

Título y portada chula

Falgueras Casajeros, Yago and Vidal Villalba, Pedro

6 de abril de 2025

Índice general

1. Documentación inicial	5
1.1. Introducción	5
1.2. Inventario de contenido	6
1.3. Arquitectura de información	6
1.4. Mapa de navegación	9
1.5. Interfaces	10
1.5.1. Prototipo manual	10
1.5.2. Esqueleto digital	16
1.5.3. Diseño final	20
1.6. Storyboard	24
1.7. Estructura de ficheros	25
2. Documentación HTML	27
2.1. Página principal	28
2.2. Descubre	29
2.3. Planifica	31
2.4. Reserva	33
2.5. About Us	35
3. Documentación CSS	37
3.1. Flexible Grids	37
3.2. CSS Multicol	40
3.3. Flex Container	44
3.4. CSS Grid	49
3.5. Bootstrap	53
3.6. Otros elementos de CSS	56
4. Documentación JavaScript	59
4.1. Inclusión de JavaScript	59
4.1.1. Acceso al DOM	59
4.1.2. Respuesta a eventos	63
4.2. Uso de jQuery y JES6	67
4.3. Carga de contenido	69
4.3.1. Formato XML	69
4.3.2. Formato JSON	71
5. Anexo	75
5.1. Tormenta de ideas	75
5.2. Cambios	76
5.2.1. Diseño Página principal	76
5.2.2. Diseño Descubre	77

5.2.3. Diseño Planifica	77
5.2.4. Diseño Reserva	78
5.2.5. Diseño About us	79
5.3. Estructura de ficheros	80

Capítulo 1

Documentación inicial

1.1. Introducción

El Problema del Viajante es una agencia de viajes ficticia en la que se ofrecerán todo tipo de servicios relacionados con la planificación y la gestión de viajes, incluyendo transporte, alojamiento, lugares de interés para visitar, ocio, restauración, etc.

El proyecto puede encontrarse en https://github.com/PedroVidalVillalba/DAW_P1, donde se encuentra también la última versión estable de este mismo documento y de la página web.

Se han utilizado como inspiración diferentes páginas web de reserva de viajes, en especial Experience travel group, aunque también se han consultado:

- Booking
- Tripadvisor
- Skyscanner
- Google Calendar

En estas páginas se puede comprobar que es habitual ofrecer una experiencia lo más completa posible respecto a la organización desde el mismo lugar el transporte y alojamiento. Pero hemos echado en falta que ninguna tenga aspectos más relacionados con la planificación en el propio viaje, como lugares de interés o restaurantes, y que incluso se pueda acceder a la reserva desde la propia página, de forma que se cree un itinerario. Además, ya no sería necesario acceder a otro lugar para indicar en el calendario las fechas indicadas, sino que ya se mostrarían en la propia página, y este calendario podría sincronizarse con el de la cuenta de Google, u otros semejantes.

También es importante destacar que el objetivo principal de la página no sería realmente obtener beneficios vendiendo viajes por todo el mundo, sino que serviría como una interfaz desde la que poder consultar gran cantidad de destinos y obtener información sobre cada uno ellos, todo en un mismo lugar. Además, si se llegase a desplegar la aplicación de forma realista, sería necesario comunicarse con otras páginas web, a las que se redirigía al usuario para que realice su reserva del viaje.

No obstante, sí que sería interesante ofrecer viajes personalizados dentro la aplicación, a un precio más alto de lo habitual pero que ofreciesen todo un itinerario planificado previamente por los trabajadores de la empresa ficticia. De esta forma, las personas que no quieran organizar los lugares de visita de su viaje, pueden acceder a esta característica.

1.2. Inventario de contenido

Inicialmente se ha realizado una tormenta de ideas, en la que se ha reflexionado sobre el tema escogido, y su contenido, audiencia, elementos visuales atractivos para el usuario, etc. Los contenidos iniciales de la tormenta de ideas se pueden consultar en el Anexo. Los elementos principales sobre los que se estructura la aplicación son:

- Reserva de hoteles, aviones y otros medios de transporte. En general, cómo ir desde el lugar donde vive el usuario hasta su destino.
- Lugares de interés para visitar, con fotos, su web, ubicación, etc.
- Lugares de ocio, restauración, y otro tipo de tiendas conocidas en la zona.
- Calendario con planificación del viaje, desde el que acceder a la información sobre el medio de transporte elegido, alojamiento, etc. y que permita organizar los lugares de visita de cada día.
- Lista de lugares y tiendas favoritos, junto a un mapa integrado en donde aparezcan destacados.
- Experiencias exclusivas, con planificaciones de viajes realizadas previamente por expertos.

Después de analizar los elementos obtenidos, filtrarlos y establecer relaciones entre sí, se ha llegado al siguiente inventario de contenido:

—Poner aquí el inventario de contenido. Hay que hacer un dibujín —

Otros elementos que también son relevantes para la justificación de la creación de la página web son:

1. **Audiencia:** un público genérico, ya que cada vez más personas viajan de forma relativamente habitual, y resulta de gran utilidad una página web desde la que poder gestionar todo el viaje.
2. **Elementos visuales:** principalmente imágenes y vídeos de cada lugar, que deben llamar la atención del usuario sin ser excesivos, y también mapas. Se creará un logo generado con IA que sea característico de la empresa.
3. **Financiación:** se podrían obtener beneficios a través de anuncios en la página. También se podría incluir un sistema de suscripción o mecenazgo con descuentos dentro de la página web, para clientes habituales.

1.3. Arquitectura de información

Una vez se dispone del inventario de contenido, se organizan estos elementos jerárquicamente en una arquitectura de la información, que dará una primera aproximación al número de páginas que se necesitarán crear y al esquema de navegación que permite relacionarlas.

Se llega así, a alto nivel, a la siguiente estructura, en la que cada elemento principal tiene varias ramas:

- **Página principal:** presentación de la página que llame la atención, *newsletter*, valoraciones, opiniones de expertos.

- **Descubre:** búsqueda de lugares visita, ocio, restauración etc., destinos de moda/recomendados, exploración, mapa de lugares favoritos.
 - **Planifica:** calendario, consejos generales para viajes al extranjero, seguro de viajes, mecanismos de reserva con seguridad.
 - **Reserva:** búsqueda de hoteles, búsqueda de aviones y otros medios de transporte, alquiler de coches, reserva de restaurantes, experiencias exclusivas.
 - **About us:** quiénes somos, nuestra historia, qué nos diferencia, FAQ, redes sociales, nuestro sistema de puntuación, certificados de calidad, premios.
 - **Otros:** términos de uso, política de cookies, política de privacidad.

