Portal web para el polideportivo municipal de Sant Joan d'Alacant

Autor: Víctor Ballester Quiles
Tutor: Jordi Ustrell Garrigos
Profesor: Joan Soler Adillon

Grado de Multimedia
Especialidad de Ingeniería Web

Junio de 2020

Créditos/Copyright

Esta obra está sujeta a una licencia de Reconocimiento – NoComercial - SinObraDerivada
3.0 España de CreativeCommons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Portal web para el polideportivo municipal de Sant Joan d'Alacant*
Nombre del autor:	*Víctor Ballester Quiles*
Nombre del colaborador/a docente:	*Jordi Ustrell Garrigos*
Nombre del PRA:	*Joan Soler Adillon*
Fecha de entrega:	*06/2020*
Titulación o programa:	*Grado de Multimedia*
Área del Trabajo Final:	*Ingeniería Web*
Idioma del trabajo:	*Español*

Resumen del Trabajo:

En la actualidad, la localidad de Sant Joan d'Alacant no ofrece a sus habitantes la posibilidad de reservar las instalaciones del polideportivo municipal a través de Internet, por lo cual, se plantea en este proyecto la realización de una aplicación web que permita a los habitantes del municipio tener una alternativa digital para hacer uso de las distintas instalaciones deportivas.

Para ello, hemos utilizado la metodología *mobile-first*, centrándonos en que la aplicación sea totalmente adaptable y pueda ser utilizada desde cualquier dispositivo. Asimismo, hemos utilizado la técnica de maquetado tanto para el diseño de la base de datos como para el aspecto visual de las distintas páginas que componen el producto final.

Hemos utilizado Bootstrap como *framework* HTML y CSS, así como Angular como *framework* JavaScript para el desarrollo del *frontend*. Por su parte, hemos utilizado Laravel como *framework* PHP para el desarrollo del *backend* y MySQL como sistema para la gestión de la base de datos.

Como resultado, obtenemos una página web que permite a sus usuarios gestionar de manera telemática las reservas de las distintas instalaciones deportivas, realizar pagos a través de una vía segura, gestionar sus suscripciones de socio, así como informarse de las distintas noticias relacionadas con el deporte municipal.

Con la realización de este proyecto, ofrecemos a los habitantes de Sant Joan d'Alacant una herramienta completamente funcional y adaptada a los tiempos actuales que facilitará en gran medida el proceso de reserva de las distintas instalaciones deportivas y aumentará la afluencia de gente joven.

Palabras clave:

Ingeniería web, aplicación web, trabajo final de grado, diseño responsive, mobile-first, sant joan d'alacant, deporte, polideportivo.

Abstract:

Nowadays, people living in Sant Joan d'Alacant cannot interact with their municipal sports center telematically, they have to be fisically present there to make any reservation and pay for it.
As an active user of this center, the main motivation to develop this project is to offer a digital and up to date alternative to people in town to make use of the sports facilities.

We have used a *mobile-first* approach to design this project, focusing on adaptability to all kind of devices. We have made several mockups as for the database as for the visual aspect of the different pages of the web app.
We used different frameworks for each part of the development, such as Bootstrap and AngujarJS for the fron-end side and Laravel for the back-end side. We also used MySQL as our database management system.

As a result, we have created a website that permits users to manage their facilities reservations and payments, manage their memberships and to be informed with local sports news.

With this project, we offer people in Sant Joan d'Alacant a fully modern and functional tool, that will change the way they interact with their sports center, making the reserve and payment process much easier and adapted to the current technology and increasing youth influx to their instalations.

Keywords:

Web Engineering, web app, final degree work, responsive design, mobile-first, sant joan d'alacant, sports, sport center.

Cita

"Yo tampoco sé vivir, estoy improvisando" – Javier Ibarra Ramos

Notaciones y Convenciones

A lo largo del documento se han utilizado las siguientes convenciones para los distintos tipos de contenido.

Título	**Tipografía** Arial Negrita **Tamaño** 20 pt **Color** Negro
Subtitulo	**Tipografía** Arial Negrita **Tamaño** 12 pt **Color** Negro
Apartado	**Tipografía** Arial Subrayado Cursiva **Tamaño** 10 pt **Color** Negro
Tabla 1. Pie de foto o tabla	**Tipografía** Arial **Tamaño** 8 pt **Color** Negro
Enlaces	**Tipografía** Arial Subrayado **Tamaño** 10 pt **Color** Azul

Código	**Tipografía** Consolas
	Tamaño 10 pt
	
	`<!-- Comentario -->`
	`<Etiqueta atributo="propiedad"> Texto HTML </Etiqueta>`

Índice

1. Introducción 11
1.1. Introducción 11
1.2. Descripción 12
1.3. Objetivos generales 14
1.3.1. Objetivos principales 14
1.3.2. Objetivos secundarios 14
1.4. Metodología y proceso de trabajo 15
1.5. Planificación 17
1.6. Presupuesto 21
1.7. Estructura del resto del documento 22
2. Análisis de mercado 24
2.1. Público objetivo y perfiles de usuario 24
2.2. Análisis DAFO 28
3. Propuesta 29
3.1. Definición de objetivos 29
3.2. Promoción de la aplicación 29
4. Diseño 31
4.1. Arquitectura general de la aplicación 31
4.2. Diseño gráfico e interfaces 33
4.2.1. Guía de estilos 33
4.2.2. UI/UX 35
4.3. Lenguajes y recursos utilizados 39
4.3.1. Software utilizado 40
4.3.2. APIs y librerías de terceros 43
5. Implementación 45
5.1. Requisitos de instalación 45
5.1.1. Cliente 45
5.1.2. Servidor 45

5.2. Instrucciones de instalación 46
5.2.1. Cliente 46
5.2.2. Servidor 46
6. Prototipos y esquema de navegación 48
6.1. Wireframes 48
6.2. Esquema entidad-relación de la base de datos 62
6.3. Sitemap 64
6.4. Ejemplos de uso de la aplicación 65
7. Conclusiones y líneas de futuro 67
7.1. Conclusiones 67
7.2. Líneas de futuro 68
Bibliografía 70
Anexos 71

Figuras y tablas

Índice de figuras

Figura 1: Pabellón exterior Polideportivo Sant Joan d'Alacant (http://www.santjoandalacant.es/es/deportes/nuevo-pabellon) 11
Figura 2: Diagrama de Gantt del proyecto 19
Figura 3: Diagrama PERT del proyecto 20
Figura 4: Esquema de comunicación SPA+API 32
Figura 5: Paleta de colores elegida y menú principal 35
Figura 6: Vista en dispositivo móvil 36
Figura 7: Tiempo de carga de la página de noticias 37
Figura 8: Elementos visuales informativos 37
Figura 9: Detalle de las distintas opciones textuales (títulos, subtítulos y párrafos) 38
Figura 10: Información visual en errores de formulario 38
Figura 11: Frameworks más populares 2019 (Fuente: Stackoverflow) 41
Figura 12: Ofertas de trabajo según el framework elegido 41
Figura 13: Instalación de Angular CLI y creación de un nuevo proyecto 42
Figura 14: Esquema de funcionamiento del sistema JWT (https://auth0.com/learn/json-web-tokens/) 44
Figura 15: Requisitos del servidor para ejecutar Laravel (https://laravel.com/docs/7.x#installation) 45
Figura 16: Frontend en localhost:4200 46
Figura 17: Ejemplo de configuración de archivo .env 47
Figura 18: Primer esquema entidad-relación 62
Figura 19: Esquema entidad-relación final 63
Figura 20: Sitemap visual del sitio web 64
Figura 21: Proceso de inicio de sesión y registro de usuarios 65
Figura 22: Proceso de reserva de una instalación 65
Figura 23: Proceso de gestión del estado de suscripción del usuario 66
Figura 24: Ejemplo de diseño en acordeón de Bootstrap 69
Figura 25: Función de registro de usuarios 73
Figura 26: Custom validator en Angular para validar el DNI 74
Figura 27: Función de inicio de sesión 75
Figura 28: Función para realizar una reserva de una instalación 76
Figura 29: Controlador para evaluación del tipo de evento generado por Stripe 77
Figura 30: Manejador de evento para un proceso de reserva pagado con éxito 78
Figura 31: Manejador de evento para una suscripción pagada con éxito 78
Figura 32: Función recursiva que genera el número de socio de manera aleatorio 79
Figura 33: Manejador de evento para modificar suscipciones 79
Figura 34: Servicio que controla los procesos de autenticación desde el lado del cliente 80

Índice de tablas

Tabla 1: Planificación inicial del proyecto 18
Tabla 2: Presupuesto de horas de trabajo 21
Tabla 3: Presupuesto de los costes de formación 21
Tabla 4: Análisis DAFO del TFG 28

1. Introducción

1.1. Introducción

Llevo viviendo en la localidad de Sant Joan d'Alacant desde los diez años – en la actualidad tengo treinta y tres–, todos ellos siendo, en mayor o menor medida, usuario activo de las instalaciones del polideportivo municipal.

Siempre me he quejado, ante mis familiares y amigos, del sistema tan desfasado utilizado por el ayuntamiento para realizar las reservas y los pagos de las instalaciones, así como para llevar el control de los socios. Cuando era adolescente, hablamos de hace quince o veinte años, época en la que mayor uso he realizado de las instalaciones municipales, no había ni siquiera la opción de realizar pagos con tarjeta, y para comprobar tu número de socio debían buscar tu nombre de entre los miles de socios en un enorme libro impreso, que imagino habría que reimprimir cada cierto tiempo para mantenerlo actualizado con las altas y bajas correspondientes. Algo tan sencillo como un ordenador y una base de datos hubiese ahorrado mucho trabajo a los funcionarios que allí trabajaban.

En los tiempos que corren, en los cuales todo el mundo lleva un teléfono móvil en el bolsillo con acceso a Internet, me parece que seguir utilizando el mismo sistema que hace veinte años es un considerable atraso. Es por ello por lo que considero que se debe actualizar la forma en la cual los usuarios interactúan con el polideportivo municipal, dotándolos de una nueva herramienta en forma de aplicación web que permita realizar todo el proceso de reserva y pago de las distintas instalaciones disponibles, así como de gestionar las altas y las bajas de los socios.

Un proyecto de estas características permite dar una imagen moderna a la localidad, a la vez que ayuda a incentivar una mayor afluencia de usuarios a las instalaciones deportivas – especialmente jóvenes, acostumbrados a realizar todas las gestiones de manera telemática –, con el consiguiente aumento de recaudación para las arcas municipales.

Figura 1: Pabellón exterior Polideportivo Sant Joan d'Alacant (http://www.santjoandalacant.es/es/deportes/nuevo-pabellon)

1.2. Descripción

Partimos de la base de que el sistema de reservas del polideportivo municipal de Sant Joan d'Alacant permanece anclado en el pasado y necesita una renovación. Por ello, con el desarrollo de este proyecto se pretende mejorar dicho sistema proporcionando un método más acorde con la tecnología disponible en la actualidad.

En la actualidad, el proceso de reservas y pagos de las instalaciones sigue el mismo patrón que hace más de veinte años, es decir, el usuario debe personarse en las dependencias municipales, consultar los horarios disponibles para las distintas instalaciones y realizar el pago allí mismo en efectivo o mediante tarjeta de crédito. Este método, que probablemente sea suficiente para algunos usuarios, no puede ser el único existente en una sociedad hiperconectada como la actual si lo que se pretende es captar el mayor número de usuarios posibles para el polideportivo, especialmente si se quiere fomentar el deporte entre los más jóvenes. Es necesario, por ello, facilitar herramientas actuales con las que la juventud esté totalmente familiarizada.

El proyecto, por lo tanto, se centra en la creación de dichas herramientas, que permitan una nueva forma de realizar el proceso de pago y reserva de las instalaciones, obteniendo como resultado del trabajo realizado una aplicación web que sirva de referencia para todas las gestiones referentes al polideportivo municipal.

La aplicación consta de dos partes totalmente desacopladas entre sí. Por un lado, un cliente que representa la parte visual del servicio y por otro lado un servidor en forma de API, que se encarga de comunicarse con nuestra base de datos y proveer de información a las vistas.

Desde esta aplicación web, un usuario cualquiera puede consultar los horarios de apertura y cierre de las instalaciones, los horarios disponibles para su reserva, realizar las reservas de manera telemática, así como su pago a través de una línea segura. Además, los habitantes de Sant Joan d'Alacant pueden también gestionar el estado de su suscripción de socio del polideportivo, pudiendo renovar o dar de baja su membresía, así como modificar los datos bancarios donde se realizarán los cargos correspondientes.

La aplicación es totalmente adaptable, con un estilo minimalista y cuidado, de forma que sea perfectamente usable en cualquier tipo de dispositivo. Para su desarrollo se ha aplicado un enfoque *mobile-first*, para asegurar que el sistema funcione a la perfección especialmente en dispositivos móviles.

El *target* principal del producto que se plantea en este proyecto es la parte de la población más joven de Sant Joan d'Alacant, personas de entre 14 y 35 años principalmente, aunque la aplicación va

dirigida a todos los potenciales usuarios de las distintas instalaciones deportivas existentes sin importar su edad.

1.3. Objetivos generales

1.3.1. Objetivos principales

Objetivos de la aplicación:

- Crear una aplicación web plenamente funcional que sea adaptable a cualquier dispositivo.
- Implementar una pasarela de pago que permita realizar cobros de manera segura.

Objetivos para el usuario:

- Poder realizar el proceso de reserva y pago de las distintas instalaciones del polideportivo municipal desde la aplicación web de una forma rápida y segura.
- Poder darse de alta o de baja como socio desde la propia aplicación web, así como gestionar la forma de pago de la suscripción.
- Poder consultar las noticias más relevantes acerca del deporte municipal en la misma aplicación web.

Objetivos personales del autor del TF:

- Conseguir aplicar correctamente tanto Angular como Laravel, de modo que la aplicación sea funcional.
- Optar a ofertas de trabajo relacionadas con la programación web gracias al aprendizaje obtenido en la realización del proyecto.
- Desarrollar un proyecto funcional que pueda ser presentado como maqueta a los responsables de la localidad y que sirva como porfolio personal para posibles entrevistas de trabajo.

1.3.2. Objetivos secundarios

- Mejorar la imagen de la localidad, dotándola de un carácter más moderno y actual.
- Conseguir un aumento de al menos el 10% en el uso de las instalaciones pasados seis meses desde el lanzamiento del producto.

1.4. Metodología y proceso de trabajo

La idea de utilizar diversos *frameworks* como Angular[1] y Laravel[2], con los cuales no estamos familiarizados, ha requerido de un proceso de formación continua a lo largo del desarrollo del proyecto, para lo cual hemos utilizado diversas fuentes de información como YouTube[3], cursos online en plataformas como Udemy[4], así como bibliografía disponible en línea.

Este proceso formativo se ha realizado a la vez que se han desarrollado las primeras etapas de trabajo, donde hemos dado forma a los *wireframes* de las interfaces gráficas de las distintas páginas que componen el portal web y hemos planteado el diseño conceptual y lógico de la base de datos. Durante este tiempo, hemos podido adquirir una base de conocimiento sobre la cual hemos podido plantear las etapas posteriores del desarrollo de la aplicación.

El producto se ha desarrollado desde cero, centrándonos en la experiencia de los usuarios de dispositivos móviles (sin descuidar el resto de los dispositivos), ya que esperamos que estos sean los que generen gran parte del tráfico de nuestra aplicación. Lo hemos hecho así ya que distintas páginas web que hemos utilizado para servicios similares (polideportivos de otras localidades o el portal web de la Universidad de Alicante) no ofrecen una experiencia de usuario del todo satisfactoria y no se adaptan correctamente a teléfonos móviles o tabletas, dificultando el uso o impidiéndolo en su totalidad en algunos casos, por lo cual no son buenos ejemplos en los cuales basar nuestro desarrollo.

La etapa de diseño del producto comienza con los prototipos de baja resolución, realizados a mano en papel, para ir definiendo el aspecto visual de la aplicación. Una vez realizados estos, se realizan los prototipos de alta resolución, mediante la utilización de la aplicación diagrams.net[5], que nos permiten obtener el aspecto visual final que lucirá el portal web.

A la hora de diseñar la base de datos, tanto el diseño lógico como el conceptual se ha realizado en papel, para poder efectuar las correcciones oportunas de forma rápida. Una vez claro el diseño, mediante la aplicación MySQL Workbench[6] se han realizado los diagramas finales en versión digital. Una vez hecho esto, para la realización del diseño físico de la base de datos se han utilizado las migraciones que ofrece Laravel para generar todas las tablas que se hayan diseñado.

Una vez alcanzados todos estos hitos (prototipos de baja y alta fidelidad terminados y diseño de base de datos finalizado), pasamos a la fase de implementación, donde procedemos al diseño del *backend*

[1] https://angular.io/
[2] https://laravel.com/
[3] https://www.youtube.com/
[4] https://www.udemy.com/
[5] https://app.diagrams.net/
[6] https://www.mysql.com/products/workbench/

del proyecto, utilizando el framework Laravel, y al diseño del *frontend* (basándonos en los wireframes elaborados en la fase de diseño) mediante Bootstrap[7] y Angular.

[7] https://getbootstrap.com/

1.5. Planificación

Comenzamos el proyecto con la **fase de análisis**, donde realizamos la propuesta de trabajo al consultor. Una vez recibida la aprobación por su parte, el primer paso es definir formalmente el proyecto a realizar, estableciendo su el alcance, el público al que va dirigido, el método de trabajo a seguir, la planificación temporal y un presupuesto aproximado de las horas que dedicaremos a cada una de las fases correspondientes. Esta primera fase finaliza con la entrega de la segunda práctica el día ocho de marzo.

Una vez finalizada la fase de análisis, damos paso a la **fase de diseño**, donde prototipamos las distintas pantallas que componen el portal web y realizamos el diseño lógico y conceptual de la base de datos, de modo que todas las relaciones entre las distintas tablas y entidades queden correctamente descritas. La fase de diseño finaliza el día 5 de abril, con la entrega de la tercera práctica.

A lo largo de la fase de diseño y la fase de análisis, realizamos un trabajo de formación paralelo, mediante cursos y bibliografía en línea, para adquirir los conocimientos necesarios acerca de Angular y Laravel, de manera que podamos aplicarlos en la siguiente fase. Esta fase de formación es continua a lo largo de todo el proyecto.

Tras la fase de diseño, damos paso a la **fase de implementación**, donde se codifican todos los elementos que componen el proyecto. Esta fase finaliza con la elaboración de la primera versión del proyecto y con la entrega de la cuarta práctica el día diez de mayo.

Concluida la fase de implementación y obtenida la primera versión beta de la aplicación, entramos en la **fase de depuración y test**, donde hemos sometido al producto a distintas pruebas de uso y se han realizado las mejoras y arreglos pertinentes.

Una vez concluida la fase de pruebas, se ha obtenido la versión definitiva del producto, dando por concluido el proyecto coincidiendo con la entrega de la última de las prácticas de la asignatura, el día 8 de junio.

