abstrayendo la parte frontal de los datos a los que acceden y de funcionalidades adicionales irrelevantes para la web.

### 4.1. Tecnologías y herramientas empleadas

- Angular con typescript (Javascript and combines elements of OOP (Object Oriented Programming), FP (Functional Programming), and FRP (Functional Reactive Programming)) y angular-cli
- Node
- Html
- Sass (preprocesador de css) con media queries (Responsive Web Design)
- Bootstrap
- Fontastic
- Git
- Validación de formularios y datos
- Sourcetree
- Visual Studio Code
- Tslint

### 4.2. Despliegue

Para ambos niveles deberemos disponer de un dispositivo electrónico con internet y un navegador web.

#### 4.2.1. Nivel Programador

##### 4.2.1.1. Ejecución en Local

El coste del despliegue en local es muy bajo ya que se emplean lenguajes, programas y frameworks de software libre por lo que no es necesario asumir un coste económico. En este caso el dispositivo debe ser un ordenador.

Para la instalación del proyecto disponemos de una [guía](#) con los pasos a seguir para su despliegue.

#### 4.2.1.2. Puesta en Producción

Para la puesta en producción de la web hemos escogido registrar un nombre de dominio y contratar un alojamiento de pago, dispondremos de mayor control y evitaremos la publicidad en nuestro sitio web.

Para empezar escogemos contratar un “servidor virtual” donde mediante el uso de una máquina virtual, la empresa nos ofrece el control de un ordenador aparentemente no compartido. Así podemos administrar los dominios de forma fácil y económica, además de elegir los programas que se ejecutan en el servidor. En el futuro podríamos cambiar a un “servidor dedicado” donde alquilamos o compramos un ordenador completo, y por tanto tendríamos el control completo y la responsabilidad de administrarlo pero despreocupándonos de su cuidado físico y la conectividad a Internet.

La compra de un dominio .com o .es puede costarnos entre 0,30€ y 10€ y un servidor virtual que podemos conseguir desde 10€, según las características deseadas.

#### 4.2.2. Nivel Usuario

El coste de despliegue para el usuario es mínimo, solo deberá disponer de lo indicado anteriormente y la url de la web.

Únicamente debe acceder a la url desde el navegador del dispositivo.

### 4.3. Pruebas

-**Pruebas de Usabilidad:** Comprobaremos el comportamiento como usuarios dentro de la aplicación web, si resulta fácil de usar, si se pierde en algún paso del flujo. En definitiva, si interactúa de manera fácil y sencilla con ella.

-**Pruebas de Interfaz de usuario:** Evaluamos el diseño de la aplicación web, el tamaño de las ventanas, tipos de letras, colores, ubicación de los controles, el responsive...

-**Pruebas de Rendimiento:** Evaluamos el tiempo de respuesta de nuestra aplicación web, de nuestros componentes y funcionalidades ante diferentes situaciones.

-**Pruebas de Compatibilidad:** Validamos que nuestra aplicación web funciona correctamente para distintos tipos de dispositivos (ordenador, tablet, móvil...) y navegadores web (firefox, chrome, internet explorer...).

Todas estas pruebas las realizaremos como “**Pruebas de Regresión**” de forma que al detectar un error o una mejora en la aplicación, lo corregiremos y volveremos a ejecutar todas las pruebas para ver si la aplicación responde correctamente y no hemos provocado ningún otro error o problema en su resolución.

Para realizar las pruebas únicamente será necesario iniciar la aplicación y acceder a ella desde el navegador, se realizarán en varios, donde mediante el inspeccionador del mismo podemos revisar lo que ocurre en todas las pantallas, las peticiones, los loggers, los puntos de parada en el código, activar el modo device y responsive permitiéndonos simular otros dispositivos y tamaños de pantalla.

Para las pruebas de usabilidad solicitaremos a otra persona que pruebe la aplicación y nos dé su opinión.

## 5. RESUMEN E INFORMACIÓN LEGAL

A continuación haré un resumen del trabajo realizado.

El primer paso que tomé fue establecer las funcionalidades deseadas y los requisitos que debía cumplir el proyecto.

En segundo lugar, realicé los modelos oportunos en UML para que me sirvieran de guía durante la implementación.

A continuación, la etapa a la que más tiempo dediqué, el desarrollo de la aplicación web.

Por último y para comprobar el correcto funcionamiento de todas las funcionalidades y casos de uso implementados, llevé a cabo las pruebas de la aplicación.