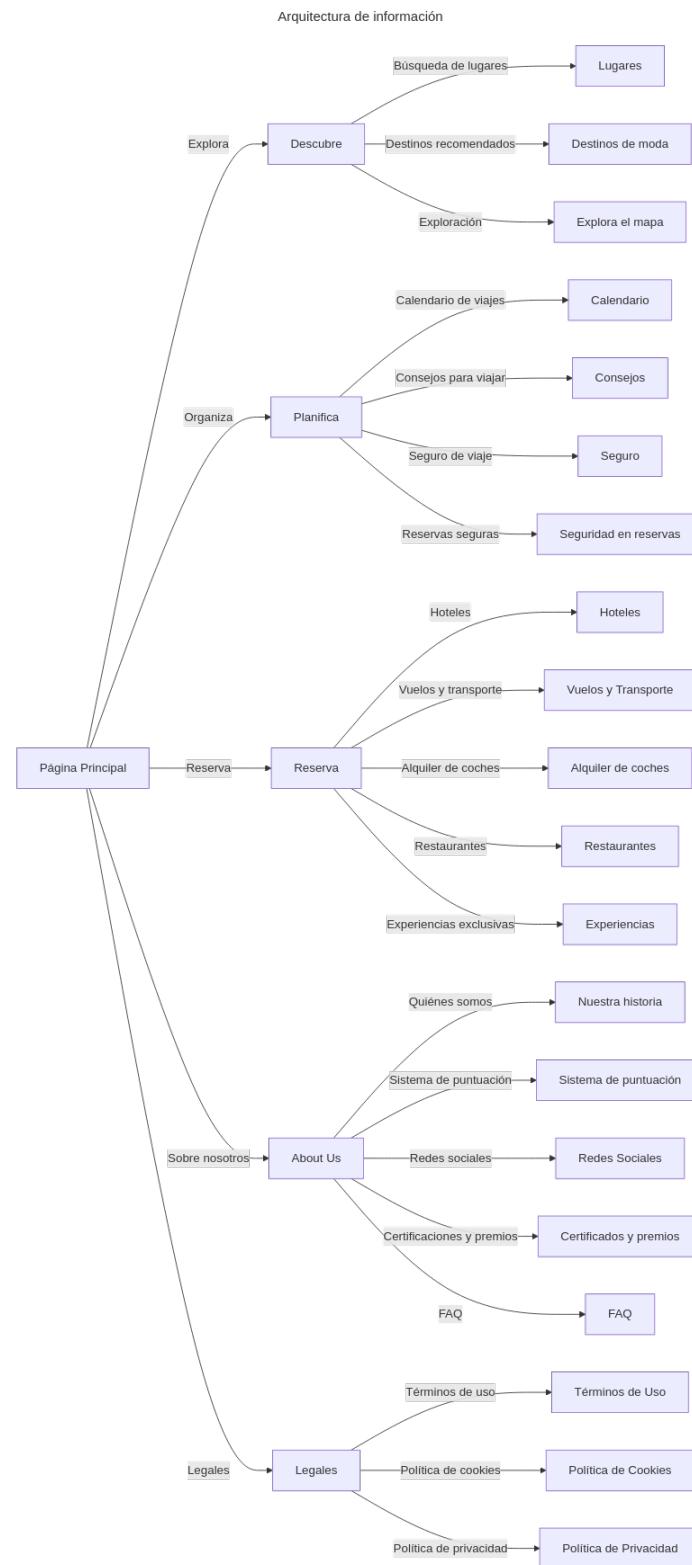


Figura 1.1: Arquitectura de información

1.4. Mapa de navegación

El mapa de navegación plasma las relaciones jerárquicas recogidas por la arquitectura de la información y las organiza en páginas concretas que serán posteriormente implementables. Como se ve en la imagen, es similar a la arquitectura de información

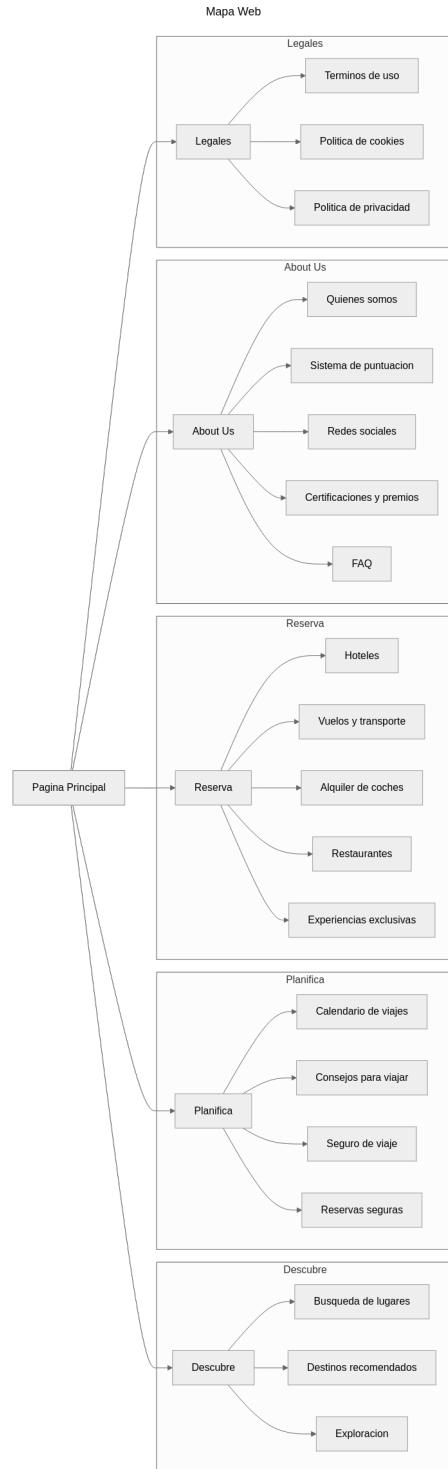


Figura 1.2: Mapa de navegación

1.5. Interfaces

Con toda la información recogida, el siguiente paso es empezar a diseñar las interfaces, en las que se muestre un prototipo de la disposición, fuentes de texto, colores, imágenes y navegación en las diferentes páginas web. En este caso, se han realizado interfaces de las 5 páginas principales: la página principal, descubre, planifica, reserva y *about us*.

Cada una incluye tres fases de desarrollo, una prototipo manual(*sketch*) con las ideas básicas que conforman el sitio web, un esqueleto digital (*wireframe*) centrado en la disposición de los elementos de acuerdo a proporciones adecuadas, y por último un diseño final (*mockup*) de la interfaz, que muestra cómo debería ser la apariencia final de la página, incluyendo colores, fuentes de texto e imágenes adecuadas.

1.5.1. Prototipo manual

A continuación se muestran los bocetos de las interfaces realizadas a mano. Dado el carácter académico de este documento, se considera relevante incluirlas, a pesar de que en un entorno profesional su uso sería poco común. Se han realizado los prototipos para cinco ventanas, las cuales se comentan a continuación.