Una vez presentado el proyecto, da comienzo la última fase, la de **evaluación**, donde nos sometemos al juicio del tribunal de evaluación, realizamos la defensa del proyecto y respondemos a las distintas cuestiones que se nos presenten acerca del trabajo realizado.

A continuación, podemos ver una tabla, a modo de resumen, con las fechas clave del proyecto, así como la fecha de los hitos más relevantes del mismo.

Tarea		Inicio	Fin	Duración (días)
Fase de análisis		19/02/2020	07/03/2020	18
	Elaboración de propuesta de TFG	19/02/2020	22/02/2020	4
	Entrega PEC 1	23/02/2020	23/02/2020	Hito
	Definición formal del proyecto 24/2/20	24/02/2020	07/03/2020	13
	Entrega PEC 2	08/03/2020	08/03/2020	Hito
Fase de diseño		09/03/2020	04/04/2020	27
	Elaboración de prototipos low-fi	09/03/2020	13/03/2020	5
	Elaboración de prototipos hi-fi	14/03/2020	04/04/2020	22
	Diseño conceptual BBDD	09/03/2020	15/03/2020	7
	Diseño lógico BBDD	16/03/2020	04/04/2020	20
	Fin wireframes	05/04/2020	05/04/2020	Hito
	Finalización del diseño de la BBDD	05/04/2020	05/04/2020	Hito
	Entrega PEC 3	05/04/2020	05/04/2020	Hito
Fase de implementación		06/04/2020	09/05/2020	34
	Codificación *frontend*	06/04/2020	09/05/2020	34
	Codificación de la BBDD	06/04/2020	12/04/2020	7
	Codificación *backend*	13/04/2020	09/05/2020	27
	Prototipo de la app	10/05/2020	10/05/2020	Hito
	Entrega PEC 4	10/05/2020	10/05/2020	Hito
Fase de pruebas		11/05/2020	07/06/2020	28
	Pruebas de usuario	11/05/2020	07/06/2020	28
	Resolución de errores	11/05/2020	07/06/2020	28
	Entrega PEC 5	08/06/2020	08/06/2020	Hito
Defensa del proyecto		15/06/2020	19/06/2020	5

Tabla 1: Planificación inicial del proyecto

En la siguiente imagen, podemos ver de manera esquemática la tabla anterior, en forma de diagrama de Gantt.

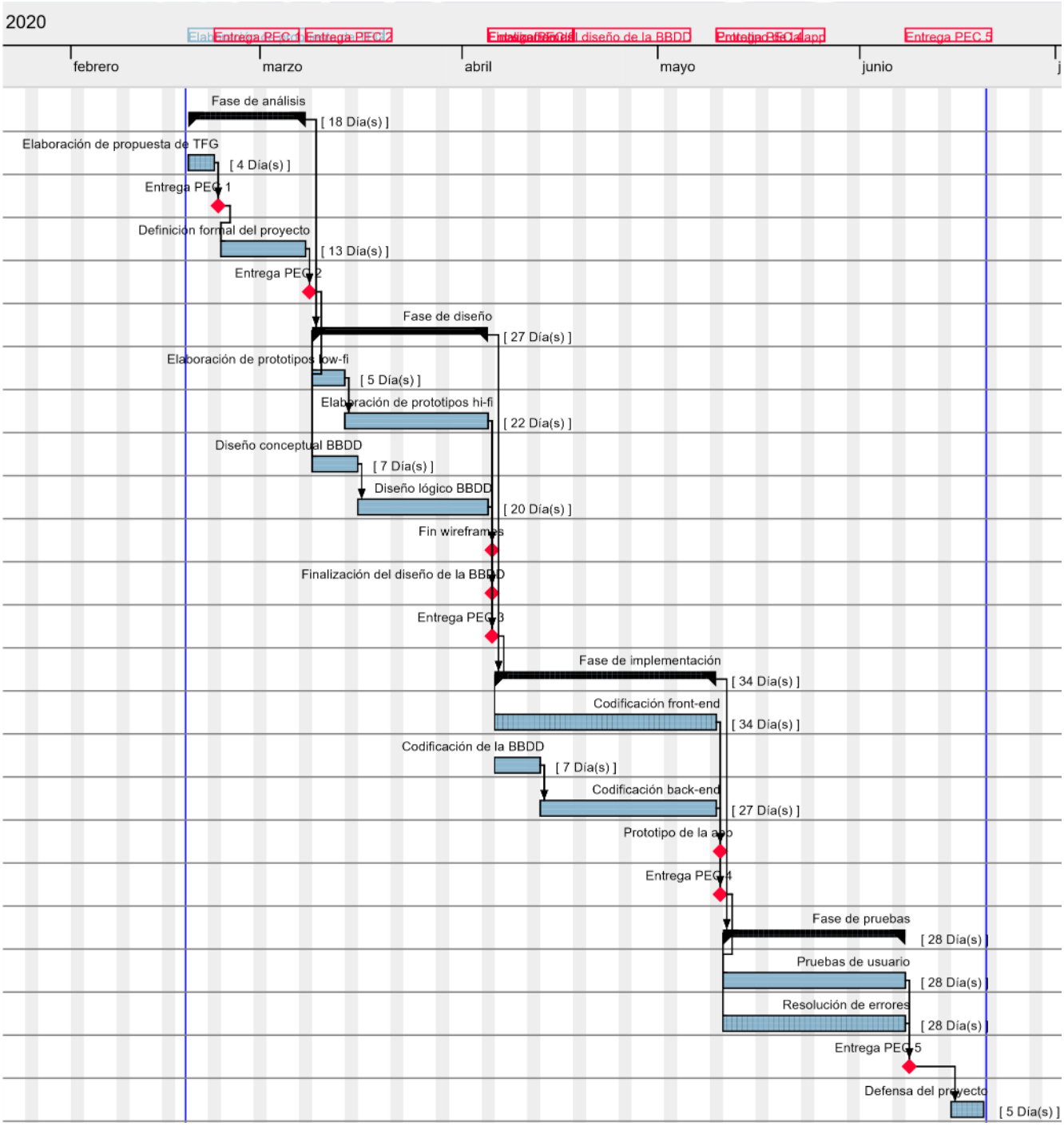


Figura 2: Diagrama de Gantt del proyecto

Por último, podemos ver el diagrama PERT, con los distintos caminos que recorrer durante la realización del proyecto.

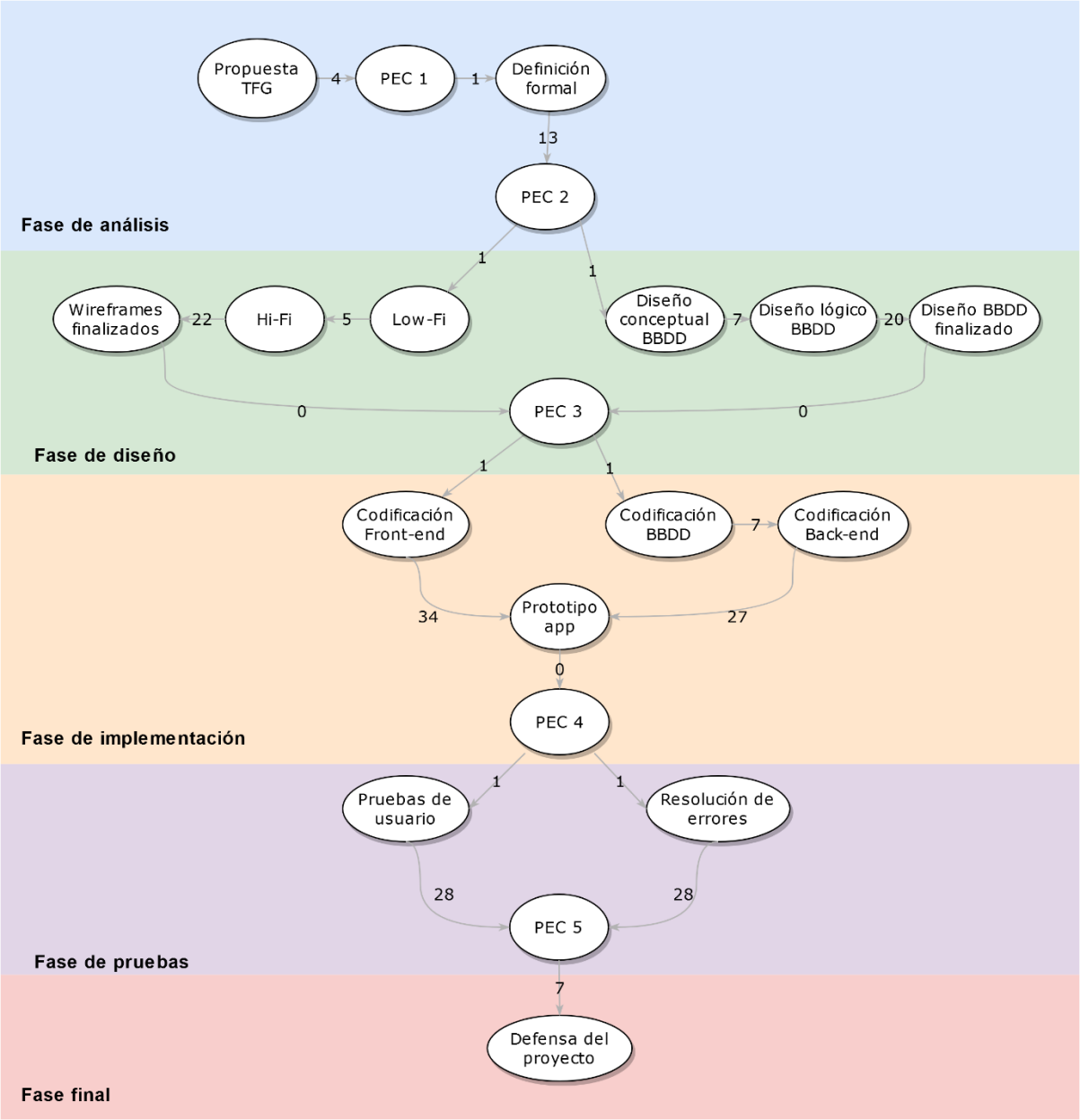


Figura 3: Diagrama PERT del proyecto

1.6. Presupuesto

En este apartado, hemos realizado el presupuesto de las horas necesarias para cada una de las tareas en las que se ha dividido el proyecto. Además, se han incluido el coste de los distintos cursos de formación que hemos necesitado realizar de manera informativa. En un principio no se contempla la venta de la aplicación, esta se ha desarrollado con un propósito didáctico, es por ello por lo que no se han añadido costes monetarios más allá de los cursos realizados.

Fase de análisis		
	Tarea	**Horas**
	Definición formal del proyecto	15
	Subtotal	**15**
Fase de diseño		
	Tarea	**Horas**
	Creación de wireframes low-fi	3
	Creación de wireframes hi-fi	20
	Diseño conceptual de base de datos	6
	Diseño lógico de la base de datos	15
	Subtotal	**44**
Fase de implementación		
	Tarea	**Horas**
	Codificación del *frontend*	30
	Codificación de la base de datos	10
	Codificación del *backend*	60
	Subtotal	**100**
Fase de pruebas		
	Tarea	**Horas**
	Pruebas de usuario	20
	Resolución de errores	35
	Subtotal	**55**
	TOTAL	**214**

Tabla 2: Presupuesto de horas de trabajo

Costes formativos		
	Curso Angular en Udemy	13,99 €
	Curso Laravel en Udemy	10,99 €
	TOTAL	**24,98 €**

Tabla 3: Presupuesto de los costes de formación

1.7. Estructura del resto del documento

Hasta el momento, hemos realizado una introducción a lo que será el contenido del proyecto a desarrollar, ofreciendo un breve resumen de la problemática a abordar, especificando la metodología a seguir en la resolución de dicha problemática, una planificación detallada de las tareas en las que se dividirá el desarrollo del proyecto, así como un presupuesto aproximado de las horas y el dinero necesarios para finalizar el trabajo.

A continuación, comentaremos brevemente el resto de los capítulos de la memoria, a modo introductorio, para poder hacernos una idea del contenido completo de este documento.

Análisis de mercado

Una vez realizada la introducción, el siguiente punto a tratar es el del análisis de mercado, donde se estudia cual es el público objetivo de nuestra aplicación y se realizan diversos perfiles de usuario. Esto nos sirve para poder segmentar a nuestra audiencia según sus distintas necesidades y poder abordar el desarrollo del producto atendiendo a todas y cada una de ellas.

Dentro de este punto, también analizamos qué antecedentes o qué aplicaciones similares a la que queremos desarrollar existen en el mercado, para analizar los pros y contras de cada una de ellas y poder obtener una visión global de la situación actual del mercado.

Por último, mediante un análisis DAFO del proyecto, analizamos sus características tanto internas como externas. Esto nos proporciona información valiosa acerca de cómo afrontar el desarrollo para aprovechar los puntos fuertes y minimizar los puntos débiles del proyecto.

Propuesta

Una vez realizado el análisis de mercado, pasamos al siguiente punto, donde explicamos de forma resumida la propuesta de trabajo de fin de grado. Dentro de esta explicación, definimos de manera detallada los objetivos y las especificaciones de nuestro producto, analizamos el posible impacto económico de la puesta en marcha del mismo y definimos una estrategia de marketing para su promoción.

Diseño

En este capítulo exponemos de manera detallada el trabajo realizado en el desarrollo del producto final. Entre otras cosas, se especifica el tipo de arquitectura de la aplicación, incluyendo diagramas explicativos e imágenes de prototipos y diseños realizados. Se crea una guía de estilo, que defina claramente los tipos de colores y tipografías utilizados en cada elemento. Asimismo, también analizamos los lenguajes de programación utilizados durante el desarrollo, justificando su elección, así como los distintos recursos de software que nos han sido de utilidad.

Demostración

En el siguiente apartado, mostramos los distintos prototipos elaborados durante la realización del trabajo, así como las distintas pruebas realizadas a la aplicación para asegurar su seguridad y su usabilidad. También se crean una serie de diagramas que ejemplifiquen los procesos principales del uso de la aplicación.

Conclusiones

Para finalizar, en el último apartado de la memoria tratamos de exponer las conclusiones más importantes extraídas del desarrollo del proyecto, tanto a nivel personal como a nivel de organización y de consecución de objetivos. También se elabora una lista de posibles mejoras a implementar en futuras versiones del producto.

2. Análisis de mercado

En este capítulo delimitamos el público objetivo al cual va dirigida nuestra aplicación y examinamos las fortalezas y debilidades del proyecto mediante un análisis DAFO.

2.1. Público objetivo y perfiles de usuario

El público objetivo del producto elaborado son los habitantes de Sant Joan d'Alacant que sean usuarios habituales de las instalaciones del polideportivo municipal. Consideramos personas de entre 14 y 55 años, que sean activas, aficionadas al deporte y que, asimismo, sean usuarias de teléfono móvil u ordenador.

También se ha desarrollado esta aplicación teniendo en cuenta un público objetivo secundario, como serían el resto de los habitantes del municipio, sean o no usuarios del polideportivo, así como personas de municipios cercanos.

El marco tecnológico actual, el cual permite realizar multitud de tareas con tan solo un teléfono móvil, hace necesario el desarrollo de una aplicación como la que presentamos, de modo que facilitemos a los usuarios la tarea del proceso de reserva y pago de las distintas instalaciones deportivas que ofrece el municipio.

A continuación, presentamos una serie de perfiles de usuario de ejemplo que nos permiten obtener una muestra de los distintos usuarios potenciales de la aplicación. Gracias a estos perfiles podemos poner cara a distintos tipos de usuario, dependiendo del rango de edad, habilidades tecnológicas o intereses y expectativas comunes, de modo que podamos satisfacer en el desarrollo de la aplicación todos los requerimientos de nuestro público primario.

Andrés Pérez
33 años

Diseñador web

"Me gustaría que la aplicación tuviese un diseño adaptable"

Presentación

Mi nombre es Andrés, tengo 33 años y trabajo como diseñador web en la empresa DiswebSU. Me encargo de realizar los diseños visuales de las distintas páginas web de los proyectos que trabajamos, aunque, por mi formación, también tengo algunas nociones de programación.

Intereses

Soy un entusiasta de las nuevas tecnologías, me gusta todo lo que tiene que ver con el mundo digital y la web en particular, y, además, me encanta hacer deporte. Mis deportes preferidos son el fútbol sala, el tenis y el pádel. También soy aficionado a la fotografía y disfruto de dar largos paseos con mis dos perros siempre que puedo.

Tecnología

Por mi trabajo, estoy más que familiarizado con el uso del ordenador y el teléfono móvil. Estoy acostumbrado al mundo digital, y, siempre que me es posible, realizo búsquedas y compras de productos a través de la red.

Objetivos

Me gustaría que la aplicación ofreciese un sistema de pago seguro y que el diseño fuese totalmente adaptable, para poder utilizarla de manera indistinta tanto desde un ordenador como desde un teléfono móvil.

Paloma Agulló
47 años

Nutricionista y profesora de yoga

"Preferiría un diseño simple y con instrucciones claras"

Presentación

Me llamo Paloma, tengo 47 años y soy nutricionista. Paso consulta en una clínica privada todas las mañanas, trabajo que compagino con las clases de yoga que imparto en mi apartamento tres tardes a la semana para grupos reducidos de entre cuatro y seis personas.

Intereses

Soy una persona activa, me encanta pasear a solas por la playa o la montaña. También me gusta nadar y correr y, siempre que puedo, asisto a las distintas carreras populares que se celebran en mi localidad y alrededores.

Tecnología

Estoy acostumbrada al uso del teléfono móvil, aunque lo utilizo para el *WhatsApp* y poco más. Utilizo el ordenador en mi trabajo como nutricionista, para llevar un control de todos mis pacientes, pero no suelo utilizarlo en mi tiempo libre. He realizado compras a través de Internet, pero prefiero ir en persona a las tiendas para ver y probar todo lo que compro.

Objetivos

Me gustaría que la web fuese muy visual. Que el proceso de consulta y reserva tenga los menos pasos posibles para no perderme por el camino. Si se pudiesen incluir instrucciones claras para cada una de las distintas etapas sería genial.

Ana Vicente

18 años

Estudiante de 2º de Bachillerato

"Me gustaría que tuviese toda la información necesaria para poder hacer todas las gestiones y consultas exclusivamente a través de la aplicación"

Presentación

Me llamo Ana, tengo 18 años y estoy finalizando mis estudios de Bachillerato científico. Me encanta todo lo que tiene que ver con la ciencia y espero obtener una buena nota en las pruebas de selectividad para poder ir a la universidad y estudiar el grado en física.

Intereses

Además de la ciencia soy una apasionada de los videojuegos y paso mis ratos muertos jugando alguna que otra partida con el ordenador. También me gusta hacer deporte, sobre todo en compañía de mis amigos. Disfruto muchos deportes de equipo, pero me gusta especialmente jugar a pádel. Soy una amante de los animales y tengo dos perras a mi cargo.