### 5.1. Aspectos por determinar y posibles ampliaciones

A continuación se listan una serie de aspectos por determinar, ampliaciones o mejoras que podrían implementarse sobre el proyecto en el futuro, ya que se encuentran fuera de nuestro alcance:

- Sección para opiniones, sugerencias y comentarios sobre la calidad del producto y servicio recibido con fines de mejorar.
- Sección de reservas online dentro de unos plazos establecidos y según el aforo y la ocupación del establecimiento y la obtención del número de ticket siempre que la reserva se realice correctamente.
- Estadísticas de los productos más solicitados y presentación de la web en base a esto.
- Gestión de estados del pedido permitiendo así la modificación del mismo por parte del empleado antes de la finalización del servicio y el cierre del pedido cuando se finaliza y realiza el pago.
- Realización de un backend para poder enlazar la web y poder utilizar datos no estáticos.
- Hacer la web y los datos almacenados en ella resistentes a refrescos de pantalla, flujos inadecuados y controlar los accesos no permitidos para los diferentes roles de la web a las diferentes pantallas disponibles.
- Implementación de una sección de subcategorías.

## 5.2. Aspectos Realizados

Construimos una arquitectura web básica de Angular y le añadimos y configuramos algunos componentes interesantes para la realización de funcionalidades más específicas, enlazamos toda la estructura con los ficheros principales de Angular de forma que cada tipo de ficheros se encuentren independientes y enlazados al padre y no entre sí.

Creamos unos ficheros de traducciones para disponer de la aplicación en Inglés(EN) y Español(ES).

Añadimos unas fuentes, iconos y funcionalidades para utilizarse toda la aplicación.

Creamos los diferentes componentes ordenados por directorios, los componentes reutilizables o independientes se colocan en el nivel superior del directorio “components” junto con las pantallas independientes como los componentes “language”, “user-logout” o “error-page” dentro de cada pantalla podemos dividir en varios componentes-secciones la misma, esto implica que una pantalla puede formarse con diferentes secciones y según el flujo de funcionamiento de las interacciones del usuario con los diferentes elementos de la web nos permite sustituir únicamente la sección cambiante o los datos de la sección cambiante, favoreciendo de esta forma la simplicidad y reutilización como por ejemplo los componentes “cocinando” o “legal-info”.

Todos los componentes para su funcionamiento se apoyan en servicios, directivas y escuchas a cambios en ciertos componentes para su reacción y ejecución de funcionalidades. Los servicios se dividen por función o objeto, ya que así el código es más ordenado y en caso de error o realización de modificaciones sencillo de seguir. Estos servicios en algunos casos realizan funcionalidades y en otros acceden a los servicios de acceso a datos donde se encuentran las llamadas a las rests, los datos de estas se almacenan en modelos de objetos que son convertidos a los modelos de las vistas, se separan para que el cambio si un parámetro o valor es modificado o diferente a lo que esperábamos, sea lo más sencillo posible, puesto que así nos limitamos a una modificación en el converter únicamente.

Las rutas de los componentes las definimos a medida que los creamos, especificando el componente principal y la estructura, esto es importante ya que las pantallas que mantienen la estructura y solo cambian el contenido principal como “private-zone” dependen de la estructura del principal “cocinando” por lo que este es el elemento padre y “private-zone” debe configurarse como elemento hijo.

Definimos estilos comunes para funcionalidades como posición de elementos o control de elementos por los diferentes tamaños de pantallas o dispositivos y para elementos del DOM, solo debemos añadir las clases correspondientes, de esta forma solo debemos realizar en el fichero scss del componente las diferencias o estilos que queramos a mayores.

### 5.3. Margo legal

El proyecto se desarrolla e implementa bajo la protección y los términos y condiciones de la licencia [GNU \(Licencia Pública General\)](#).

A los efectos de lo previsto en la RGPD, del 25 de Mayo de 2018, Reglamento General de Protección de Datos, debemos avisar al usuario y solicitar su aceptación de los mismos. Todos los datos de carácter personal solicitados a los usuarios como consecuencia de su relación con nosotros, así como aquellos que se originen o nos comuniquen en el futuro, serán protegidos e incluidos en nuestras bases de datos y tratados por Sandra Acha Nine como responsable de la web.

La finalidad de pedirle sus datos es gestionar los mismos para que pueda hacer uso de la web donde le ofrecemos productos y servicios, con fines económicos, cuya información y precios están a su disposición y pueden ser solicitados, su legitimación se realiza solo a través del consentimiento del propio usuario, sin el consentimiento en las secciones donde se le solicite, no se le permitirá continuar con el proceso correspondiente.