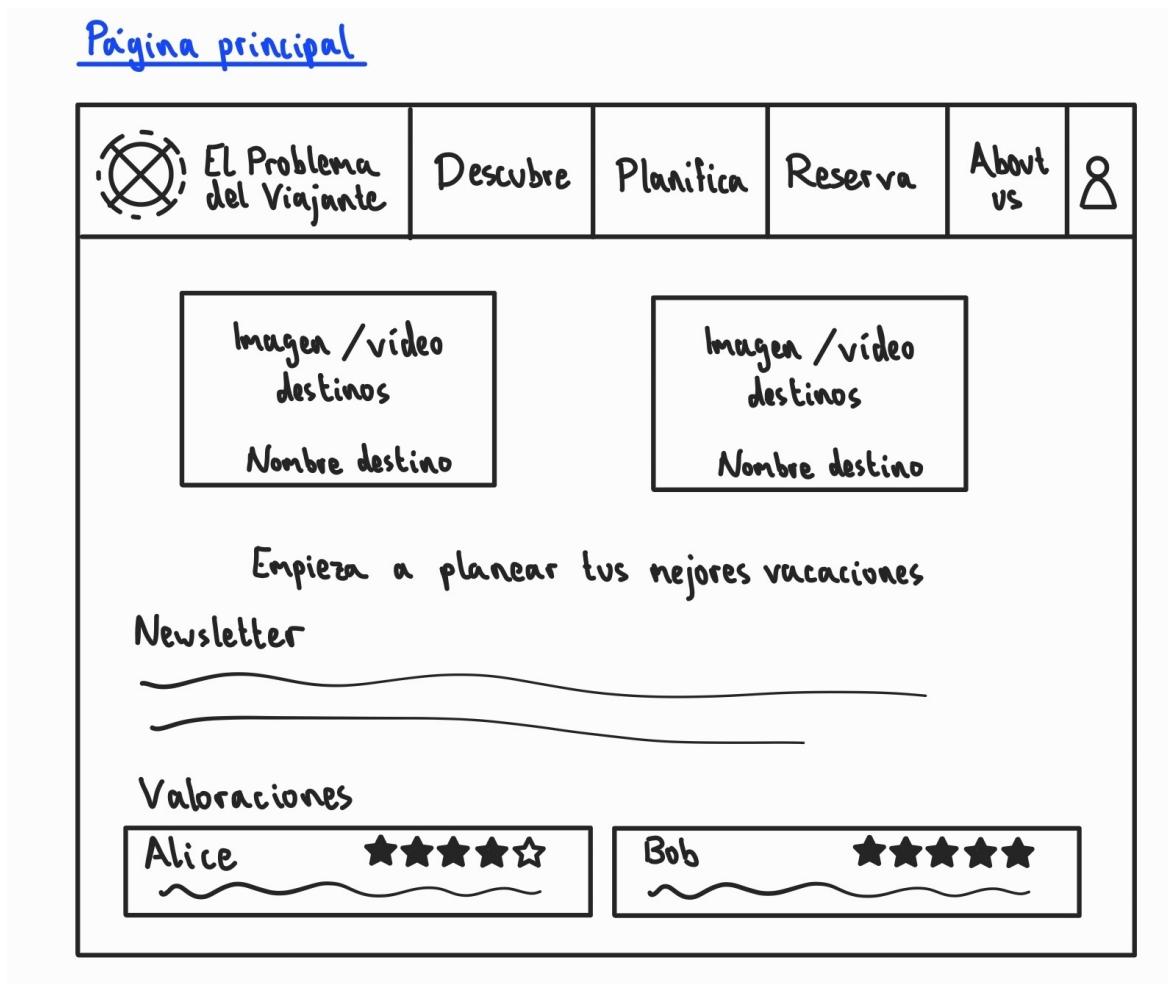


Figura 1.3: Prototipo Página principal

En esta figura se puede ver la **página principal** del sitio web. Aquí, se ve la barra de navegación global, que será común a todas las páginas, unos cuantos elementos visuales sobre algunos destinos destacados para captar la atención del visitante, un apartado de *newsletter* con novedades importantes y una sección con valoraciones de clientes.

Desde la barra de navegación, el usuario podrá registrarse por medio de correo electrónico para poder realizar valoraciones, que se guarden sus viajes, crear una planificación colaborativa y obtener mejores recomendaciones.

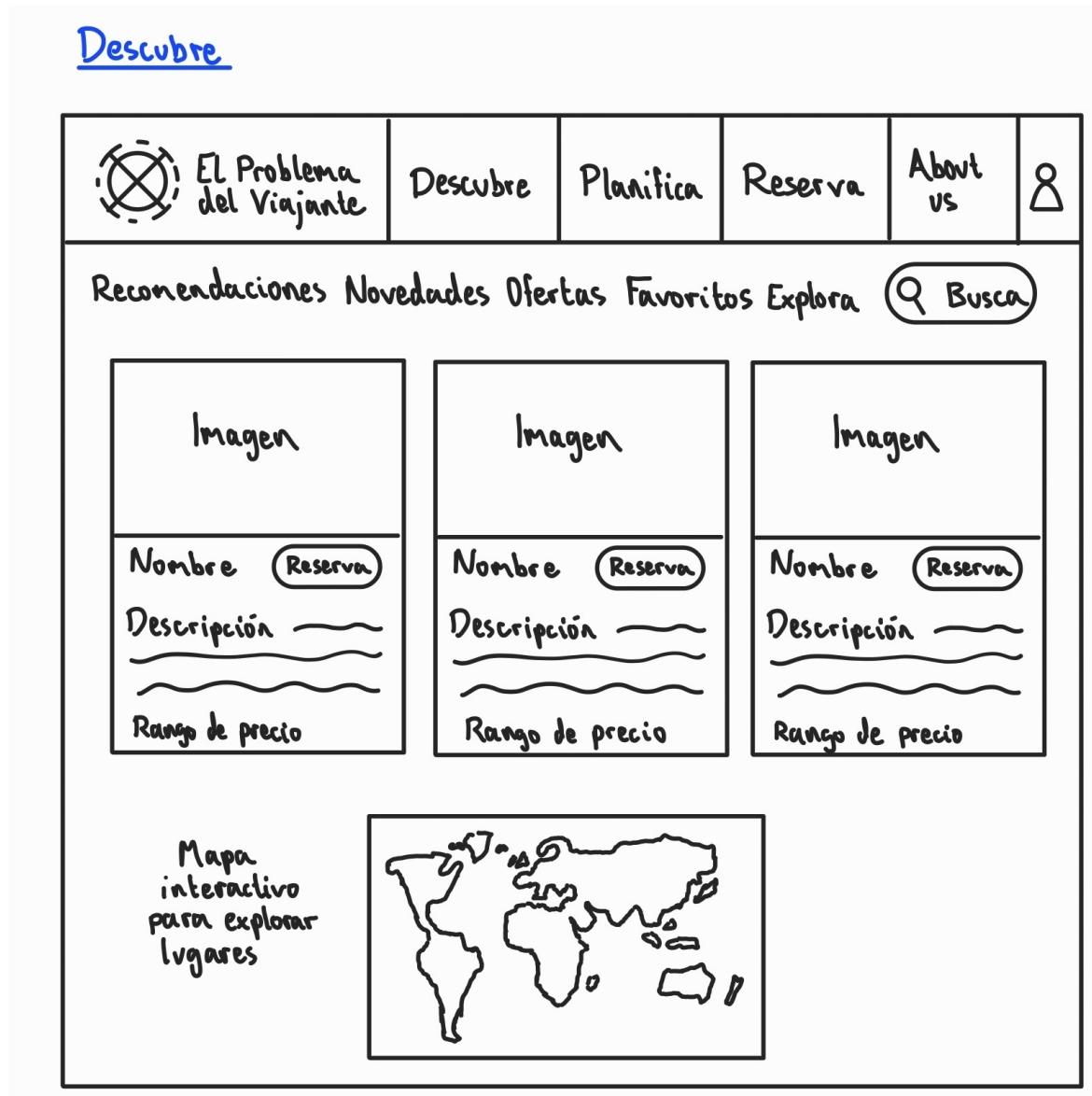


Figura 1.4: Prototipo Descubre

En el apartado de **Descubre** se pretende que los clientes de la página que no tienen todavía claro cuál es el destino del viaje que quieren puedan explorar sus opciones.

Se ve, en primer lugar, una barra de navegación secundaria en la que aparecen diferentes subsecciones. En cada una de estas se presentarán destinos asociados al criterio empleado. En todas estas aparecerán diversos paquetes de viajes, mostrando imágenes y descripciones de los lugares a los que viajar. Asimismo, se incluye un mapa interactivo en el que aparecerán destacados los lugares mencionados en las tarjetas que aparecen en la página. Interactuando con este mapa, el cliente podrá, además, filtrar la búsqueda a un destino en concreto. Se incluyen además algunos destinos recomendados, consejos generales para la planificación de viajes y un apartado sobre la seguridad en las reservas.