Tecnología

Soy una adicta a las redes sociales. Uso sobre todo *Instagram*, aunque también *WhatsApp* y *Twitter*. También consumo contenido en redes como *Twitch*, más enfocadas al mundo del videojuego.

Aunque tengo un ordenador en casa, que utilizo básicamente para los trabajos del instituto, todo lo realizo a través del teléfono móvil. Comprar ropa, entradas para el cine o para conciertos, comprar y vender objetos de segunda mano, consultar los horarios del transporte público, etc. Los pagos con el móvil o a través de Internet no son ningún misterio para mí, no utilizo la tarjeta físicamente para nada.

Objetivos

Solo necesito que la web sea fiable y funcional. Que contenga toda la información necesaria para no tener que depender de tener que ir de manera presencial al polideportivo o llamar a algún número de teléfono.

2.2. Análisis DAFO

En este apartado, analizaremos las características internas y externas del proyecto para identificar los aspectos favorables y desfavorables del mismo, de modo que podamos planear las mejores estrategias de desarrollo y mantenimiento futuras.

	Aspectos favorables	**Aspectos desfavorables**
	Fortalezas	**Debilidades**
Análisis interno	Diseño adaptable. Permite un uso fluido de la aplicación tanto en teléfonos móviles, como en tabletas u ordenadores.	Los precios están regulados por una ordenanza municipal. No podemos variarlos para competir con otros centros deportivos.
	Oportunidades	**Amenazas**
Análisis externo	El pago móvil es cada vez más popular. Herramientas como Bizum [8] están teniendo un gran auge entre la población. Sería interesante valorar la posibilidad de facilitar esta herramienta como método de pago.	Los *eSports* están de moda entre los más jóvenes, lo que supone la pérdida de una parte de nuestro público objetivo, que prefiere quedarse en casa jugando con el ordenador o la videoconsola antes que realizar deporte de manera tradicional. Es importante tener esto en cuenta e incorporar una oferta de actividades atractivas para ellos, de forma que se incentive el uso del polideportivo por parte de la juventud.

Tabla 4: Análisis DAFO del TFG

[8] https://bizum.es/

3. Propuesta

Tras el análisis de mercado realizado en el apartado anterior, en este tercer capítulo de la memoria establecemos con claridad los distintos objetivos que se quieren alcanzar con el desarrollo de la aplicación web y sus características principales, así como la estrategia de publicidad a seguir para promocionar el uso del producto en la localidad.

3.1. Definición de objetivos

El **objetivo principal** que debe satisfacer nuestra aplicación es la de permitir a cualquier usuario completar el proceso de reserva y pago de las distintas instalaciones deportivas disponibles en el polideportivo municipal de Sant Joan d'Alacant.

Asimismo, también debe permitir a los usuarios identificarse como socios del polideportivo para beneficiarse de precios especiales, así como gestionar sus suscripciones pudiendo renovarlas o incluso darse de alta o de baja en el servicio.

A continuación, se enumeran las distintas características de la aplicación creada.

- Adaptable a cualquier dispositivo.
- Diseño responsivo.
- *Single-Page Application*.
- Pasarela de pago mediante tarjeta de crédito/débito.
- Posibilidad de pago mediante PayPal y Bizum.
- Registro de usuarios.

3.2. Promoción de la aplicación

Para dar a conocer la aplicación entre la población del municipio e incentivar su uso, hemos elaborado una serie de carteles promocionales que serán expuestos en los espacios publicitarios habilitados por el ayuntamiento en calles y plazas, así como en el propio polideportivo. También hemos diseñado una serie de *banners* publicitarios para incorporar tanto en la versión impresa como en la digital de la revista municipal *La Rambla*[9].

La aplicación será completamente gratuita para cualquier usuario, sea o no vecino del municipio, y para su mantenimiento se contará con el dinero destinado por el ayuntamiento a la concejalía de deportes.

[9] https://costacomunicaciones.es/la-rambla-de-sant-joan

Los precios de las distintas instalaciones deportivas vienen fijados por una ordenanza municipal[10], por lo tanto, no podrán variar a menos que dicha ordenanza sea derogada por una nueva, en cuyo caso se actualizarían sin tardanza para reflejar los nuevos precios acordados.

[10] http://www.santjoandalacant.es/sites/default/files/2017-12/ORDENANZAS%20Instalaciones.pdf

4. Diseño

Una vez que hemos definido el tipo de proyecto que hemos realizado, en este apartado vamos a describir su contenido de una manera más exhaustiva.

En primer lugar, estudiaremos la arquitectura de la aplicación, describiendo las distintas partes que la conforman y analizando la forma en la que se relacionan entre ellas.

En segundo lugar, analizaremos el proyecto desde el punto de vista gráfico, mediante una guía de estilos, mediante la cual mostraremos las distintas decisiones tomadas para la elaboración de la parte visual.

Finalmente, enumeraremos algunas de las opciones disponibles en el mercado para el desarrollo de un sitio web como el elaborado en este proyecto y trataremos de justificar las herramientas utilizadas.

4.1. Arquitectura general de la aplicación

La arquitectura del sitio web está compuesta por dos elementos principales. Por un lado, la arquitectura del *frontend* o cliente y, por otro lado, la arquitectura del *backend* o servidor.

Para el *frontend*, hemos decidido utilizar la arquitectura *Single Page Application* (SPA), utilizando para ello el framework Angular, que es el encargado de generar y gestionar las vistas del sitio web. Por otro lado, para la parte del *backend*, hemos decidido desarrollar una API mediante el framework Laravel, que es el encargado de realizar las operaciones oportunas y de comunicarse con nuestra base de datos para proporcionar la información necesaria para renderizar nuestras vistas en el navegador.

El uso de este tipo de arquitectura nos permite tener un sistema totalmente desacoplado entre el cliente y el servidor, siendo cada uno de ellos el encargado de realizar sus propias tareas, comunicándose entre sí mediante peticiones asíncronas.

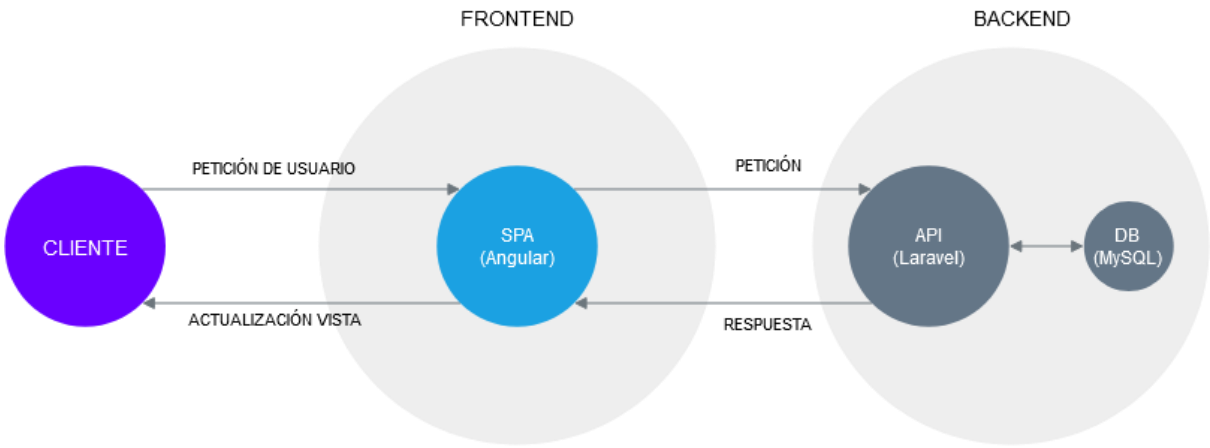


Figura 4: Esquema de comunicación SPA+API

Como vemos en el diagrama anterior, nuestro navegador es el encargado de realizar peticiones a nuestra SPA, en base a acciones del usuario (clic en botones, envío de formularios, etc.). Nuestro frontend recoge estas peticiones y actualiza las vistas en el cliente.

Si fuese necesario, en respuesta a las acciones del usuario, Angular realiza peticiones asíncronas a nuestro servidor que recoge la información necesaria de la base de datos y la devuelve en formato JSON. Este es "un formato ligero de intercambio de datos"[11] que se transmite con rapidez y que permite a Angular actualizar los distintos componentes de la aplicación web de manera fluida sin recargar la página en ningún momento.

Gracias a esta arquitectura, conseguimos una web ligera, que carga los distintos componentes con rapidez aprovechando la capacidad de procesamiento del ordenador del cliente, dejando al servidor la única tarea de comunicarse con la base de datos y servir paquetes de datos al frontend.

[11] https://www.json.org/json-es.html

4.2. Diseño gráfico e interfaces

En este subapartado, mostraremos la guía de estilos generada para el desarrollo del sitio web, donde especificaremos el logotipo utilizado, la paleta de colores elegida, la tipografía, los iconos y los botones, para que sirva de referencia para futuras modificaciones.

Posteriormente, analizaremos las decisiones tomadas en concepto de usabilidad y de experiencia de usuario, que ayuden a que nuestra página web sea más amigable con los usuarios.

4.2.1. Guía de estilos

Logotipo

POLIDEPORTIVO Sant Joan d'Alacant

Paleta de colores

Lorem ipsum dolor	Lorem ipsum dolor	Lorem ipsum dolor
River Bed #343A40 52, 58, 64	**White** #FFFFFF 255, 255, 255	**Amber** #FFC107 255, 193, 7
Color para tipografía, títulos de página, pie de página y botones principales	Color para el fondo de pantalla, para el texto sobre fondos oscuros y para el fondo de botones secundarios	Color de contraste para el *header*

Tipografía

Aa

ABCDEFGHIJKLMNOP
QRSTUVWXYZ
abcdefghijklmnopqrstu
vwxyz
1234567890(,.;:?!$&*)

Montserrat

Tamaño: 400
Estilo: normal

Link o import

```
<link href="https://fonts.googleapis.com/css2?family=Montserrat&display=swap"
rel="stylesheet" type="text/css">

<style>
   @import url("https://fonts.googleapis.com/css2?family=Montserrat&display=swap "
</style>
```

CSS

```
font-family: "Montserrat, sans-serif";
```

Iconos

Botones

Botón principal

Botón secundario

4.2.2. UI/UX

Entendemos por usabilidad como "la facilidad de uso que ofrece una página web. En cambio, en la experiencia de usuario entra en juego […] el nivel de satisfacción que alcanzan (las visitas) al navegar por la web." (Corchado, 2018)

Se puede afirmar, por lo tanto, que la usabilidad es uno de los pilares de la experiencia de usuario. Podemos desarrollar una página web sin errores técnicos, que resulte fácil de utilizar y con una sencilla navegación pero que no proporcione una buena experiencia de uso al navegar por ella, bien sea por que las tareas a realizar sean demasiado complicadas, la carga de las distintas páginas sea muy lenta, la información no esté bien estructurada o el diseño visual sea pobre o confuso. Sin embargo, no podremos tener nunca una buena experiencia de usuario si la página no es usable, es decir, si el usuario es incapaz de completar las tareas y acciones de la web con facilidad.

Atendiendo a las definiciones anteriores, a continuación, se exponen las distintas decisiones adoptadas a lo largo del desarrollo del proyecto en ambos sentidos.

Por un lado, se ha tratado de dotar a la aplicación de un **estilo visual atractivo**, primando siempre la sencillez y evitando sobrecargar la interfaz con demasiados colores, imágenes o iconos que desvíen la atención del usuario. Eso por esto que la paleta de colores elegida cuenta con tan solo tres de ellos, uno principal, uno de relleno y otro de contraste. Además, el menú de usuario se ha simplificado para no contener elementos innecesarios y su diseño es muy liviano, sin elementos que distorsionen la información.

Figura 5: Paleta de colores elegida y menú principal

También se ha diseñado la aplicación teniendo en cuenta el principio *mobile-first*, es decir, planificando las vistas para dispositivos móviles en primer lugar y ajustando después este diseño a resoluciones

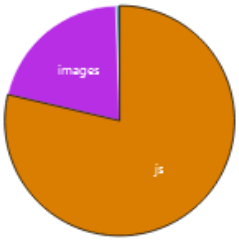
mayores. Esto hace que el **sitio web** sea totalmente **adaptable**, lo que facilita el uso del servicio por parte de cualquier tipo de usuario, bien sean clientes de escritorio, móviles o tabletas. Este hecho asegura un mayor alcance para el producto y el acceso a un público potencial mucho mayor.

Figura 6: Vista en dispositivo móvil

En cuanto a la **velocidad de carga** de las distintas páginas, varía dependiendo de la cantidad de información que se requiere de la base de datos y del número de peticiones realizadas. A pesar de ello la carga es bastante rápida gracias a la propia naturaleza asíncrona de la tecnología utilizada, lo que ayuda a que el usuario se mantenga en la página web y se mantenga una tasa de rebote baja[12].

Además de intentar que la velocidad de carga sea lo más rápida posible, se ha optado por utilizar distintos indicadores visuales que se muestran cuando la página se encuentra cargando información, de manera que el usuario sea consciente de que *algo está pasando* y permanezca a la espera del resultado sin abandonar el sitio web.

[12] https://www.ticbeat.com/lab/5-claves-mejorar-tasa-rebote/



Usando caché

	Tipo	Tamaño	Transferido	Tiempo	Tiempo no bloqueante
9	js	5514,97 KB	5631,72 KB	0,05 s	0,05 s
4	images	1458,35 KB	1458,97 KB	0,01 s	0,01 s
4	html	28,41 KB	11,37 KB	0,30 s	0,30 s
3	xhr	7,67 KB	10,30 KB	0,71 s	0,71 s
1	css	1,96 KB	1,23 KB	0,03 s	0,03 s
1	ws	0 KB	0,13 KB	0,00 s	0,00 s

Respuestas cacheadas: 2
Solicitudes totales: 22
Tamaño: 7011,36 KB
Tamaño transferido: 7113,71 KB
Tiempo: 1,10 segundos
Tiempo no bloqueante: 1,10 segundos

Figura 7: Tiempo de carga de la página de noticias

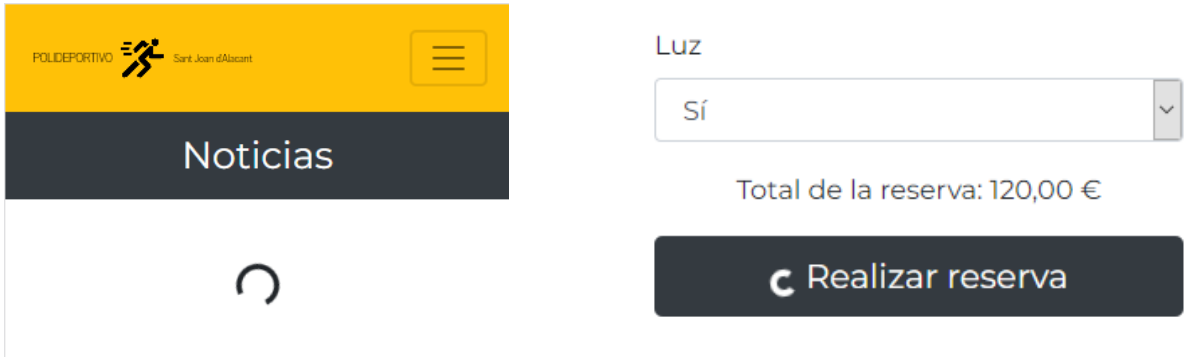


Figura 8: Elementos visuales informativos

En cuanto a la **tipografía**, se ha elegido una fuente sin serifa, ya que mejora la legibilidad en páginas web y especialmente en dispositivos de pequeño tamaño.

Se ha decidido mantener la misma tipografía para todo el sitio, en lugar de utilizar dos tipografías distintas – es habitual utilizar una fuente sin serifa para el cuerpo y una con serifa para los títulos –, y para diferenciar los distintos elementos (títulos, subtítulos y párrafos) se ha optado por jugar con los tamaños de la fuente y los colores, así como otros elementos visuales como el subrayado.

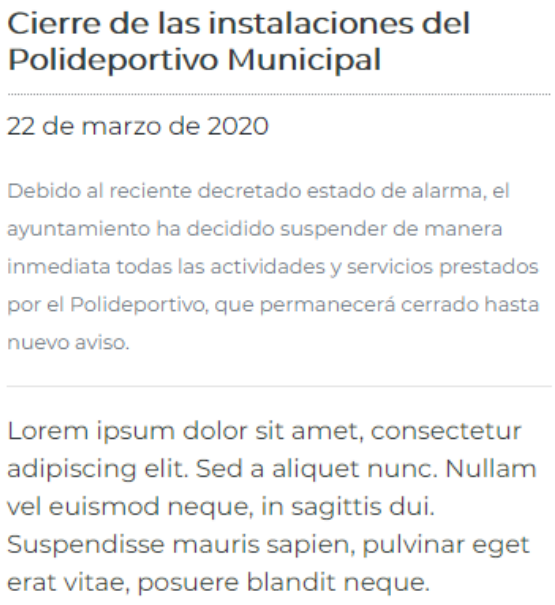


Figura 9: Detalle de las distintas opciones textuales (títulos, subtítulos y párrafos)

Por último, cabe comentar las soluciones adoptadas respecto a la **interacción** los usuarios **con los distintos formularios** de la página web.

Se han incluido una serie de validaciones tanto en el *frontend* como en el *backend*, para evitar que el usuario introduzca información errónea que provoque un mal funcionamiento del sistema. Para avisar al usuario cuando algún dato introducido en el formulario es erróneo, de acuerdo con nuestros requisitos, se muestran por pantalla todo tipo de advertencias y se señalan utilizando colores llamativos los distintos campos que contienen errores. El formulario no se reinicia hasta que el envío de información se realiza con éxito, de modo que el usuario no deba rellenar los mismos campos una y otra vez cuando se encuentre algún fallo.

Figura 10: Información visual en errores de formulario

4.3. Lenguajes y recursos utilizados

En la actualidad existen numerosas opciones para realizar un desarrollo web, dependiendo de si se tienen conocimientos de programación o no y del propósito general de la aplicación que se desee desarrollar.

Por un lado, tenemos gestores de contenido como Wordpress[13] o Joomla[14], de propósito más general, u otros más específicos enfocados hacia el mundo del comercio electrónico como Prestashop[15], Magento[16] o Shopify[17], los cuales ofrecen la posibilidad de crear sitios web en pocos pasos, en base a una interfaz gráfica y basando sus estilos visuales en plantillas predefinidas. Es, sin duda, una opción muy popular y cuya gran ventaja es que permite poner en marcha proyectos sin la necesidad de tener conocimientos acerca de desarrollo web.
Por otro lado, si se tienen nociones de programación, el abanico de opciones se incrementa de manera exponencial, a pesar de que el desarrollo típico siempre lo compone la creación de un cliente o *frontend*, un servidor o *backend* y una base de datos.