El consentimiento otorgado para el tratamiento de los datos podrá ser revocado por parte del usuario en cualquier momento, cuando legalmente proceda, si bien dicha revocación no tendrá efectos retroactivos. También se le informa al usuario que podrá ejercitar sus derechos de acceso, rectificación, cancelación y oposición de los mismos, dirigiendo una solicitud por escrito a Cocinando con NIF R7596042G, Rúa de San Clemente, s/n, 15705 Santiago de Compostela, A Coruña o a través de la dirección de correo electrónico [a13sandraaan@iessanclemente.net](mailto:a13sandraaan@iessanclemente.net).

Todas las obligaciones de confidencialidad referidas a los datos anteriores y cualquier otra que, aun no estando declarada anteriormente, sea razonable exigible con arreglo a la buena fe o esté prevista en la legislación vigente en cada momento, permanecerán en vigor durante toda la relación y con posterioridad a la finalización de la misma.

Según lo descrito en la AGPD, Agencia Española de Protección de Datos, se le notificará al usuario la Política de Cookies de la web que deberá aceptar, si este continúa navegando en la misma se considerarán aceptadas.

Estas serán propias y de terceros y se utilizarán para mejorar la experiencia del usuario y nuestra web analizando la navegación con fines estadísticos, de mejorar la web e identificar la sesión y los accesos a las diferentes secciones de la web, algunas públicas y otras con accesos restringidos.



## 5.4. Conclusiones

Al inicio de un proyecto se debe realizar un análisis exhaustivo de los paquetes y librerías que se van a utilizar junto con el framework y las versiones compatibles entre ellos además del estado de dichas librerías, evitando seleccionar estas que no se hayan actualizado en un periodo de tiempo. No darse cuenta de que alguna librería presenta alguno de los problemas mencionados anteriormente al inicio del proyecto y encontrarte con dichos problemas a posteriori, implica muchas modificaciones y la complejidad de las mismas es exponencialmente mucho más alta que al inicio, produciéndose un problema de estimación de proyecto que en la mayoría de los casos provoca retrasos en las entregas o incumplimiento de los objetivos marcados.

El proyecto se ha ido realizando por pantallas y secciones siguiendo el flujo deseado de la aplicación, en algún momento se ha programado algún componente que se utilizaría más adelante en el flujo, esto ha supuesto un error ya que al llegar al punto del desarrollo e intentar integrarlo se han necesitado cambios en el mismo, provocando aunque sea mínimamente una ineficiencia y un desgaste de tiempo que de la otra forma no se hubiera producido.

Se debería reservar un tiempo extra al dar por finalizado un proyecto para modificaciones y/o mejoras, ya que siempre las hay y si no lo hemos reservado es muy probable de estas no puedan realizarse o no se asuma el tiempo extra que implicaría.

El análisis detallado de cada pantalla al inicio es importante para establecer los componentes o estilos que crearemos reutilizables y la estructura de los componentes en el proyecto y aunque siempre surgen cambios, problemas o ideas nuevas, esto nos ayuda a codificar con un objetivo claro y en cualquier caso, si surge algo, que los cambios sean mínimos.

Lo más importante que he aprendido durante el desarrollo de este proyecto y mi trayectoria profesional es que a pesar de que las cosas no siempre salen como uno espera, a medida que pasa el tiempo y la experiencia aumenta, también lo hacen tus capacidades para minimizar los errores, buscar y analizar la información a tu alcance para salir del problema por ti mismo y emplear tu tiempo más eficazmente.

## 5.5. Bibliografía

Documentación framework ANGULAR v5: <https://v5.angular.io/docs>

Documentación Bootstrap para Angular:

<https://valor-software.com/ngx-bootstrap/old/2.0.3/#/getting-started>

Documentación Responsive: <https://www.npmjs.com/package/ngx-responsive>

Documentación flex css: <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Documentación css: <https://www.w3schools.com/css/>

Fuentes: <https://github.com/JulietaUla/Montserrat>

Fuentes svg del proyecto: <http://fontastic.me/>

Librerías de idiomas: <https://github.com/ngx-translate/core> y <https://github.com/ngx-translate/http-loader>

Documentación librerías de idiomas: <https://www.npmjs.com/org/ngx-translate>

Servicio para controlar las cookies: <https://www.npmjs.com/package/ngx-cookie>

Documentación y librería de loggers: <https://www.npmjs.com/package/ngx-logger>

Reglas Tslint: <https://palantir.github.io/tslint/rules/>