Planifica

El Problema del Viajante	Descubre	Planifica	Reserva	About us																																				
<p>Invita <</p> <p>Viajeros</p> <ul style="list-style-type: none"> • Alice • Bob • Charlie 	<p>Agenda</p> <p>< Febrero 2025 ></p> <table border="1"> <thead> <tr> <th>LUN</th> <th>MAR</th> <th>MIE</th> <th>JUE</th> <th>VIE</th> <th>SÁB</th> <th>DOM</th> </tr> </thead> <tbody> <tr> <td>3</td> <td>4</td> <td>5</td> <td>6</td> <td>7</td> <td>8</td> <td>9</td> </tr> <tr> <td>10</td> <td>11</td> <td>12</td> <td>13</td> <td>14</td> <td>15</td> <td>16</td> </tr> <tr> <td>17</td> <td>18</td> <td>19</td> <td>20</td> <td>21</td> <td>22</td> <td>23</td> </tr> <tr> <td>24</td> <td>25</td> <td>26</td> <td>27</td> <td>28</td> <td></td> <td></td> </tr> </tbody> </table> <p>Fechas disponibles</p>					LUN	MAR	MIE	JUE	VIE	SÁB	DOM	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28		
LUN	MAR	MIE	JUE	VIE	SÁB	DOM																																		
3	4	5	6	7	8	9																																		
10	11	12	13	14	15	16																																		
17	18	19	20	21	22	23																																		
24	25	26	27	28																																				
<p>Destinos recomendados</p> <div style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p>Imagen</p> <p>Nombre <input type="button" value="Reservar"/></p> <p>Descripción</p> <p>Rango de precio</p> </div> <div style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p>Imagen</p> <p>Nombre <input type="button" value="Reservar"/></p> <p>Descripción</p> <p>Rango de precio</p> </div> <p>Consejos generales</p> <p>Reserva con seguridad</p>																																								

Figura 1.5: Prototipo Planifica

En la sección de **Planifica**, el cliente podrá planificar su viaje con, posiblemente, otros usuarios. Así, están disponibles una columna en la que se puede invitar a otros viajeros y un calendario en el que cada uno de los participantes en un viaje puede marcar las fechas que tiene disponibles para una planificación más eficiente.

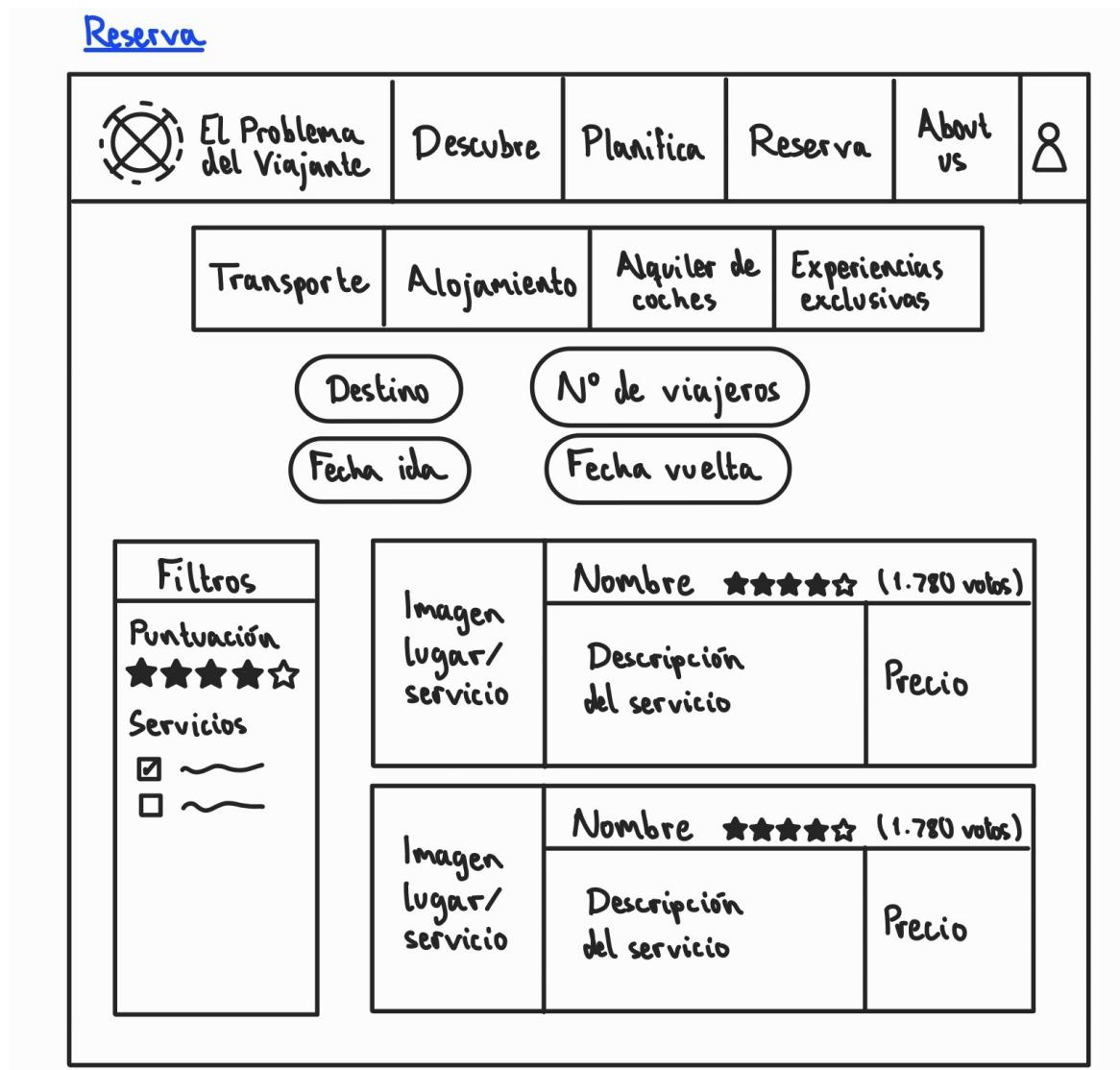


Figura 1.6: Prototipo Reserva

En la página de **Reserva** el usuario podrá gestionar las reservas necesarias para su viaje. Estas incluyen el transporte hasta el lugar, alojamiento, alquiler de vehículos y otras experiencias exclusivas facilitadas por nuestro sitio web.

Se incluye un filtro principal por destino, número de viajeros y fechas de ida y vuelta. Hay otros filtros secundarios que se podrán aplicar sobre la puntuación y el tipo de servicio. Una vez seleccionados estos campos, aparecerán los servicios recomendados por la web, con una imagen, su nombre, descripción y precio para que el usuario pueda gestionar su reserva.