Cada uno de estos componentes utiliza sus propias tecnologías y lenguajes de programación. Así, por ejemplo, para el desarrollo del *frontend* se utiliza HTML[18] para diseñar el esqueleto de la página, CSS[19] para aplicar los distintos estilos visuales y JavaScript[20] para dotarlo de interactividad, mientras que para el desarrollo del backend existen numeras posibilidades como Java[21], Python[22] o PHP[23], entre muchas otras. Asimismo, también existen una gran cantidad de motores de base de datos disponibles, como MySQL[24], PostgreSQL[25] o MongoDB[26].

---

13 https://es.wordpress.com/
14 https://www.joomla.org/
15 https://www.prestashop.com/es
16 https://magento.com/
17 https://www.shopify.es/
18 https://developer.mozilla.org/es/docs/Web/HTML
19 https://developer.mozilla.org/es/docs/Web/CSS
20 https://developer.mozilla.org/es/docs/Web/JavaScript
21 https://www.oracle.com/java/
22 https://www.python.org/
23 https://www.php.net/
24 https://www.mysql.com/
25 https://www.postgresql.org/
26 https://www.mongodb.com/es

Ahora bien, teniendo todo lo anterior en cuenta, una opción puede ser la de realizar el desarrollo web desde cero utilizando directamente los lenguajes que deseemos y aplicando cada una de sus reglas y convenciones propias, sin embargo, en la actualidad, esto es algo que se realiza en casos muy puntuales donde se requieren sitios web altamente personalizados y muy específicos.

Hoy en día, lo más habitual es utilizar lo que se conocen como *frameworks*, que no son más que herramientas de software que sirven para agilizar el desarrollo de aplicaciones web. Estos *frameworks* proporcionan entornos de desarrollo robustos y eficientes, que han sido ampliamente probados por profesionales, suelen estar bien documentados y habitualmente tienen detrás de ellos una gran comunidad que sirve de soporte para solucionar las posibles dudas que tengamos.

Una de las grandes ventajas de los *frameworks* es que se encargan de gran parte de las tareas de bajo nivel, facilitando mediante el uso de métodos preconstruidos y librerías la realización de las tareas de alto nivel, lo cual ayuda a crear aplicaciones escalables, modulares y mantenibles, asegurándonos de que estas crecen en un entorno controlado.

4.3.1. Software utilizado

4.3.1.1. Frameworks

Para la realización de este proyecto, se ha optado por la utilización de Angular como *framework* para el *frontend* y Laravel como *framework* para el *backend*, mientras que para la base de datos se ha elegido el motor MySQL.

El principal motivo para utilizar Angular reside en la alta demanda de profesionales existente en el campo del desarrollo *frontend* con experiencia en el uso de dicho *framework*. Según los resultados de una serie de encuestas realizadas por el portal Stackoverflow[27], comprobamos que entre los profesionales del desarrollo web hasta un 32% utiliza Angular como framework en sus proyectos.

Si realizamos una búsqueda de empleo en un portal como es Linkedin, podemos ver que Angular es la tecnología más demandada dentro de los frameworks Javascript, por delante de otras como React o Vue. Lo mismo ocurre si realizamos la búsqueda en otro portal muy conocido como Infojobs. Podemos ver los resultados en la figura 12.

Así pues, aprovechando la realización del proyecto que nos ocupa, era un buen momento para aprender a utilizar este framework, pensando en las opciones laborales que se nos pueden presentar.

[27] https://insights.stackoverflow.com/survey/2019#technology-_-web-frameworks

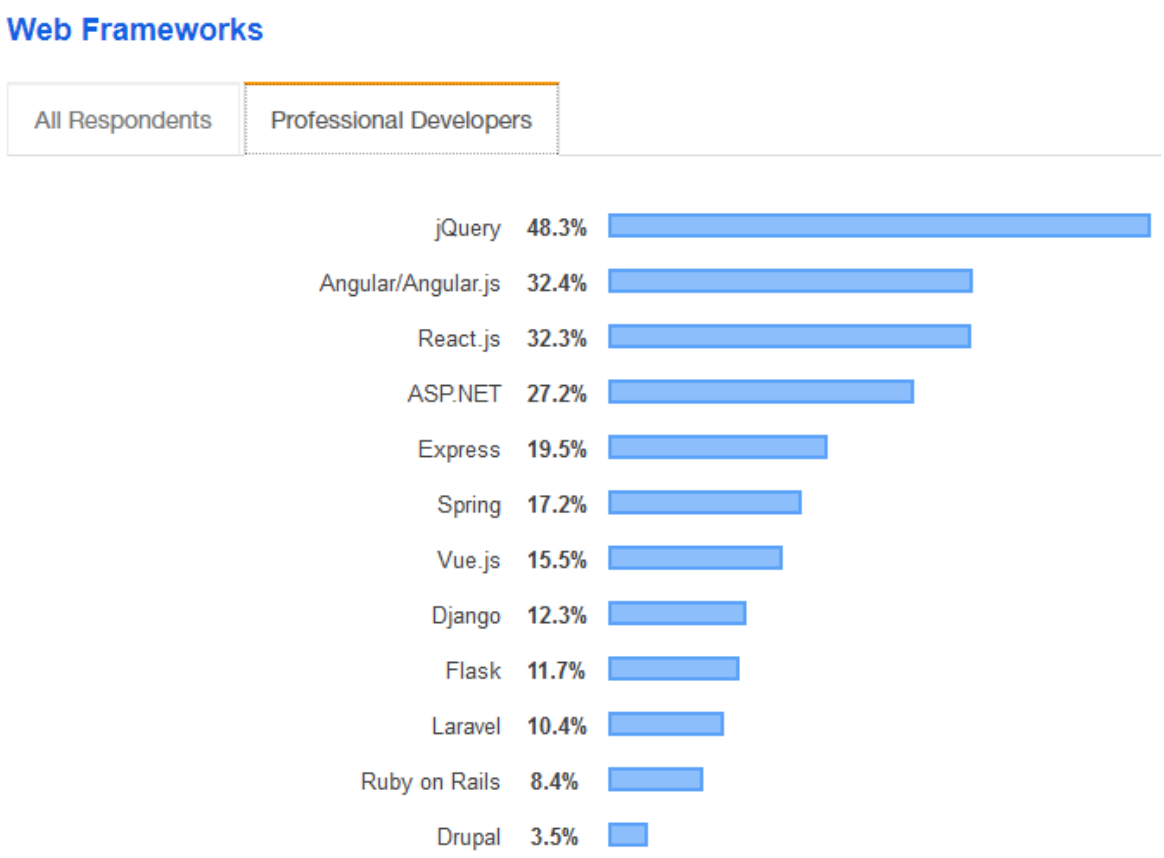


Figura 11: Frameworks más populares 2019 (Fuente: Stackoverflow)

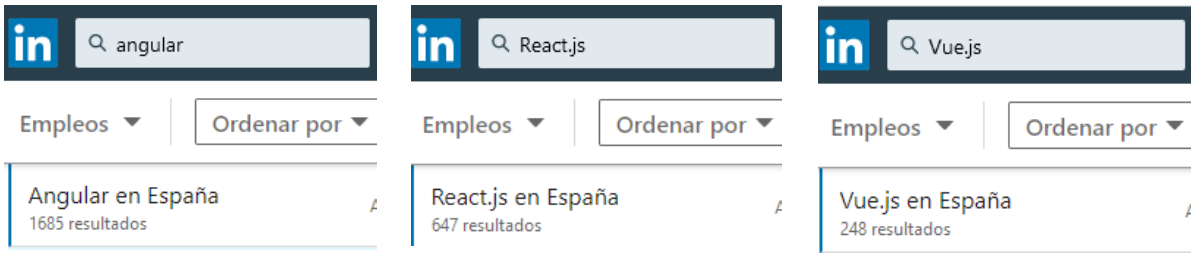


Figura 12: Ofertas de trabajo según el framework elegido

Laravel también aparece en la lista que podemos ver en la figura 11, y, a pesar de situarse detrás de otros frameworks del lado del servidor más populares como ASP.NET, Express, Spring o Django, cuenta con una nada desdeñable cuota de un 10% de usuarios, siendo el único que utiliza el lenguaje PHP de los listados, motivo por el que ha sido el elegido. Al tener algo de experiencia con el desarrollo PHP, estudiando a lo largo de los estudios de grado multimedia, se ha considerado interesante también el aprendizaje de este framework.

En cuanto a la base de datos se ha decidido utilizar MySQL, ya que es la que hemos utilizado en todos los proyectos y ejercicios desarrollados a lo largo de la carrera y porque ya teníamos algo de experiencia con su uso.

4.3.1.2. Herramientas de diseño

Para la realización de los distintos diseños gráficos, como los prototipos de baja fidelidad que podremos ver en detalle en el capítulo 6 de esta memoria, o el logotipo de la página web, se ha optado por el uso de la aplicación web diagrams.net, integrada en Google Drive.

Para el diseño y confección del prototipo de la base de datos, hemos utilizado MySQL Workbench, un software que nos permite crear bases de datos desde cero, añadiendo las tablas y datos necesarios y estableciendo las relaciones entre ellas, generando posteriormente el esquema UML de forma automática.

4.3.1.3. IDE y entorno de desarrollo

Para la edición del código hemos utilizado el entorno de desarrollo Visual Studio Code, de Microsoft.

Para el uso de Angular, hemos seguido las instrucciones que aparecen en la documentación oficial[28]. En primer lugar, es necesario realizar la instalación de Node.js[29]. Una vez instalado Node.js, podemos hacer uso del cliente npm (Node package manager), desde donde podremos realizar la instalación de Angular CLI. Esta interfaz nos permite, entre muchas otras cosas, crear nuevos proyectos Angular en local directamente desde la línea de comandos.

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

```
ng new my-app
```

Figura 13: Instalación de Angular CLI y creación de un nuevo proyecto

Para poder utilizar Laravel y crear nuestro servidor, también hemos seguido las instrucciones de la documentación oficial.

Existen dos opciones para trabajar con Laravel en un entorno local. Una de ellas es instalar directamente la aplicación en el ordenador, lo cual requiere como mínimo la instalación de PHP, un

[28] https://angular.io/guide/setup-local

[29] https://nodejs.org/es/

servidor Apache, un gestor de bases de datos, el gestor de dependencias Composer[30]. Esto puede conseguirse instalando algún paquete tipo XAMPP[31] o WAMP[32]. Sin embargo, existe otra opción, recomendada en la documentación y que es la que se ha utilizado para la creación de nuestra API, como es la instalación de Laravel Homestead[33], un entorno virtualizado que satisface todos los requisitos necesarios para la ejecución de Laravel.

4.3.2. APIs y librerías de terceros

Durante el desarrollo de nuestro proyecto, hemos utilizado algunas APIs y librerías de terceros que nos han ayudado a implementar algunas de las funcionalidades clave de nuestra página web.

Para crear la pasarela de pago, que permite a un usuario pagar por sus reservas y por sus suscripciones de socio, hemos utilizado la API de Stripe. Para su implementación hemos optado por el método *Stripe Checkout*[34], ya que de esa manera redirigimos a nuestros usuarios a una página web gestionada por Stripe que implementa todas las medidas y validaciones necesarias para asegurar una transacción segura. De este modo, evitamos tener que almacenar en nuestra base de datos información sensible de nuestros clientes, como números de tarjetas de crédito, y es Stripe quien realiza el manejo y validación de esta información.

En Angular, hemos utilizado también la librería RxJS[35], que nos permite utilizar el tipo de datos *Observable* para poder reaccionar al flujo de datos generado por las distintas peticiones asíncronas a nuestra base de datos. De este modo, podemos generar respuestas en nuestras vistas en base a los datos que nos vayan llegando desde el servidor de un modo reactivo.

Por último, para implementar el inicio de sesión y proteger algunas de las rutas de nuestra API de modo que solo sean accesibles para usuarios con una sesión iniciada válida, hemos utilizado el paquete jwt-auth[36] de Sean Tymon. Para validar a nuestros usuarios utiliza tokens generados de manera aleatoria en base a una clave secreta generada de manera automática que solo conoce el servidor. El token se envía al cliente cuando inicia sesión en nuestro sistema, y se utiliza en la cabecera de cada petición realizada a nuestra API para garantizar que proviene de un usuario validado.

---

30 https://getcomposer.org/

31 https://www.apachefriends.org/es/index.html

32 http://www.wampserver.es/

33 https://laravel.com/docs/7.x/homestead

34 https://stripe.com/docs/payments/checkout

35 https://angular.io/guide/rx-library

36 https://jwt-auth.readthedocs.io/en/develop/laravel-installation/

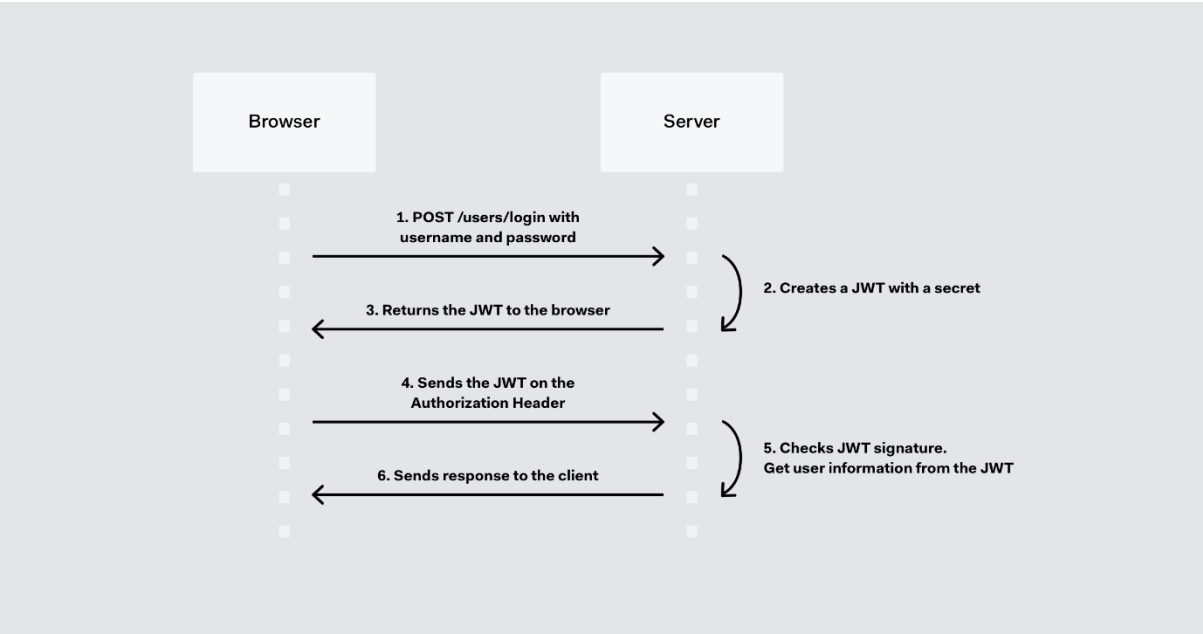


Figura 14: Esquema de funcionamiento del sistema JWT (https://auth0.com/learn/json-web-tokens/)

5. Implementación

En esta sección, comentaremos los requisitos e instrucciones necesarias para poder realizar la instalación del *frontend*, *backend* y base de datos de nuestro proyecto en un entorno de desarrollo local.

5.1. Requisitos de instalación

5.1.1. Cliente

Para poder ejecutar el código fuente del *frontend* en un ordenador en entorno local necesitaremos tener instalado en nuestra máquina Node.js y su administrador de paquetes *npm*.

5.1.2. Servidor

Asimismo, para ejecutar nuestro servidor en local, será necesario la instalación de Laravel Homestead, para preparar un entorno virtualizado con todos los componentes necesarios para su ejecución, tal y como está explicado en la documentación oficial de Laravel referenciada en el punto 4.3.1.3. En caso de no querer realizar esta instalación, deberemos asegurarnos de que nuestro ordenador dispone de un servidor local con, al menos, las características que podemos ver en la figura 15.

- PHP >= 7.2.5
- BCMath PHP Extension
- Ctype PHP Extension
- Fileinfo PHP extension
- JSON PHP Extension
- Mbstring PHP Extension
- OpenSSL PHP Extension
- PDO PHP Extension
- Tokenizer PHP Extension
- XML PHP Extension

Figura 15: Requisitos del servidor para ejecutar Laravel (https://laravel.com/docs/7.x#installation)

5.2. Instrucciones de instalación

5.2.1. Cliente

Una vez instalado Node.js y npm, deberemos, desde una consola de comandos, dirigirnos hasta el directorio donde se encuentre alojado el proyecto y ejecutar el comando "npm install". De este modo, instalaremos todas las dependencias necesarias para que el proyecto pueda funcionar sin problemas.

```
ruta_fronted/npm install
```

Una vez hecho esto, bastará con ejecutar el comando "ng serve" para arrancar un servidor local que apuntará al a dirección localhost:4200, desde donde podremos acceder a la aplicación web.

```
ruta_frontend/ng serve
```

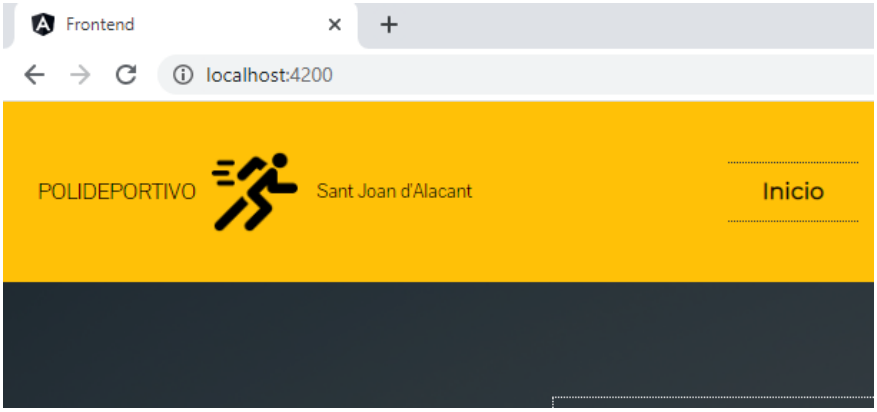


Figura 16: Frontend en localhost:4200

5.2.2. Servidor

Para ejecutar nuestro servidor, debemos copiar todos los archivos en la carpeta correspondiente. A continuación, deberemos instalar todas las dependencias y paquetes necesarios para asegurarnos su funcionamiento mediante la instalación de Composer.

```
ruta_api/composer install
```

Una vez instaladas todas las dependencias, tenemos que generar una clave de encriptación secreta que utilizará Laravel de manera interna para codificar varios elementos sensibles del sistema.

```
php artisan key:generate
```

A continuación, debemos crear una base de datos vacía, utilizando el software que mejor nos convenga, la cual alojará las tablas que generará Laravel para que nuestro proyecto pueda almacenar la información correctamente.

Una vez creada la base de datos, debemos editar el archivo .env, que se encuentra en la raíz del proyecto, donde tenemos que introducir los datos de la base de datos creada para que Laravel pueda comunicarse con ella. Entre la información necesaria se encuentra el tipo de base de datos utilizada, la dirección del host, el puerto, el nombre de la base de datos y el nombre de usuario y contraseña de acceso.

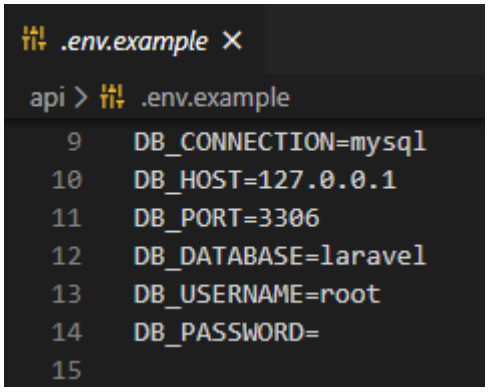


Figura 17: Ejemplo de configuración de archivo .env

Por último, debemos crear las tablas pertinentes en la base de datos creada. Para ello haremos uso de las *migrations* de Laravel, donde tenemos todas las tablas precodificadas y que serán añadidas a la base de datos ejecutando el comando *migrate*.

```
php artisan migrate
```

Ahora ya tenemos el servidor configurado y tal solo tenemos que ejecutarlo mediante el comando *serve*. Ahora, el servidor estará ejecutándose en local a través del puerto 8000.

```
php artisan serve
```

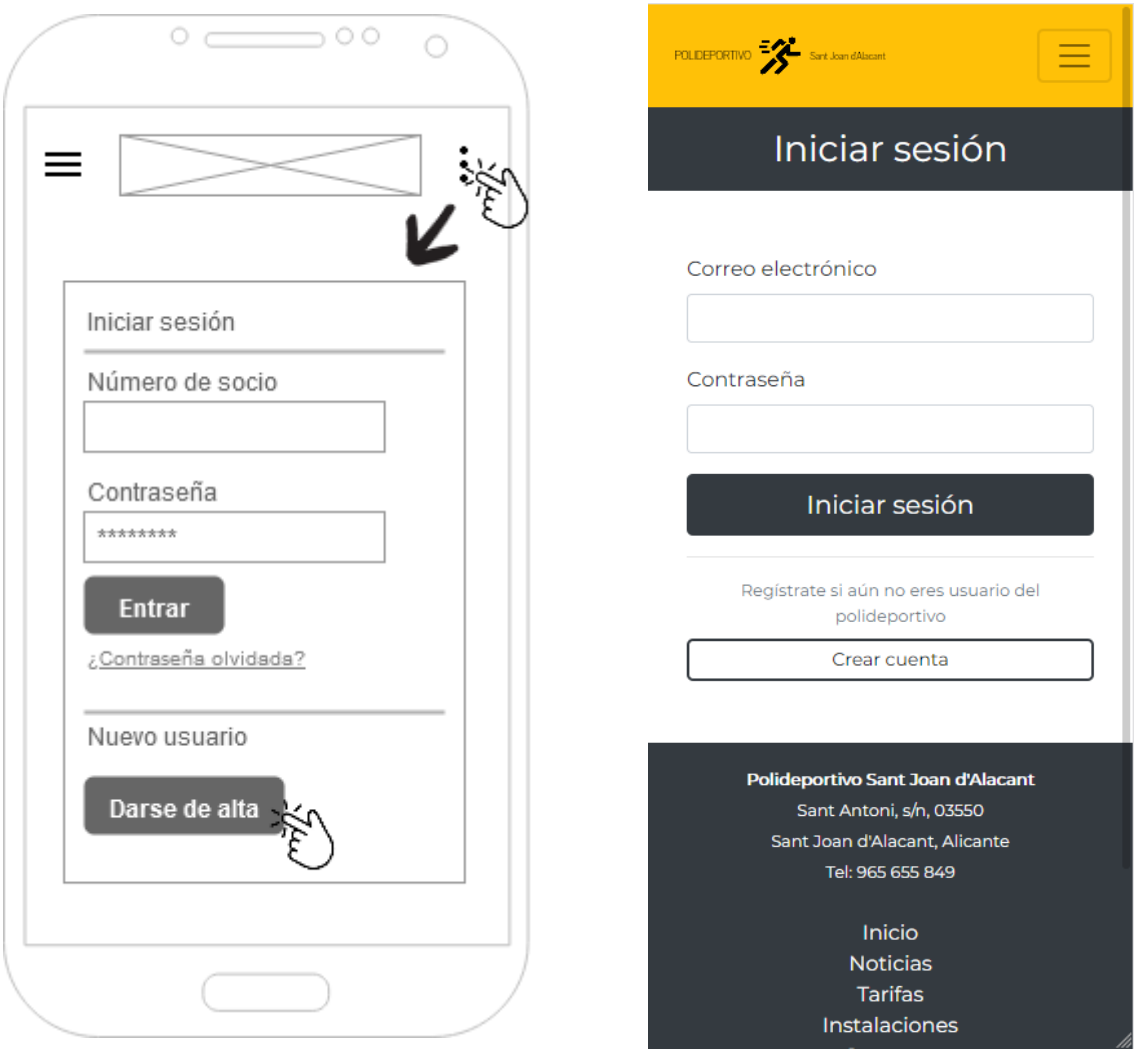
6. Prototipos y esquema de navegación

En este apartado se muestran los distintos prototipos que se han realizado durante la realización del proyecto. Por un lado, veremos los wireframes creados de las distintas páginas que componen la aplicación web y, por otro, veremos el diseño de la base de datos.

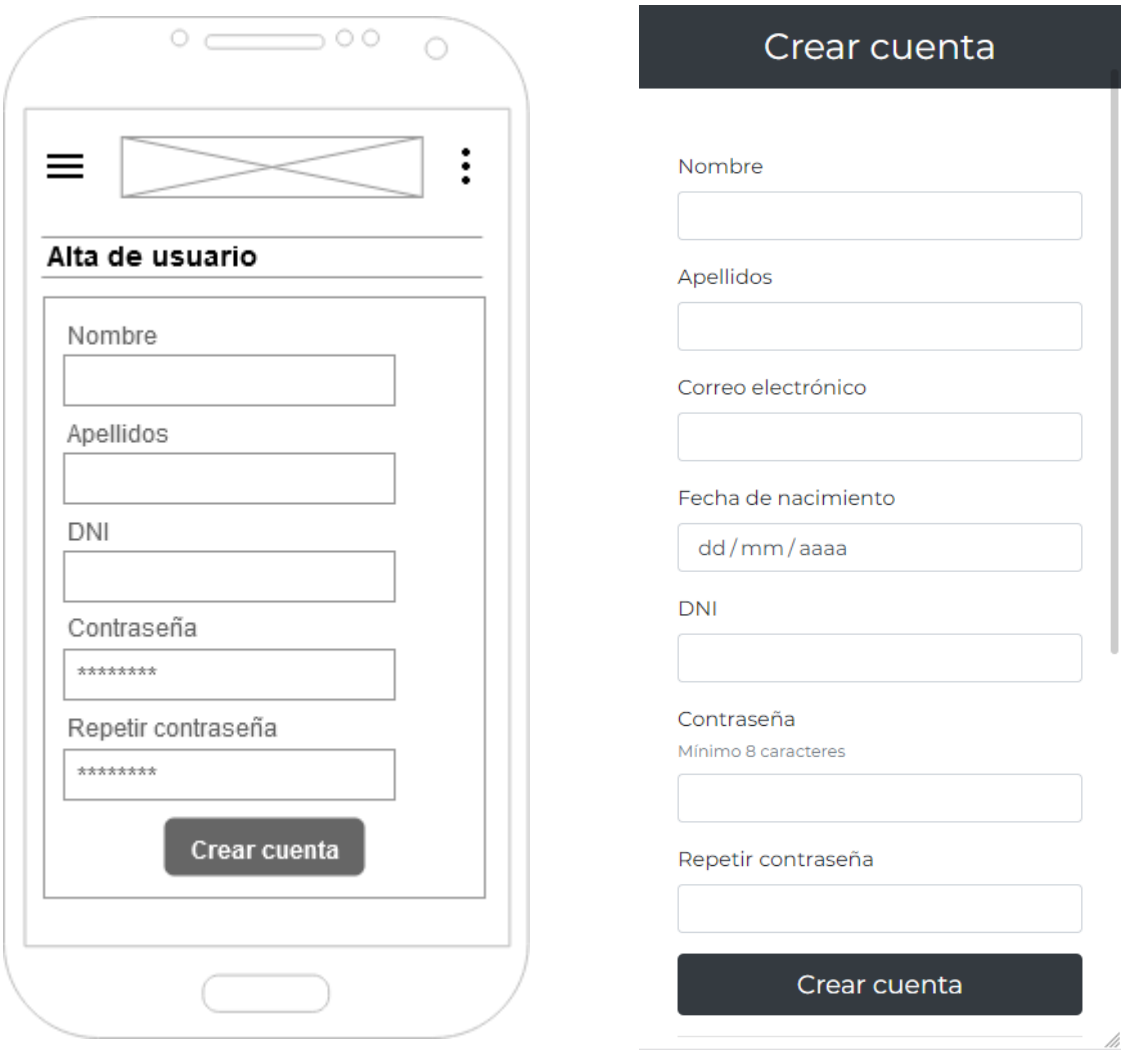
6.1. Wireframes

1. Página de inicio. Desde aquí podemos desplegar el menú y navegar a todas las otras páginas que componen nuestro sitio. Se puede acceder a esta página en cualquier momento pulsando sobre el logotipo o desde Menú > Inicio.

2. Menú de navegación. Podemos ver los enlaces para acceder al perfil y para cerrar sesión, así como el acceso al formulario de reservas. Un usuario que no ha iniciado sesión vería un enlace a "Iniciar sesión" en lugar de "Perfil" y "Cerrar sesión" y no vería el enlace a "Reservar".



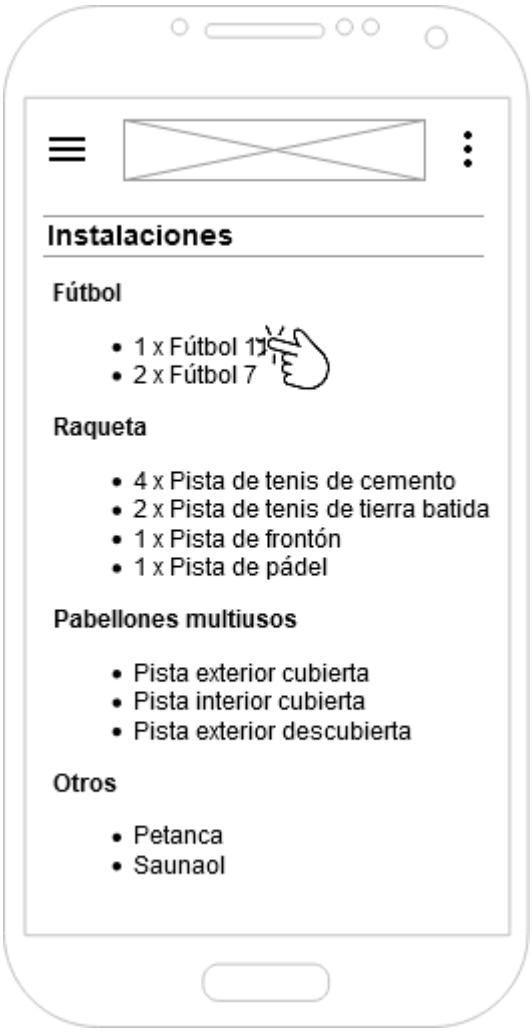
3. Inicio de sesión. Formulario de inicio de sesión, al cual se accede a través del enlace del menú. Como vemos, se ha cambiado el criterio de acceso planteado en el wireframe inicial, que preveía el uso de un botón en la parte derecha y finalmente se ha incluido como un enlace dentro del propio menú.

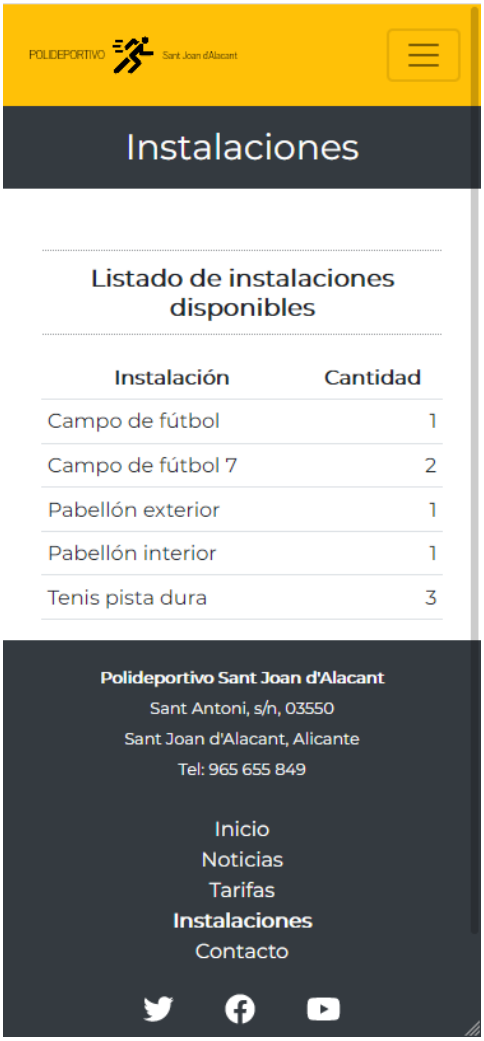


4. Registro de usuario. Formulario para el registro de usuario, al cual se accede desde la página de inicio de sesión, pulsando el botón "Crear cuenta".

5. Noticias. Página que muestra las últimas noticias relacionadas con el deporte de la localidad y con el polideportivo. Accesible desde el menú pulsando sobre "Noticias".

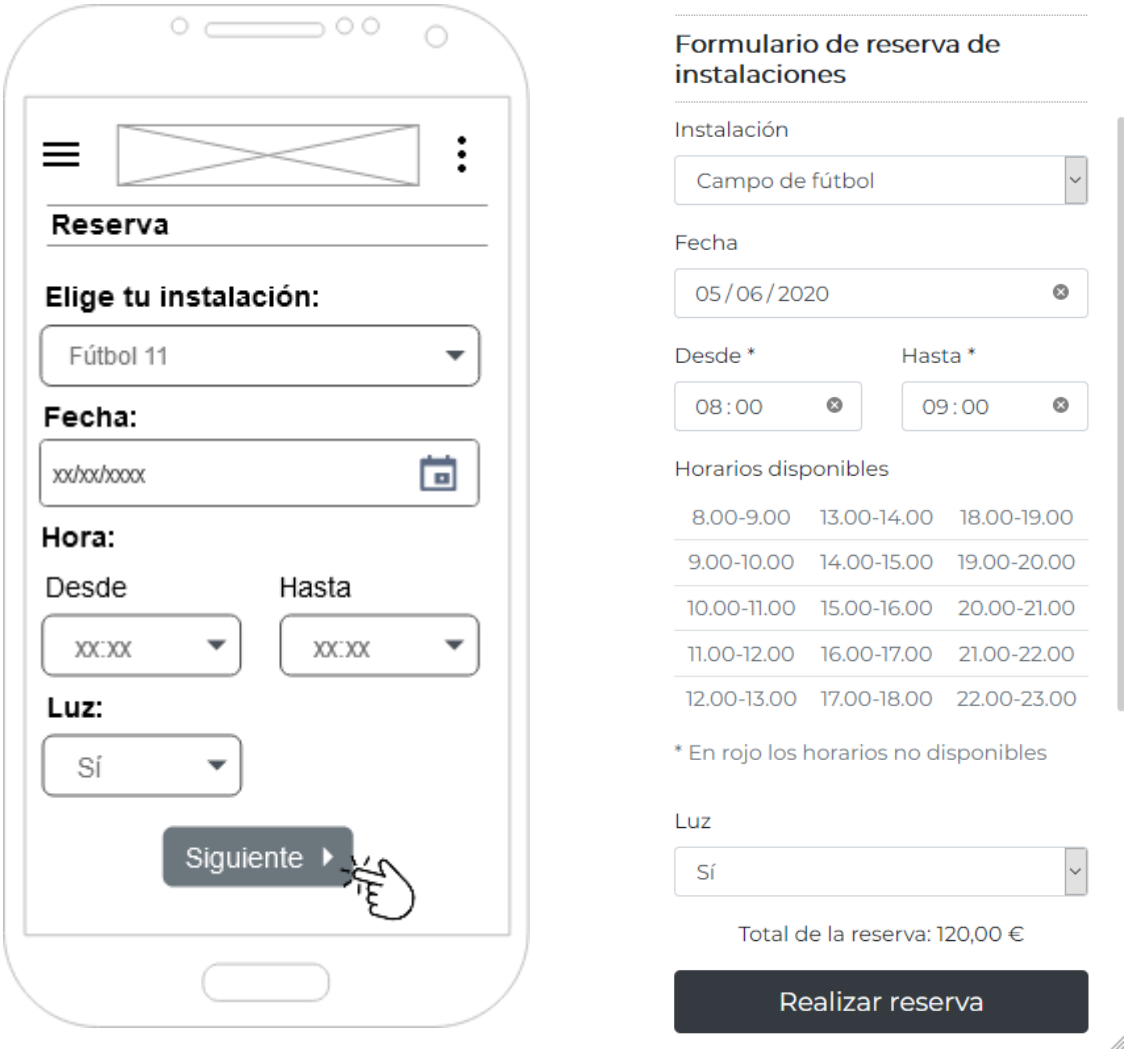
6. Noticia individual. Página que muestra una noticia en concreto. El diseño final ha variado respecto a la maquetación. La imagen se ha colocado entre el subtítulo de la noticia y el cuerpo de esta, al mismo tiempo que se ha añadido la fecha y el lugar de la noticia. Esta página es accesible pulsando sobre el titular de la noticia en la página Noticias.



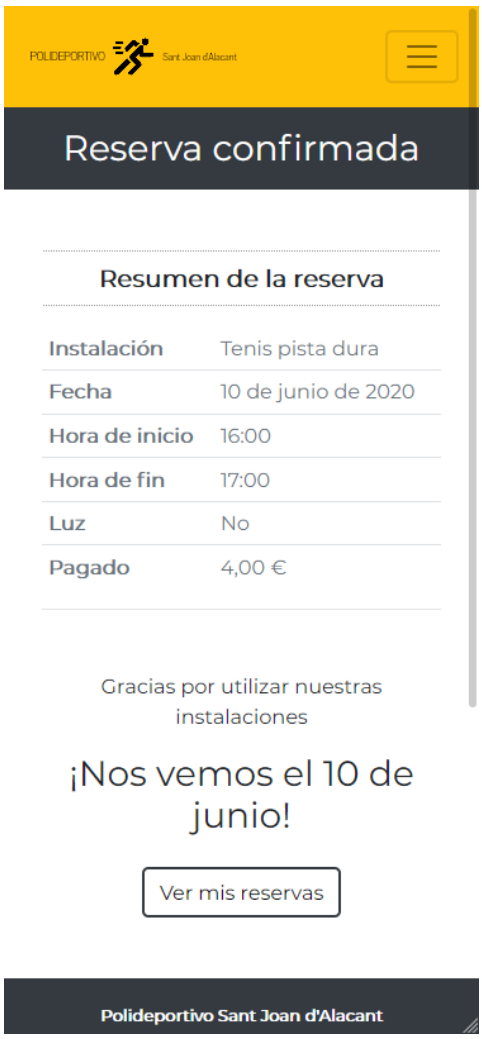


7. Instalaciones. Listado de las distintas instalaciones disponibles en el polideportivo. La página es accesible directamente desde el menú pulsando sobre el botón "Instalaciones".