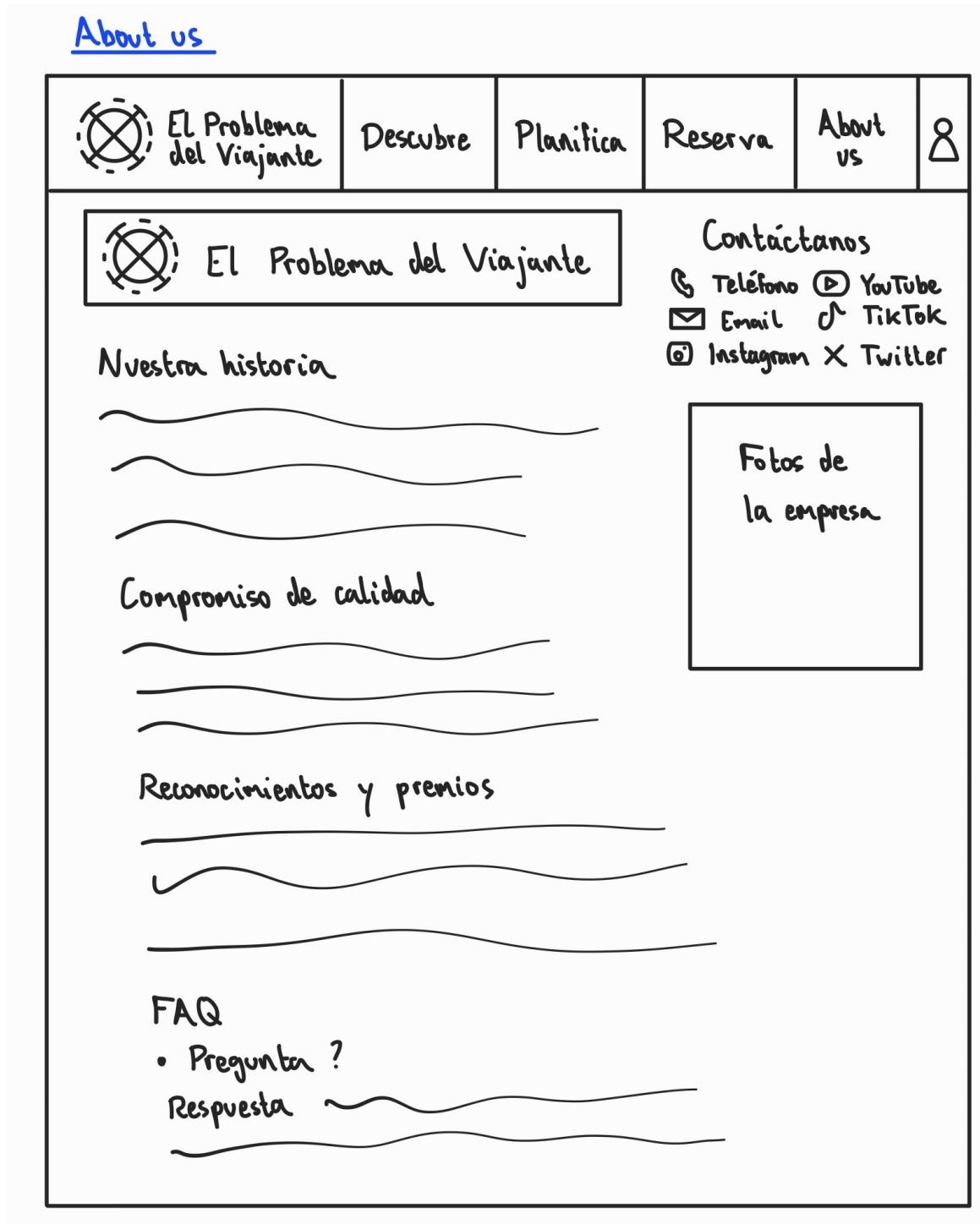


Figura 1.7: Prototipo Página principal

La página de **About us** es principalmente de texto con información sobre la empresa ficticia. Hay información de contacto, la historia de la empresa, reconocimientos y premios, indicadores de calidad y preguntas frecuentes (FAQ).

1.5.2. Esqueleto digital

El *wireframe* ayuda a establecer las proporciones entre los elementos de la página y a establecer la jerarquía de información del documento en términos de tamaño y posición. Para hacer el diseño, se ha utilizado el sitio web Figma está basado en la regla de las 12 columnas, en las que cada una de las páginas se divide verticalmente en 12 partes, el cual es un número suficientemente pequeño para ser fácilmente manejable (a diferencia de trabajar, por ejemplo, empleando píxeles) pero a la vez es lo suficientemente grande y tiene bastantes divisores como para permitir flexibilidad en el diseño. En los diseños que se muestran a continuación se marcan el número de columnas que ocupan los elementos destacados.

Cabe destacar que para crear un diagrama con un tamaño razonable, todos los diagramas empleados se crearon en un tamaño equivalente a la resolución HD estándar (1920x1080 píxeles). Sin embargo, la expectativa de cualquier usuario de un sitio web es que las páginas admitan siempre desplazamiento en vertical para visualizar toda la información de la página. El desplazamiento en horizontal, si bien posible y utilizado, es menos habitual. Por este motivo, se representa el número de columnas pero no el de filas, y muchos de los elementos aparecen artificialmente distorsionados en la horizontal para poder entrar en el tamaño fijado, lo cual no se mantendrá en la página real.

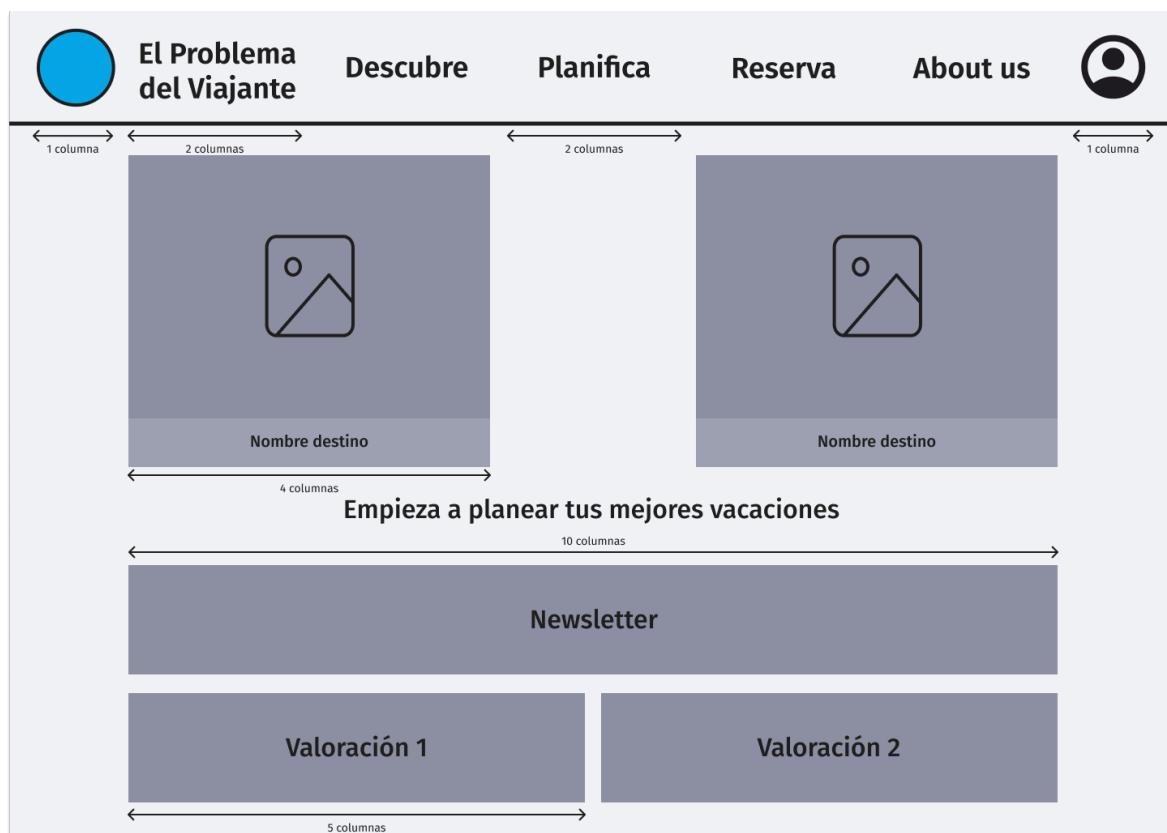


Figura 1.8: Wireframe Página principal

Como se ha comentado, se muestran las columnas que ocupa cada elemento. Cabe destacar que la versión final para un dispositivo móvil diferirá significativamente respecto a la imagen mostrada, ya que habrá que tener en cuenta su orientación vertical, por lo que no se mostrarán diseños a dos columnas como este. Por otro lado, el *newsletter* y las valoraciones no se verían completamente al ingresar en la página web, sino que habría que desplazarse verticalmente para visualizar su contenido.