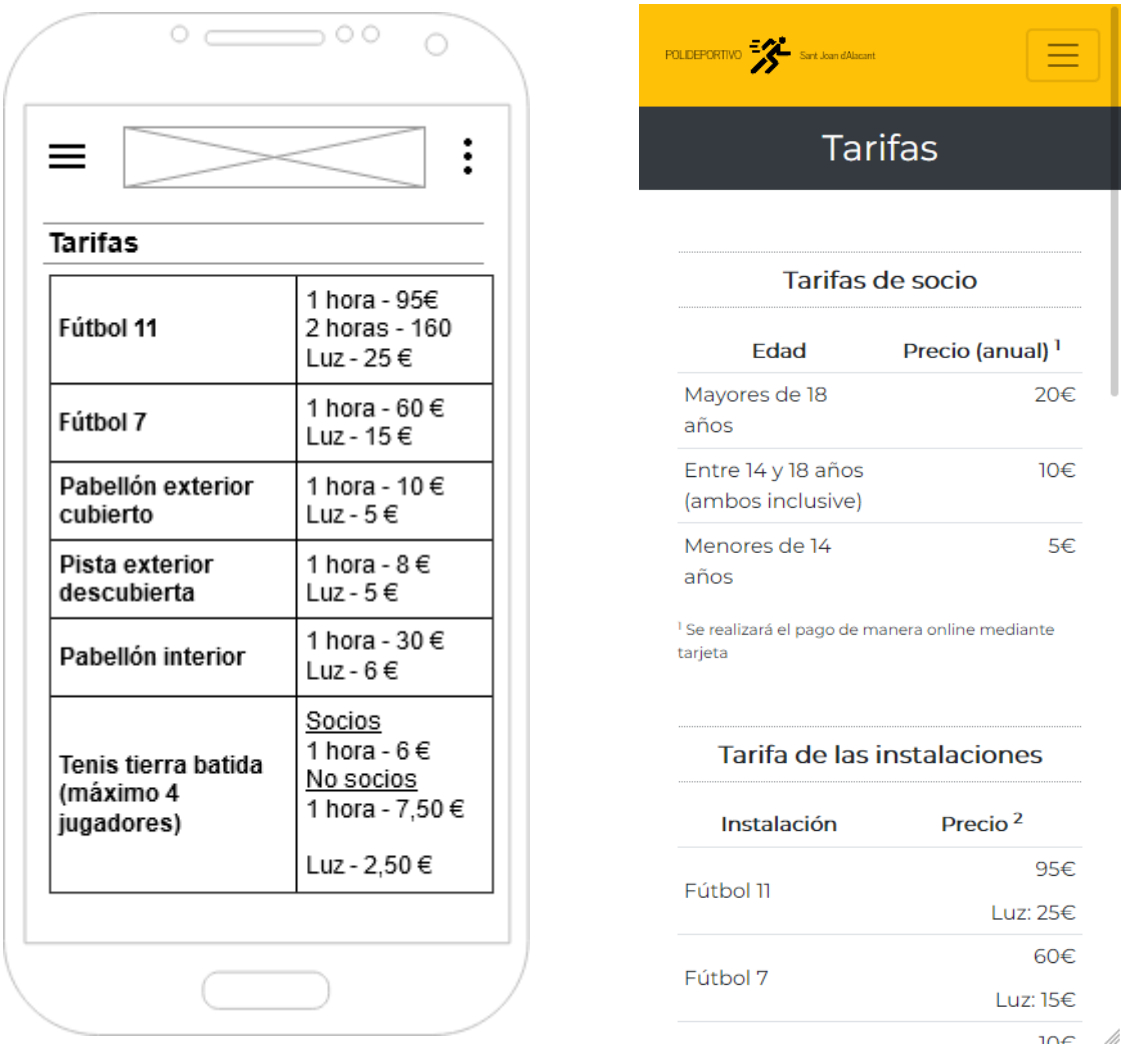
8. Detalle de instalación. Información detallada de la instalación seleccionada. Esta página es accesible desde la página de instalaciones, pulsando sobre cualquier instalación.



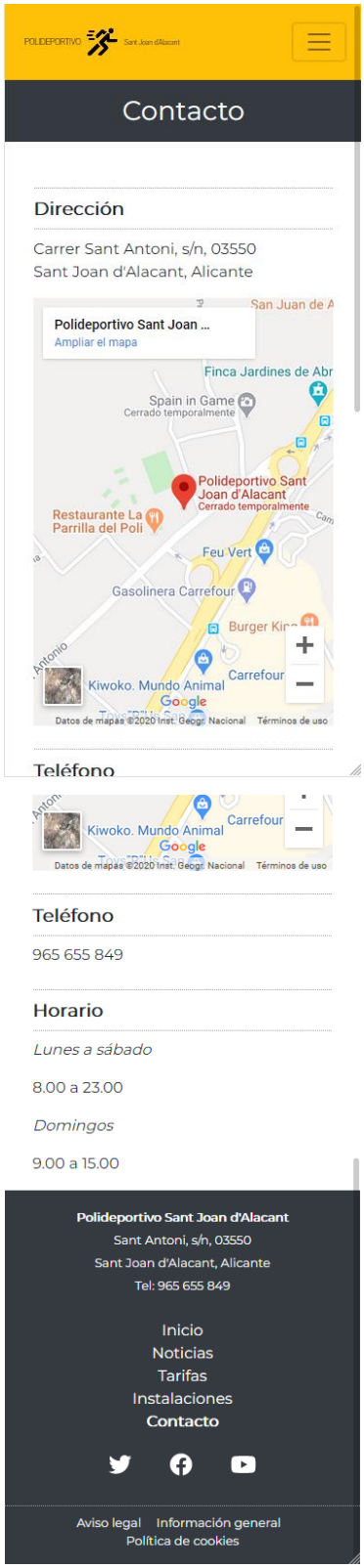
9. Formulario de reserva. Página para realizar la reserva de las distintas instalaciones. Esta página es accesible tan solo para usuarios registrados y con la sesión iniciada. Se accede directamente desde el menú a través del enlace "Reservar".



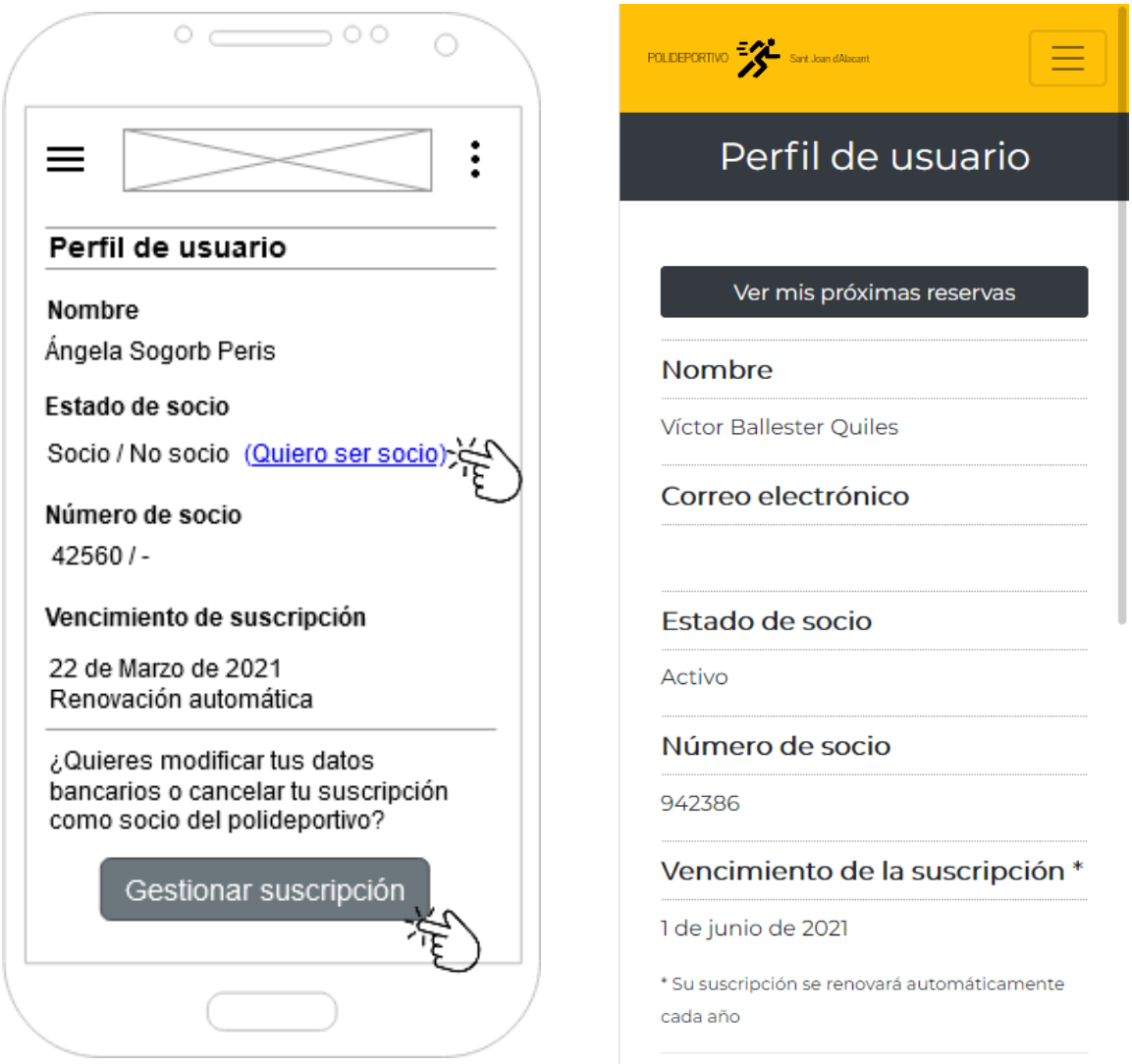
10. Resumen de la reserva. Se accede a esta pantalla a través de la pasarela de pago de Stripe, que es la encargada de redirigir al usuario una vez ha confirmado el pago por una de las instalaciones del polideportivo.



11. Tarifas. Página con la información de las tarifas de socio y de las distintas instalaciones disponibles en el polideportivo. Esta página es accesible desde el menú a través del enlace "Tarifas".



12. Contacto. Página con la información de contacto. Esta página es accesible desde el menú a través del enlace "Contacto".



13. Perfil de usuario. Página con la información del usuario. Esta página tan solo es accesible para usuarios registrados y con la sesión iniciada. Desde esta página se pueden consultar las reservas hechas por el usuario y suscribirse o de suscribirse como socio del polideportivo.

A lo largo del desarrollo del proyecto se ha tomado la decisión de no implementar algunas de las páginas que habían sido prototipadas en un inicio. A continuación, vemos las imágenes los prototipos que han sido finalmente descartados.

Estos diseños han sido descartados debido a la forma en la que hemos decidido gestionar finalmente los pagos en la aplicación web. En un principio se pensó realizar la gestión desde nuestro propio servicio, asumiendo la responsabilidad de gestionar información sensible como números de tarjetas de crédito o cuentas bancarias. Finalmente, decidimos utilizar un servicio de terceros como Stripe para implementar una pasarela de pago externa. Por lo tanto, a la hora de gestionar las suscripciones y las cancelaciones, el proceso se realiza a través de este servicio y no son necesarias estas pantallas.

6.2. Esquema entidad-relación de la base de datos

En cuanto al diseño de la base de datos, en este subapartado vamos a mostrar el esquema entidad-relación, que muestra todos los elementos que participan en el entorno de nuestra aplicación y de qué manera se relacionan entre ellos.

El esquema mostrado en la figura 18 se corresponde con el primer boceto de base de datos realizado en la fase de prototipado. La base de datos final ha sufrido algunas variaciones, con la adición de algunas tablas y campos para almacenar información de la integración de la página web con el servicio de Stripe.

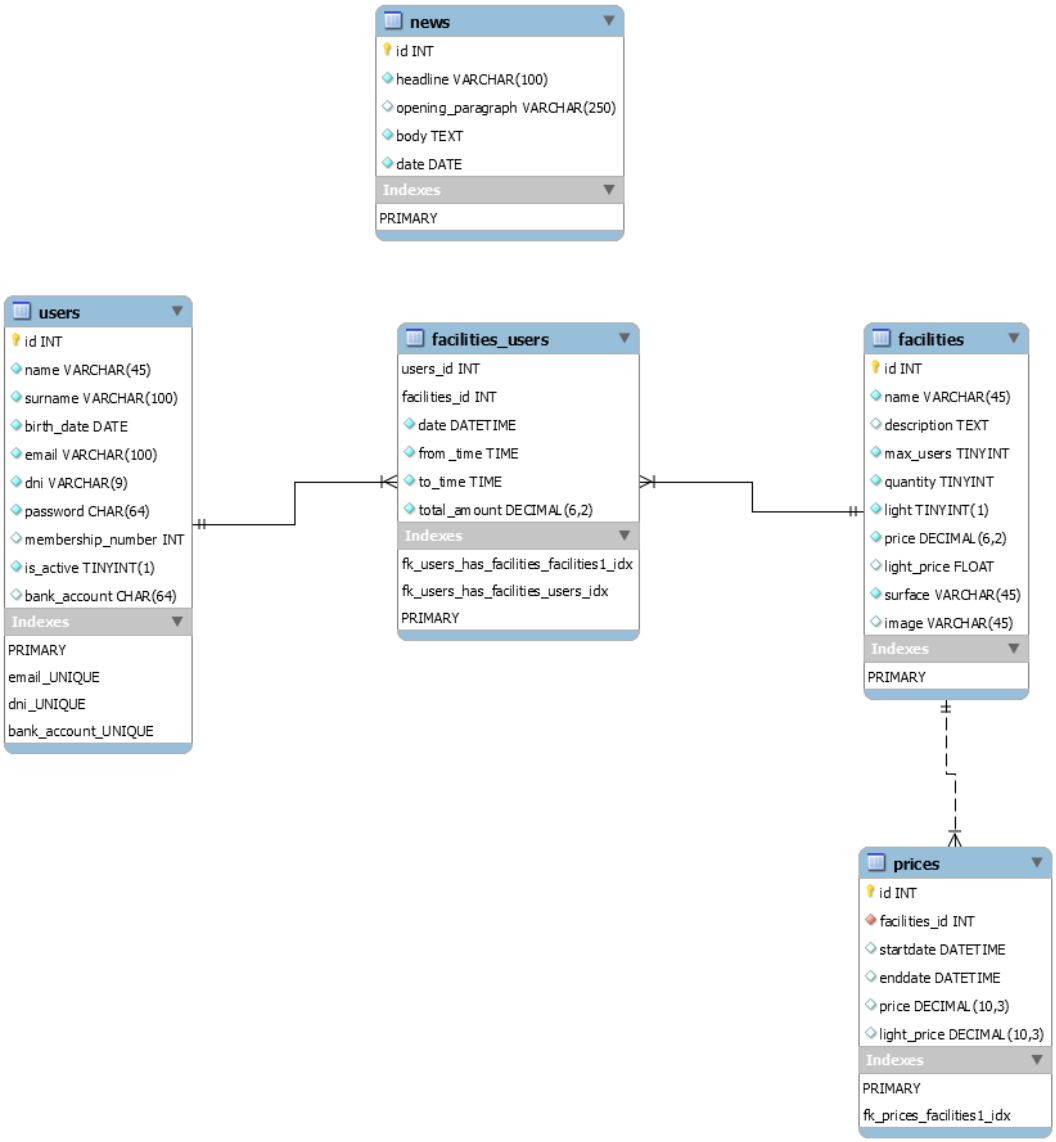


Figura 18: Primer esquema entidad-relación

El esquema final una vez finalizado el desarrollo del proyecto podemos verlo en la figura 19.

Figura 19: Esquema entidad-relación final

6.3. Sitemap

En la figura 20 podemos ver un mapa del sitio web elaborado en el proyecto, con todas las páginas disponibles y la navegación que puede realizar el usuario entre todas ellas.

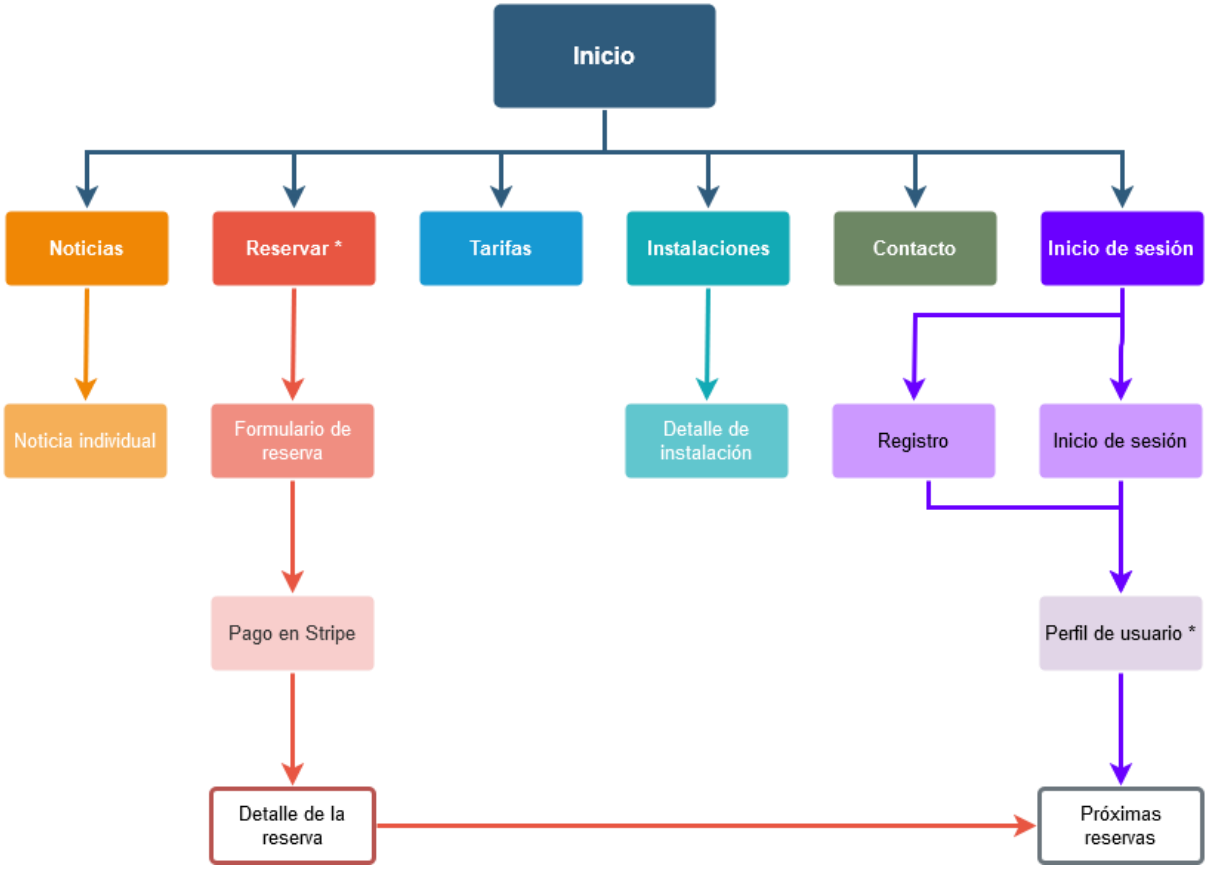


Figura 20: Sitemap visual del sitio web

6.4. Ejemplos de uso de la aplicación

En este subapartado mostraremos mediante una serie de diagramas de flujo los pasos a realizar para completar los procesos más relevantes del sitio web.

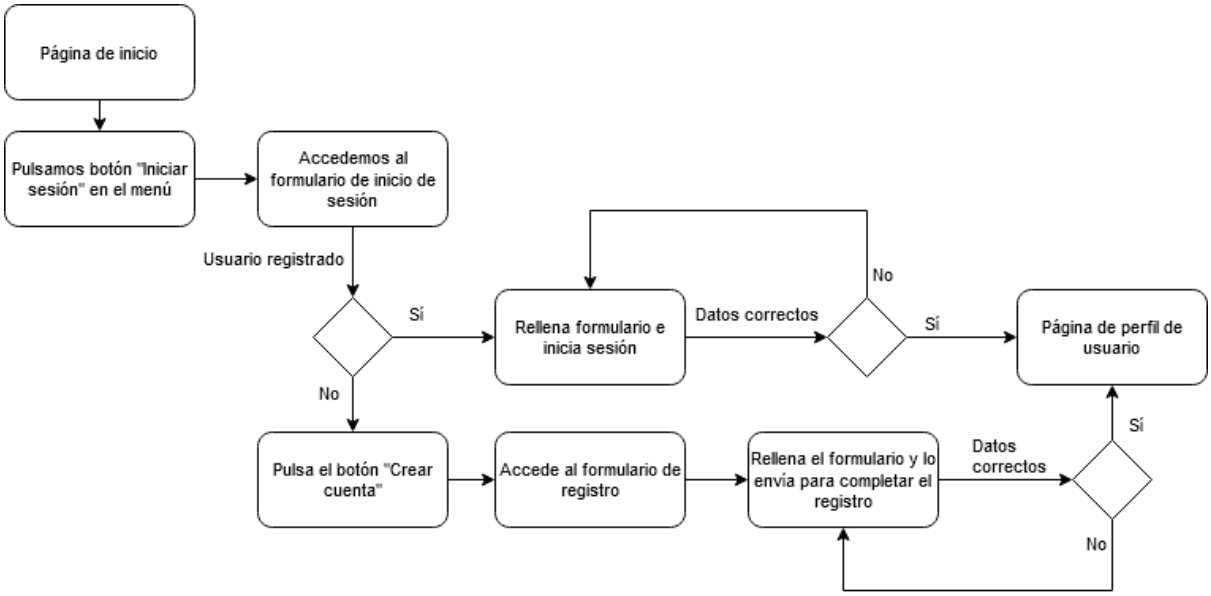


Figura 21: Proceso de inicio de sesión y registro de usuarios

Para los supuestos presentados en las figuras 22 y 23 es necesario que el usuario esté registrado y haya iniciado sesión previamente en la página, de lo contrario no podrá acceder al enlace "Reservas" del menú de inicio ni a su perfil de usuario.

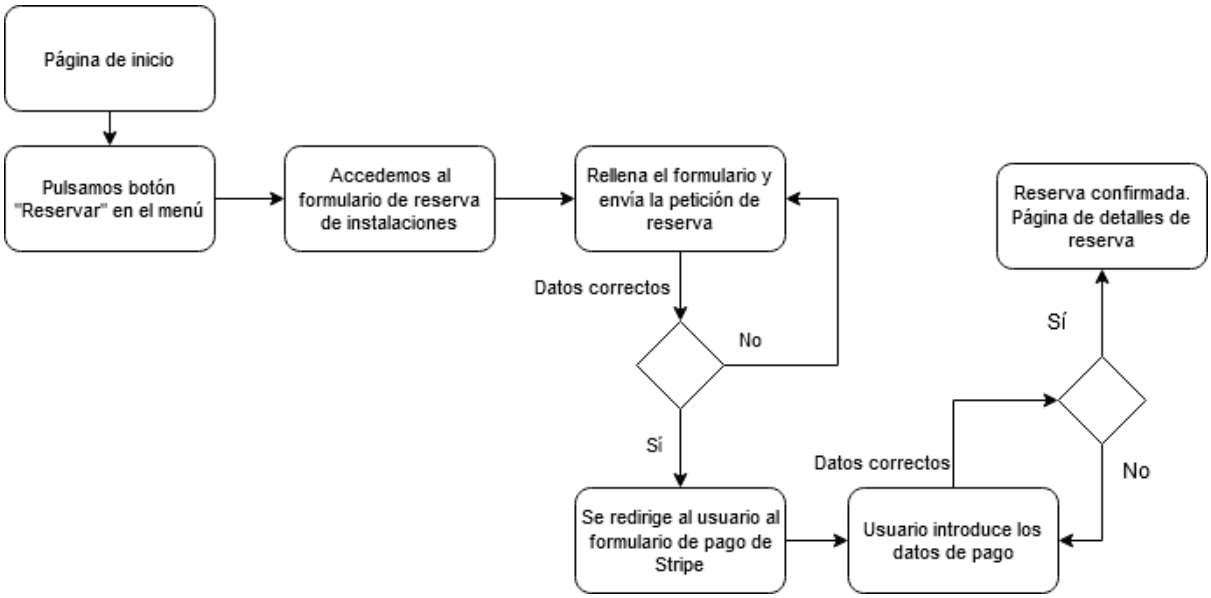


Figura 22: Proceso de reserva de una instalación

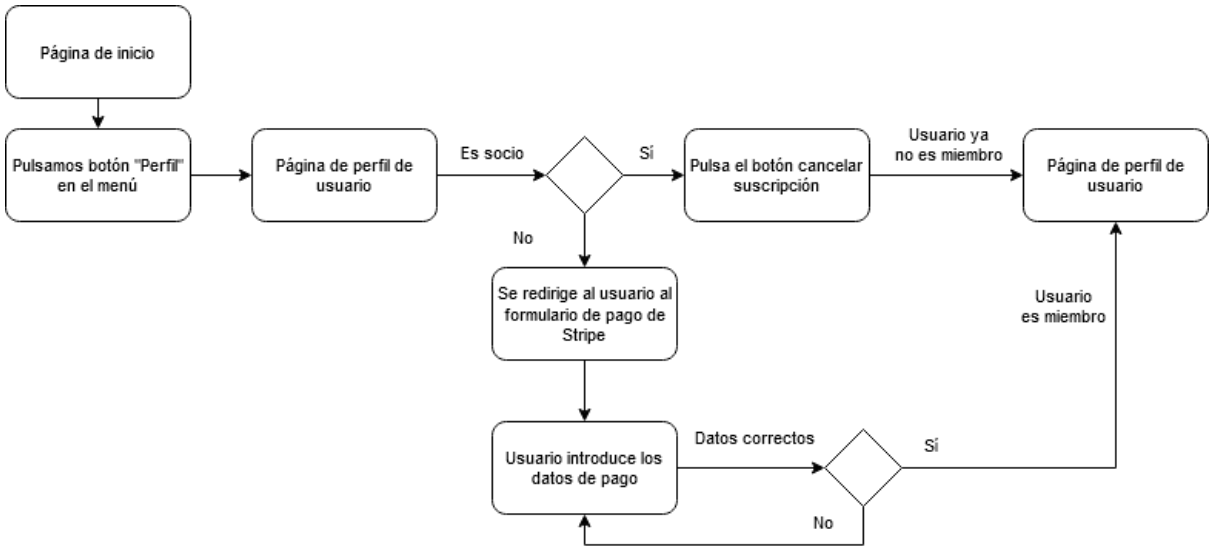


Figura 23: Proceso de gestión del estado de suscripción del usuario

7. Conclusiones y líneas de futuro

En el último apartado de esta memoria y como colofón al trabajo desarrollado durante los últimos meses, reflexionamos acerca de los resultados obtenidos gracias la elaboración de este proyecto, tanto desde el punto de vista personal como desde el punto de vista de la consecución de los distintos objetivos marcados al comienzo.

Además, también se comentan las posibles mejoras que se podrían realizar en futuras versiones del producto y sus posibilidades de crecimiento.

7.1. Conclusiones

A nivel personal, el desarrollo de este proyecto ha supuesto un desafío importante, tanto a nivel de organización y gestión del tiempo disponible, como a nivel de esfuerzo requerido para la consecución de los objetivos iniciales.

Debido a los tiempos tan convulsos que estamos atravesando, debido a la crisis derivada de la pandemia por coronavirus, mi trabajo me ha requerido muchas más horas de las habituales, por lo que la planificación inicial expuesta en el apartado 1.5. ha sido imposible de cumplir. Como consecuencia, hemos tenido que cambiar de enfoque y optar por una metodología de desarrollo alternativa, mediante la cual, a medida que hemos ido adquiriendo los conocimientos necesarios a través de los distintos cursos realizados, hemos ido desarrollando partes funcionales de la aplicación tanto a nivel de *frontend*, como de base de datos y *backend*. Por esto, la aplicación ha ido creciendo paso a paso de manera conjunta en todos sus niveles, en lugar de realizar un desarrollo en cascada como estaba previsto en un principio. Gracias a la adopción de una metodología ágil, hemos tenido mayor flexibilidad para adaptarnos a los cambios e imprevistos y hemos podido crear un producto funcional hasta donde nuestros conocimientos y el tiempo disponible nos ha permitido.

Pese a los problemas de planificación que hemos afrontado, hemos conseguido completar los objetivos principales marcados inicialmente. La lista de funcionalidades que posee la aplicación queda reflejada en el siguiente listado:

- Hemos desarrollado una solución web totalmente adaptable, usable desde diversos dispositivos independientemente del tamaño de sus pantallas.
- El usuario es capaz de consultar la información deportiva que afecta a su localidad, a través de la página de noticias.

- Es posible realizar el registro de un usuario y el inicio de sesión que permite el acceso a áreas privadas de la web de manera exitosa.
- Un usuario registrado es capaz de reservar una instalación para una fecha y horario concretos y realizar su pago de manera online a través de la plataforma Stripe.
- Un usuario puede gestionar su suscripción como socio del polideportivo y realizar los pagos de manera online también a través de la plataforma Stripe.

Con todo, el proyecto ha servido para aprender a utilizar nuevas herramientas que son muy demandadas en la actualidad dentro del mundo del desarrollo web, como son Angular y Laravel, además de probar nuestra capacidad de sacrificio y de trabajo bajo presión afrontando una fecha de entrega concreta.

Por todo ello, el proceso de desarrollo del proyecto expuesto en esta memoria, pese a sus dificultades y contratiempos, ha sido muy enriquecedor a nivel personal y nos ha proporcionado experiencias que seguro serán muy beneficiosas para nuestro futuro laboral.

7.2. Líneas de futuro

El proyecto ha dado como resultado una página web funcional, que permite realizar las tareas básicas de registro de usuarios, inicio de sesión y pago de instalaciones y suscripciones de manera online.

A pesar de ello, todavía han quedado elementos por depurar y mejorar, así como funcionalidades que podrían añadirse en un futuro para enriquecer la experiencia de uso del producto. Mostramos algunas de las mejoras que se podrían implementar en el siguiente listado:

- **Mejorar el sistema de autenticación de usuarios.** En la actualidad utilizamos un sistema basado en tokens para realizar la autenticación de usuarios. Cuando un usuario inicia sesión, el servidor, tras validar sus credenciales, envía un token generado de manera aleatoria al usuario. Este usuario es responsable de enviar dicho token en cada petición realizada, de modo que el servidor pueda comprobar que proviene de una fuente autorizada. Idealmente, este token debe caducar en un período corto de tiempo (10-15 minutos) y se debe renovar una vez expirado, de manera que, si un usuario malicioso intercepta nuestro token, no disponga de tiempo suficiente para suplantar nuestra identidad. En nuestro sistema, este token no caduca, ya que no hemos sido capaces de implementar la funcionalidad de refresco con éxito, sin embargo, es algo en lo que habría que invertir tiempo, ya que mejoraría en gran medida la seguridad del producto.

- **Mejorar el diseño visual.** A pesar de que el diseño visual de una página web es algo muy personal, este podría mejorarse y personalizarse todavía más. En este proyecto hemos utilizado principalmente diseños predefinidos del *framework* Bootstrap, ya que permiten desarrollar una web adaptable y con unos estilos sencillos pero efectivos de manera rápida. El objetivo que perseguíamos con este proyecto no era el de mejorar nuestras habilidades con el lenguaje CSS y se ha preferido centrar los esfuerzos en otros aspectos del desarrollo.

- **Mejorar página de instalaciones.** Una mejora que teníamos en mente pero que no hemos podido realizar por falta de tiempo es la de combinar la página del listado de instalaciones con la de las instalaciones individuales. La idea es realizar un diseño mediante un sistema de acordeón[37], como el ofrecido por Bootstrap, para poder ver el listado de instalaciones y sus detalles debajo de cada elemento de la lista cuando el usuario pulsa encima de ellas.

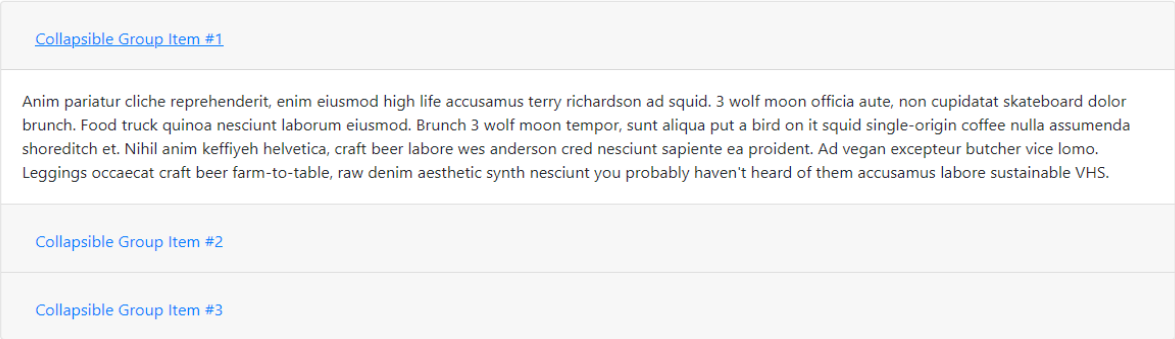


Figura 24: Ejemplo de diseño en acordeón de Bootstrap

- **Distintos roles de usuario.** Una mejora que sería necesaria para mejorar el mantenimiento del servicio sería la creación de distintos roles para los usuarios de la aplicación, como administrador, editor y usuario. Un editor, además de las funciones propias de un usuario, sería el encargado de poder añadir, editar y eliminar las noticias que aparecen en la página web, mientras que un administrador, además, sería capaz de gestionar las reservas o las suscripciones (cancelarlas, posponerlas o modificarlas) desde la propia interfaz del sitio web, sin tener que modificar la base de datos de manera directa.

[37] https://getbootstrap.com/docs/4.3/components/collapse/#accordion-example

Bibliografía

Ajuntament de Sant Joan d'Alacant (2004). Ordenanza fiscal reguladora del precio público por la utilización de instalaciones deportivas del organismo autónomo local "Fundación Deportiva Municipal de Sant Joan d'Alacant" y escuelas deportivas municipales (en línea) [Consultado abril 2020] http://www.santjoandalacant.es/sites/default/files/2017-12/ORDENANZAS%20Instalaciones.pdf

CORCHADO, Beatriz (2018). ¿Qué es la experiencia de usuario y cómo se diferencia de la usabilidad? (en línea) [Consultado mayo de 2020] https://es.godaddy.com/blog/que-es-la-experiencia-de-usuario-y-como-se-diferencia-de-la-usabilidad/?xpPartner=HEI

DevMarketer (2018). How to Setup a Laravel Project You Cloned from Github.com (en línea) [Consultado mayo 2020] - https://devmarketer.io/learn/setup-laravel-project-cloned-github-com/

DÍAZ, Juan (2017). Curso Laravel (en línea) [Consultado Abril 2020] https://www.youtube.com/watch?v=0sHSrqyZCnM&list=PLU8oAlHdN5Bk-qkvjER90g2c_jVmpAHBh

MANZO, Jean (2017). API vs Webhooks (en línea) [Consultado Junio 2020] https://medium.com/@jdevmanzo/api-vs-webhooks-4745bffcfa65

PEARSON, Chuck (2016). Typography on the web (en línea) [Consultado 22 mayo 2020] https://medium.com/rareview/typography-on-the-web-4cd494d6b165

SINGH, Vijay (2020). What is Frameworks? [Definition] Types of Frameworks (en línea) [Consultado mayo 2020] - https://hackr.io/blog/what-is-frameworks

Stackoverflow (2019). Developer Survey Results (en línea) [Consultado mayo 2020] https://insights.stackoverflow.com/survey/2019#technology-_-web-frameworks

SCHWARZMÜLLER, Maximilian (2020). Angular – The Complete Guide (en línea) [Consultado abril 2020] https://www.udemy.com/course/the-complete-guide-to-angular-2/learn/lecture/13914134#overview

TICBeat Lab (2019). 5 claves para mejorar la tasa de rebote de tu página web (en línea) [Consultado mayo de 2020] - https://www.ticbeat.com/lab/5-claves-mejorar-tasa-rebote/

Anexos

Anexo A: Glosario

API. *Application Programming Interface* o interfaz de programación de aplicaciones. Se trata de una aplicación que proporciona una capa de abstracción para que dos aplicaciones web puedan comunicarse entre sí compartiendo información con independencia de cómo estén construidas.

Backend. Parte de una aplicación web encargada del acceso y la gestión de los datos.

Entidad-relación. Herramienta utilizada para el modelado de datos en la cual se muestran las entidades participantes dentro de un sistema concreto y la manera en la que se relacionan entre ellas.