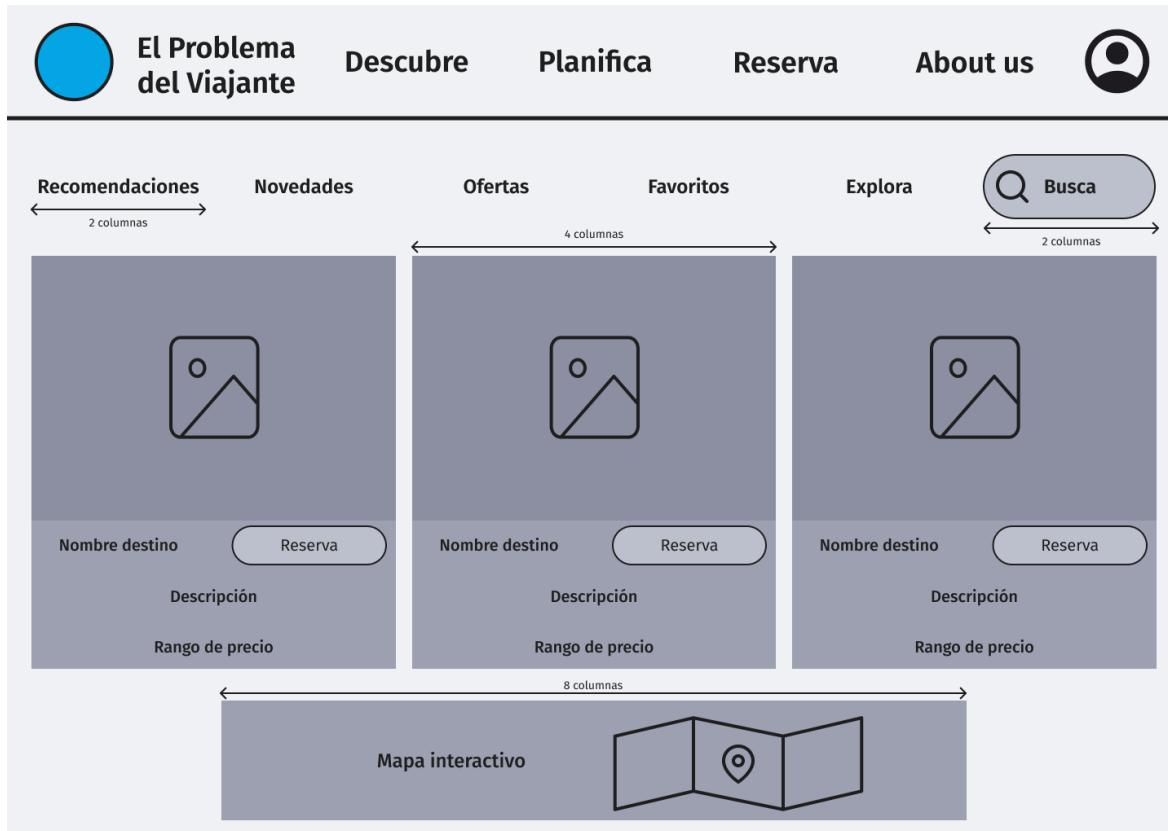


Figura 1.9: Wireframe Descubre

En este caso se opta por un diseño a tres columnas, siguiendo la regla de los tercios, en el que las imágenes aparecerían en la parte central de pantalla, y habría una barra de navegación secundario para elegir diferentes formas de búsqueda. De nuevo, en la versión móvil, el diseño sería distinto.

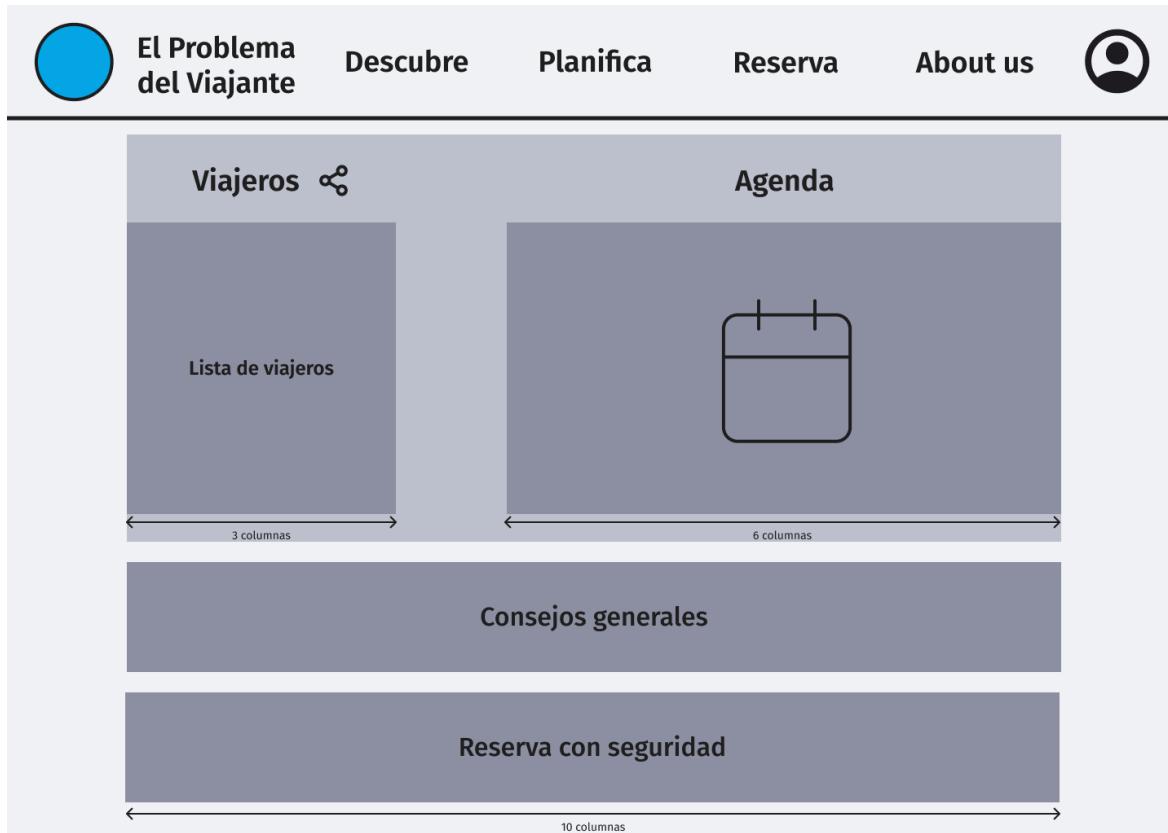


Figura 1.10: Wireframe Planifica

En esta pestaña se muestra un calendario compartido por todos los viajeros que estén apuntados a un cierto viaje, junto a las fechas de salida y llegada, los lugares de visita reservados, etc. Más abajo, se encuentran consejos para los viajes, los cuales no se verían completamente en la forma final, ya que ocuparían más espacio del que aparece aquí.

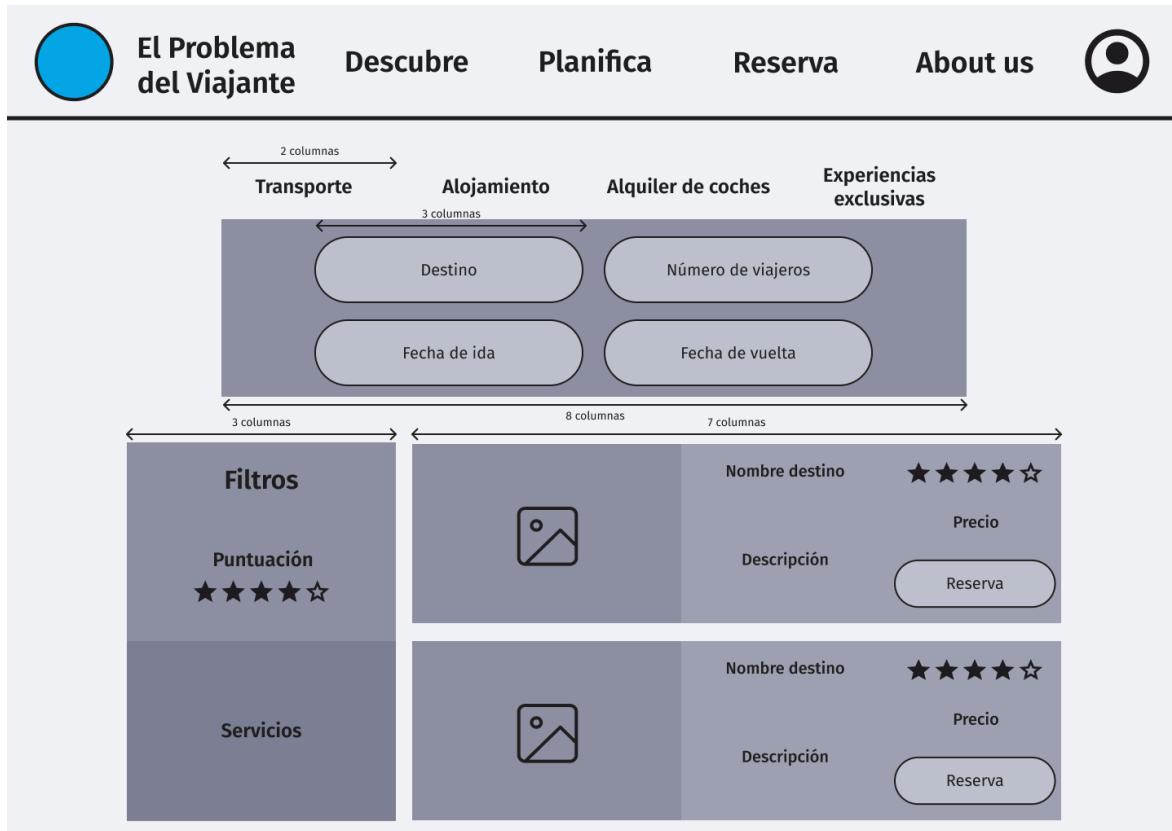


Figura 1.11: Wireframe Reserva

En este wireframe se muestra otra barra de navegación secundaria diferente, con las diferentes modalidades de reserva, y múltiples opciones de personalización del viaje, desde la fecha hasta la puntuación y los servicios ofertados. El diseño está inspirado en el de las páginas web de reservas de viajes como Tripadvisor.

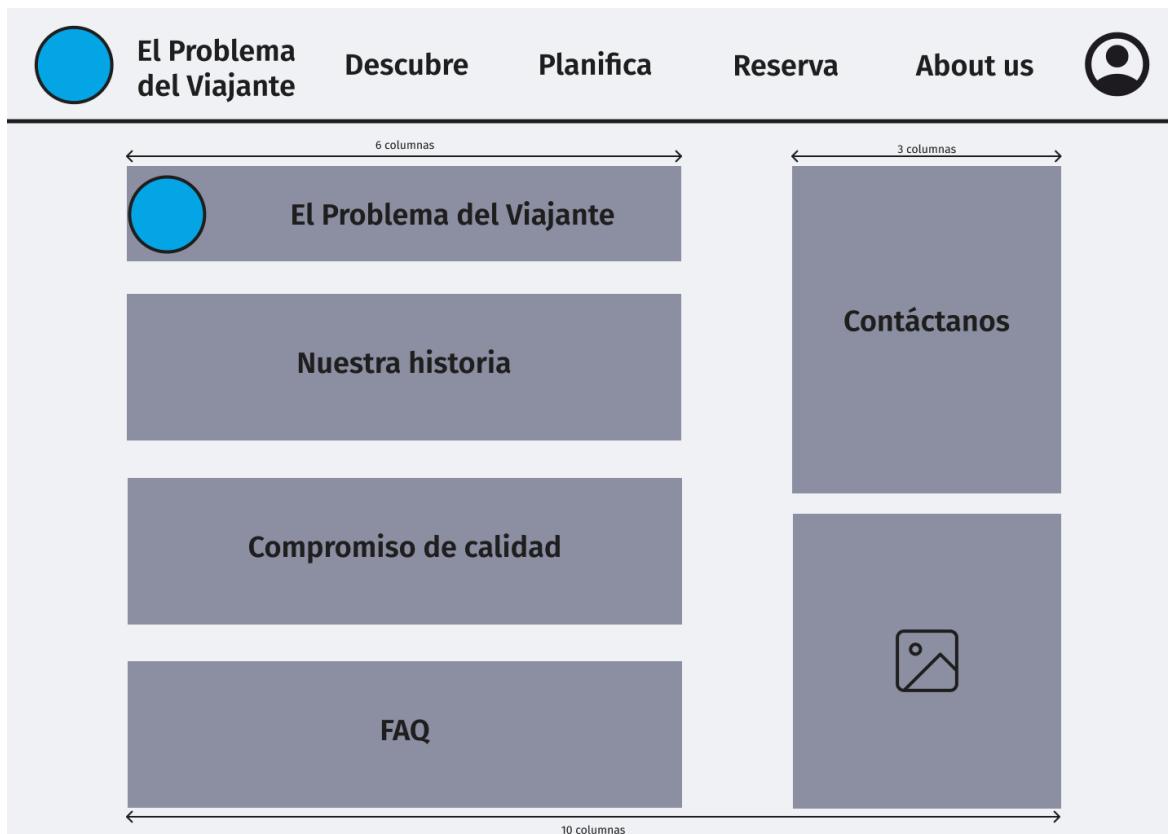


Figura 1.12: Wireframe About us

Se opta por un diseño con una columna principal con los elementos más relevantes como la historia de la empresa, los indicadores de calidad y las preguntas frecuentes, y una columna secundario con las redes sociales y una imagen de los integrantes de la empresa.

1.5.3. Diseño final

El uso de *mockups* permite ver la apariencia final de la página, incluyendo colores, fuentes de texto, logos e imágenes que acompañan al diseño y que hasta este momento se habían obviado. En cuanto la paleta de colores, se optó por usar una paleta ya creada y disponible de forma abierta: Catppuccin Latte, a la que se le han añadido algunos colores propios, reutilizados de otros proyectos.

Como tipografía elegida se utilizará Helvetica. Debido a que no está disponible en la herramienta utilizada para el diseño final, se ha usado una letra similar, Fira Sans.

No se añade una descripción a cada una de las imágenes, como se ha hecho en los dos apartados anteriores, ya que muestran la misma disposición del contenido que los *wireframes*, añadiendo los respectivos colores y formateos de texto.