Framework. Entorno de trabajo que proporciona un sistema robusto y seguro como base a partir de la cual crear contenido de manera más ágil y rápida.

Frontend. Parte una aplicación web encargada de gestionar la visualización y la capa de presentación de la información.

JSON. JavaScript Object Notation o notación de objecto de JavaScript. Es un formato de texto que se utiliza para intercambiar datos, principalmente entre un cliente y un servidor.

SPA. Single Page Application o aplicación de una sola página. Son páginas web que cargan todo su contenido de manera dinámica en una sola página sin realizar ningún tipo de recarga, actualizando los distintos módulos que la componen a medida que el usuario interactúa con la aplicación.

Anexo B: Entregables del proyecto

Memoria del proyecto

Se adjunta el presente documento como parte de los entregables del proyecto, donde se detallan las motivaciones para su realizació, la planificación del trabajo, el diseño e implementación realizados y las conclusiones extraídas de su desarrollo.

Código fuente de la API y el frontend

Se entregan dos archivos comprimidos por separado. Uno de ellos contiene el proyecto Angular con el código del *frontend* y el otro contiene el proyecto Laravel con el código del backend. Para su correcta instalación deben seguirse los pasos mencionados en el apartado "5.2. Instrucciones de instalación".

Presentación en PowerPoint

Se entrega una presentación del proyecto en PowerPoint donde se resume el trabajo realizado a lo largo de todo el proyecto y se comentan las motivaciones para acometer su desarrollo.

Vídeo demostración del producto

Vídeo donde se realiza una demostración de uso del producto, donde podemos ver todas las secciones que componen el sitio web, el comportamiento tanto en versión móvil como de escritorio y los pasos para realizar las tareas principales de la aplicación.

Hoja de autoevaluación

Documento donde realizamos una evaluación crítica del trabajo realizado durante la realización del proyecto.

Vídeo presentación del proyecto

Vídeo donde, utilizando la presentación PowerPoint y el vídeo de demostración, realizamos la defensa del proyecto elaborado.

Anexo C: Capturas de pantalla

A continuación, se muestran capturas de pantalla de las partes más relevantes de nuestro código, que se encargan de realizar las principales funcionalidades de la página web.

En la figura 25 podemos observar la función *store*, que se encarga de recibir en el *backend* los datos que provienen del formulario de registro. Antes de almacenar los datos, que han sido validados en el cliente, se vuelven a validar de nuevo, asegurándonos de encriptar la contraseña para una mayor seguridad.

```
public function store(Request $request)
{
    $validatedData = $request->validate([
        'name'       => ['bail', 'required', 'string'],
        'surname'    => ['bail', 'required', 'string'],
        'birth_date' => ['bail', 'required', 'date'],
        'dni'        => ['bail', 'required', 'size:9',
                         'regex:/^[0-9]{8}[TRWAGMYFPDXBNJZSQVHLCKE]$|[XYZ][0-9]{7}[TRWAGMYFPDXBNJZSQVHLCKE]$/i',
                        function ($dni, $value, $fail) {
                            $dni_letter    = substr($value, -1, 1);
                            $dni_number    = substr($value, 0,8);
                            $dni_number    = str_replace(['X','Y','Z'], [0, 1, 2], $dni_number);
                            $valid_letters = "TRWAGMYFPDXBNJZSQVHLCKE";

                            if(substr($valid_letters, $dni_number%23, 1) !== $dni_letter) {
                                $fail('DNI is invalid');
                            }
                        }],
        'password'   => ['bail', 'required', 'min:8'],
        'email'      => ['bail', 'required', 'email']
    ]);

    $user = new User;

    $user->name       = $request->name;
    $user->surname    = $request->surname;
    $user->birth_date = $request->birth_date;
    $user->dni        = $request->dni;
    $user->password   = Hash::make($request->password);
    $user->email      = $request->email;

    $user->save();

    return response('{"Message":"Usuario registrado"}', 201)
            ->header('Content-Type', 'text/plain');
}
```

Figura 25: Función de registro de usuarios

Validar el DNI conlleva una serie de cálculos más complejos, que en el *frontend* se han realizado a través de un validador personalizado, como podemos ver en la figura 26.

```
export class DniValidatorDirective implements Validator {

  validate(control: AbstractControl): ValidationErrors | null {

    const dni = control.value;
    const dniRegex = /[0-9]{8}[A-Z]{1}|[XYZ]{1}[0-9]{7}[A-Z]{1}/gi;

    if (dniRegex.test(dni)) {
      const letter = dni.substring(dni.length-1);
      const number = dni.substring(0, dni.length - 1);
      //Corrects number to validate both NIF and NIE numbers
      const corrected_number = number.replace(/[X-Z]/, function(letter: string) {
        if (letter === 'X') {
          return 0;
        } else if (letter === 'Y') {
          return 1;
        } else if (letter === 'Z') {
          return 2;
        }
      });
      const valid_letters = "TRWAGMYFPDXBNJZSQVHLCKE";
      const correct_letter = valid_letters.charAt(corrected_number%23);

      if (correct_letter === letter) {
        return null;
      } else {
        return { appDniValidator: true };
      }
    } else {
      return { appDniValidator: true };
    }
  }
}
```

Figura 26: Custom validator en Angular para validar el DNI

En la figura 27, podemos ver la función que gestiona el proceso de inicio de sesión en el backend. La función recibe los parámetros de la petición (correo electrónico y contraseña) e intenta validarlos. Si la validación es correcta, devuelve una respuesta en formato JSON el id de usuario que ha iniciado sesión, el token de acceso generado, el tipo de token y el tiempo de expiración (que en nuestro caso no utilizamos ya que hemos configurado el sistema para que no expire nunca).

El usuario que recibe esta información usa este token para validar cada una de las peticiones posteriores que realice.

```php
public function login(Request $request)
{
    $credentials = $request->only('email', 'password');
    $email = $request->email;

    if ($token = $this->guard()->attempt($credentials)) {
        return response()->json([
            'user' => User::where('email', $email)->first('id'),
            'access_token' => $token,
            'token_type' => 'bearer',
            'expires_in' => $this->guard()->factory()->getTTL() * 60
        ]);
        //return $this->respondWithToken($token);
    }

    return response()->json(['error' => 'Unauthorized'], 401);
}
```

Figura 27: Función de inicio de sesión

Para realizar el proceso de reserva de una instalación, el usuario debe rellenar un formulario con todos los datos de fecha, horario, instalación requerida y si desea también iluminación. Al enviar el formulario se ejecuta la función *onBooking*, que recibe como parámetro el formulario enviado. Si el formulario es válido y las fechas y horas elegidas están disponibles, comienza con la gestión de los datos para generar el pago correspondiente.

En primer lugar, se crea una variable con los datos necesarios para que Stripe pueda generar una sesión de pago correcta desde el servidor. Para ello, necesitamos obtener desde nuestra base de datos los *id* de los productos que queremos reservar. Una vez obtenemos los datos y se ha completado la variable *request*, se realiza una nueva llamada al servidor, al cual se le envía dicha variable para que devuelva un id de sesión válido. Cuando recibimos la respuesta, añadimos los datos de la reserva a la base de datos (a la espera de que el pago sea confirmado) y, con el id de sesión generado por Stripe en el servidor redirigimos al usuario a la pasarela de pago para que confirme la transacción.

La función completa podemos verla en la figura 28 en la página siguiente.

```
onBooking(form: NgForm) {
  if (form.valid == !this.already_booked) {

    this.loading = true;

    this.calculateTotalAmount();

    //Get Stripe pricesId and set the line_items object
    this.http.get('//localhost:8000/api/prices/' + this.facilitySelected).subscribe( prices_id => {
      this.facility_price_id = this.user.is_active && prices_id['facility_price_member_id'] ?
                                prices_id['facility_price_member_id'] : prices_id['facility_price_id'];
      this.light_price_id = prices_id['light_price_id'];

      let request = {
        'email': this.user.email,
        'line_items': [
          {
          'price': this.facility_price_id,
          'quantity': this.hours
          }
        ],
        'mode': 'payment'
      }

      //Adds light price_id if exists
      if(this.light_price_id && this.lightSelected === 1) {
        request.line_items.push({'price': this.light_price_id, 'quantity': this.hours});
      }

       //Creates Stripe session in backend and redirects to checkout
       this.http.post('//localhost:8000/api/stripe', request).subscribe( async response =>{
         //Booking details to store in database
         let bookingData = {
           user: 1,
           facility: this.facilitySelected,
           date: this.dateSelected,
           start_time: this.start_time,
           end_time: this.end_time,
           light: this.lightSelected,
           total_amount: this.total_amount,
           token: response['id']
         }

         //Store booking data in DB before calling Stripe
         this.http.post('//localhost:8000/api/bookings', bookingData).subscribe( async () => {
           const stripe = await loadStripe('pk_test_eT7IJcXEfzDloYoASwoEqzCx00L7pDWLjc');
           this.loading = false;
           const {error} = await stripe.redirectToCheckout({
             sessionId: response['id']
           })
         });
       });
    });
  }
}
```

Figura 28: Función para realizar una reserva de una instalación

Para poder reaccionar a los distintos eventos que se generan cuando se realiza una sesión con Stripe, este servicio utiliza lo que se conocen como *webhooks*[38]. Esto no son más que llamadas que se realizan desde Stripe a una dirección de nuestra API que estará enlazada con un controlador específico cuyas funciones se encargarán de realizar diversas tareas dependiendo del tipo de evento generado.

En la figura 29 podemos ver la función *handleWebhook* que recibe las peticiones realizadas y almacena en la variable $event el tipo de evento recibido. En función de este evento, ejecutamos una serie de operaciones u otras.

```
class StripeWebhookController extends Controller
{
    public function handleWebhook(Request $request) {

        // Set your secret key. Remember to switch to your live secret key in production!
        // See your keys here: https://dashboard.stripe.com/account/apikeys
        \Stripe\Stripe::setApiKey('sk_test_7uRFAEbSSI2bg8NqiRU3yOXg00czGlVRRA');

        // You can find your endpoint's secret in your webhook settings
        $endpoint_secret = 'whsec_TnRXyCIBlRLKQpqU1zOcaRUbuq636uPZ';

        $payload = @file_get_contents('php://input');
        $sig_header = $_SERVER['HTTP_STRIPE_SIGNATURE'];
        $event = null;

        try {
            $event = \Stripe\Webhook::constructEvent(
                $payload, $sig_header, $endpoint_secret
            );
        } catch(\UnexpectedValueException $e) {
            // Invalid payload
            http_response_code(400);
            exit();
        } catch(\Stripe\Exception\SignatureVerificationException $e) {
            // Invalid signature
            http_response_code(400);
            exit();
        }
```

Figura 29: Controlador para evaluación del tipo de evento generado por Stripe

[38] https://medium.com/@jdevmanzo/api-vs-webhooks-4745bffcfa65

A continuación, se muestran las respuestas a los tres eventos que se han configurado en este controlador.

En la figura 30 vemos el código que se ejecuta cuando se genera un evento de tipo *checkout.session.completed*, que es enviado por Stripe cuando algún pago ha sido realizado con éxito. En este momento, actualizamos la tabla donde almacenamos todas las reservas realizadas para hacer constar que el pago ha sido aceptado y confirmar la reserva.

```
// Handle a single facility payment
if ($event->type == 'checkout.session.completed') {
    $session = $event->data->object;

    FacilitiesUsers::where('token', $session->id)->update(['payment_accepted' => 1]);
}
```

Figura 30: Manejador de evento para un proceso de reserva pagado con éxito

```
// Handles subscriptions payments
if ($event->type == 'invoice.payment_succeeded') {

    $session = $event->data->object;

    $email = $session->customer_email;
    $user = User::where('email', $email)->first('id');
    $subscription_id = $session->lines->data[0]->subscription;
    $customer = $session->customer;
    $start_date = date("Y-m-d", $session->lines->data[0]->period->start);
    $expiration_date = date("Y-m-d", $session->lines->data[0]->period->end);

    Subscription::where('users_id' , $user->id)->update([
        'stripe_subscription_id' => $subscription_id,
        'stripe_customer_id' => $customer,
        'active' => 1,
        'membership_start' => $start_date,
        'membership_end' => $expiration_date
    ]);

    $membership_number = $this->generateRandomNumber(100000, 999999);

    User::where('email', $email)->update([
        'is_active' => 1,
        'membership_number' => $membership_number
    ]);

}
```

Figura 31: Manejador de evento para una suscripción pagada con éxito

En la figura 31 que se muestra arriba, vemos el código que se ejecuta cuando Stripe emite un evento del tipo *invoice.payment_succeeded*, que se genera cuando una suscripción ha sido pagada con éxito.

Entonces, obtenemos todos los datos necesarios de la suscripción y los almacenamos en la tabla *subscriptions*, al mismo tiempo que actualizamos el estado de nuestro usuario en la tabla *users* como activo.

En este proceso, se genera un número de socio aleatorio que se asigna al usuario que se intenta suscribir como socio. El proceso para generar este número y comprobar que no se repita en nuestra tabla de usuarios lo realiza la función recursiva *generateRandomNumber*, que podemos ver en la figura 32.

```
protected function generateRandomNumber($min, $max) {

    $number = mt_rand($min, $max);

    if ($this->membershipNumberExists($number)) {
        return generateRandomNumber();
    }

    return $number;
}
```

Figura 32: Función recursiva que genera el número de socio de manera aleatorio

Por último, en la figura 33 podemos ver el código que se ejecuta cuando el evento recibido por Stripe es del tipo *customer.subscription.updated*. Este evento es generado por Stripe cuando un usuario cancela o reactiva su suscripción, modificando el valor del atributo *cancel_at_period_end* a *true* o *false* respectivamente. De este modo, se indica que la suscripción permanecerá activa hasta el final del periodo contratado y luego será cancelada o bien que seguirá activa y se renovará año a año.

```
if ($event->type == 'customer.subscription.updated') {

    $session = $event->data->object;

    Subscription::where('stripe_subscription_id', $session->id)->update([
        'cancel_at_period_end' => $session->cancel_at_period_end
    ]);
}
```

Figura 33: Manejador de evento para modificar suscipciones

Para finalizar este anexo, mostramos la figura 34, en la que puede verse el servicio *AuthService* con las distintas funciones encargadas de gestionar la autenticación desde el lado del cliente.

Podemos distinguir principalmente tres funciones, como son *signup*, *login* y *logout*. En cada una de ellas se realiza una llamada al servidor, que será el responsable de gestionar cada uno de estos tres procesos de autenticación.

La función *signup* se utiliza para registrar un usuario en nuestra base de datos. Cuando recibimos la respuesta del servidor, devolvemos el Observable a la función principal que reacciona al flujo de datos recibido y llama a la función *login* para automáticamente iniciar la sesión del usuario registrado y redirigirlo a la página del perfil de usuario.

```
@Injectable({providedIn: 'root'})
export class AuthService {

    private currentUserSubject: BehaviorSubject<any>;
    private loginError: BehaviorSubject<any>;

    constructor(private http: HttpClient, private router: Router) {
        this.currentUserSubject = new BehaviorSubject<any>(JSON.parse(localStorage.getItem('currentUser')));
        this.loginError = new BehaviorSubject<any>(false);
    }

    signup(userData: UserData) {
        return this.http.post<any>('//localhost:8000/api/users', userData, { withCredentials: true });
    }

    login(loginData: LoginData) {
        this.http.post<any>('//localhost:8000/api/auth/login', loginData).subscribe( response => {
            localStorage.setItem('token', response.access_token);
            localStorage.setItem('user', response.user.id);

            this.currentUserSubject.next(response.user.id);
            this.router.navigate(['../perfil-usuario']);
        }, loginError => {
            console.log(loginError);
            if(loginError.error.error === "Unauthorized") {
                this.loginError.next(true);
            }
        });
    }

    logout() {
        this.http.post('//localhost:8000/api/auth/logout', {}).subscribe( () => {
            localStorage.removeItem('user');
            localStorage.removeItem('token');
            this.currentUserSubject.next(null);
            this.router.navigate(['/']);
            console.log("LOGOUT");

        }, logoutError => {
            console.log(logoutError);
        });
    }
```

Figura 34: Servicio que controla los procesos de autenticación desde el lado del cliente

Por otro lado, la función *login* recibe como respuesta en formato JSON por parte del servidor, donde se incluye el id del usuario que inicia sesión y el token de acceso. Cuando recibimos esta información, guardamos ambos datos en el *localStorage* del navegador del cliente, de este modo la sesión permanecerá iniciada a menos que el usuario explícitamente decida cerrar sesión.

Finalmente, la función *logout* simplemente elimina las dos variables almacenadas del navegador y redirige al usuario a la página de inicio.

Anexo D: Currículum Vitae

Mi nombre es Víctor Ballester Quiles y actualmente resido en la localidad de Sant Joan d'Alacant, en la provincia de Alicante.

Los estudios de Grado Multimedia son los segundos estudios universitarios a los que accedo, después de acabar un tanto desencantado con la ingeniería de telecomunicaciones que comencé a estudiar al acabar el instituto y que decidí abandonar en el último curso.

Descubrí mi pasión por el desarrollo web a través de un curso ofrecido por el servicio de empleo de mi comunidad y decidí matricularme en la UOC en el grado multimedia, dado que siempre he tenido inquietudes por otras ramas como el sonido y la fotografía.

Nunca he tenido la posibilidad de trabajar en ninguna empresa relacionada con el desarrollo web, por lo que espero que estos estudios me permitan mejorar en el ámbito profesional y convertir una afición en un modo de vida.

Datos personales	**Víctor Ballester Quiles** 20-07-1986 (33 años) Carrer del Mercat, 32 03550 Sant Joan d'Alacant 677 180 516 vballesterq@gmail.com
Formación	**Grado Multimedia** 2015 – 2020, Universitat Oberta de Catalunya **FP Grado Superior de Sonido** 2006 – 2008 IES Luis García Berlanga
Formación complementaria	**Curso de desarrollo web (630 horas)** 2014 – 2015, IG Formación Alicante **First Certificate in English** 2009
Experiencia laboral	**Prácticas como desarrollador web en 3DIDS.com (300 horas)** Febrero 2019 – Mayo 2019 Trabajo con HTML, CSS y JavaScript en base a proyectos desarrollados con distintos gestores de contenido (Shopify, Prestashop y Wordpress). Iniciación a distintas herramientas de marketing digital (Google Analytics y Google Tag Manager). Desarrollo en HTML, CSS y JavaScript de la landing page www.agripinopeluquero.com.
Idiomas **Conocimientos**	**Inglés:** medio oral y escrito - **Valenciano:** nativo - **Castellano:** nativo Conocimientos básicos de programación front-end (HTML, CSS, Javascript, Angular, Bootstrap) Conocimientos básicos de programación back-end (PHP, MySQL, Java, Laravel)
Otros datos de interés	Carné de conducir tipo B y vehículo propio. Actualmente trabajando con una jornada de 30h/sem. Disponibilidad horaria parcial de mañanas con posibilidad de jornada completa.