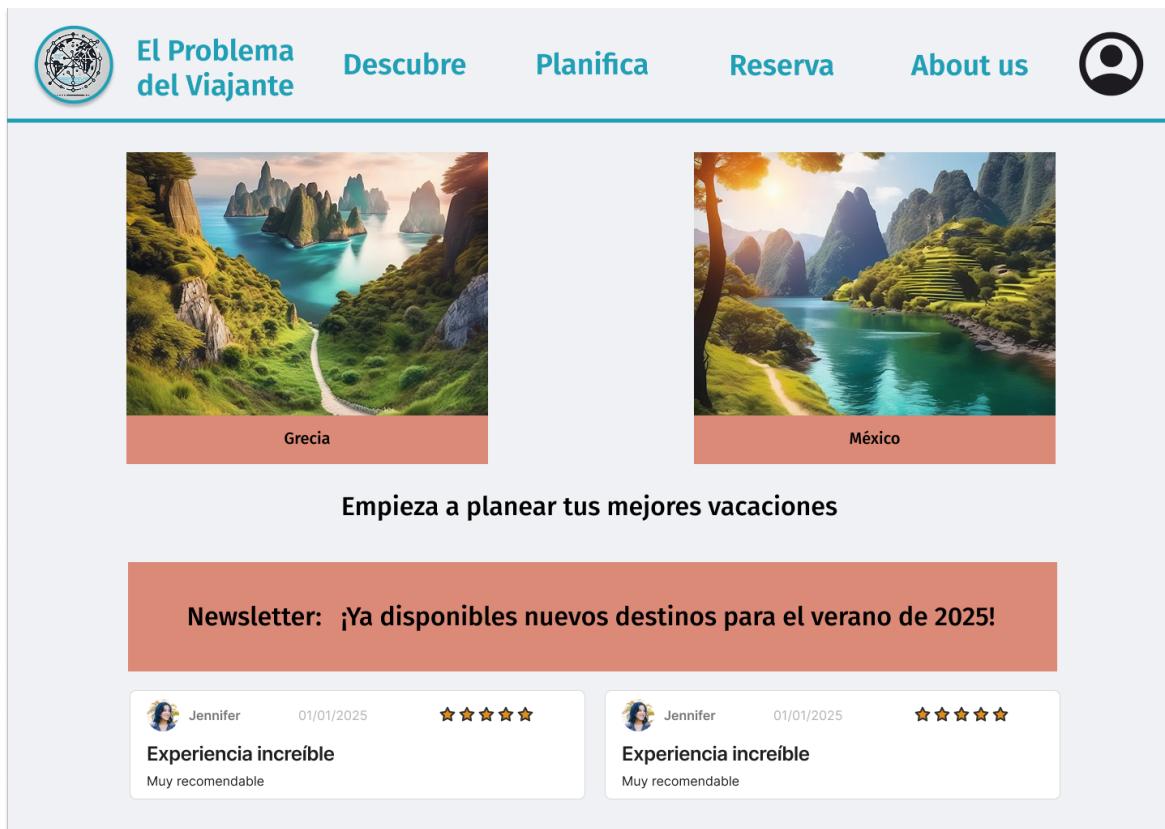


Figura 1.13: Mockup Página principal

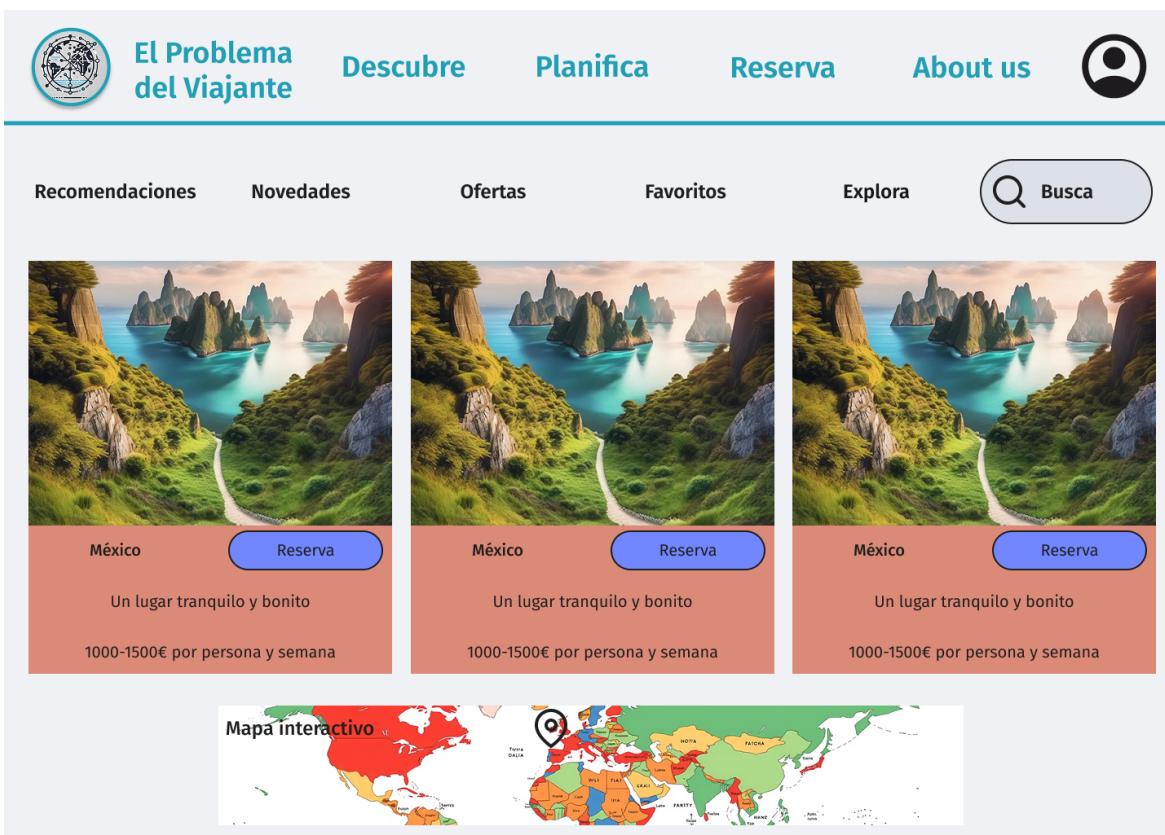


Figura 1.14: Mockup Descubre

The mockup shows the 'Planifica' tab selected in the top navigation bar. Below it, there's a 'Viajeros' section with two users: Alice and Bob. A date input field shows '17/08/2025'. To the right is a calendar for August 2025, with the 17th highlighted. Below the calendar is a 'Consejos generales' (General tips) section containing placeholder text. Further down is a 'Reserva con seguridad' (Secure booking) section with similar placeholder text.

Figura 1.15: Mockup Planifica

The mockup shows the 'Reserva' tab selected in the top navigation bar. It features sections for 'Transporte', 'Alojamiento', 'Alquiler de coches', and 'Experiencias exclusivas'. Under 'Alojamiento', there are fields for 'Destino', 'Número de viajeros', 'Fecha de ida', and 'Fecha de vuelta', along with a 'Busca' button. On the left, there's a 'Filtros' panel with 'Puntuación' (5 stars) and 'Servicios' (checkboxes for Desayuno, Piscina, and Aire acondicionado). Two hotel cards are shown: 'Hotel La Montaña' (5 stars, from 150€, Reserva button) and another 'Hotel La Montaña' (5 stars, from 150€, Reserva button).

Figura 1.16: Mockup Reserva

The mockup displays the website's header with navigation links: 'El Problema del Viajante', 'Descubre', 'Planifica', 'Reserva', 'About us', and a user icon. Below the header, there's a circular profile picture placeholder. The main content area features several sections: 'El Problema del Viajante' (with a small globe icon), 'Nuestra historia' (with a brief description), 'Compromiso de calidad' (with a statement about quality), and 'FAQ' (with a list of questions and answers). To the right, a blue-bordered box titled 'Contáctanos' lists contact information for various platforms: Teléfono, Email, Instagram, YouTube, TikTok, and Twitter (X). A photograph of two men standing outdoors at night, holding coffee cups, is positioned next to the contact section.

El Problema del Viajante

Nuestra historia

Esta empresa comenzó como un simple trabajo para una asignatura de la carrera y se convirtió en ...

Compromiso de calidad

Desde un primer momento, lo más importante es garantizar la mejor experiencia para el usuario, desde la reserva hasta la estancia.

FAQ

- Pregunta
Respuesta
- Pregunta
Respuesta

Contáctanos

- Teléfono
- Email
- Instagram
- YouTube
- TikTok
- Twitter (X)

Figura 1.17: Mockup About us

1.6. Storyboard

El *storyboard* permite visualizar de forma dinámica el sitio web. Esto permite ilustrar cómo interactúan elementos y colores, cómo funciona el sistema de navegación propuesto y, en general, tener una idea de si el sitio web representa aquello para lo que fue diseñado.

Para ello, se añade información sobre los diseños finales del apartado anterior, donde se muestra cómo interactuar con los elementos de la página. A continuación se muestran tres casos de uso.

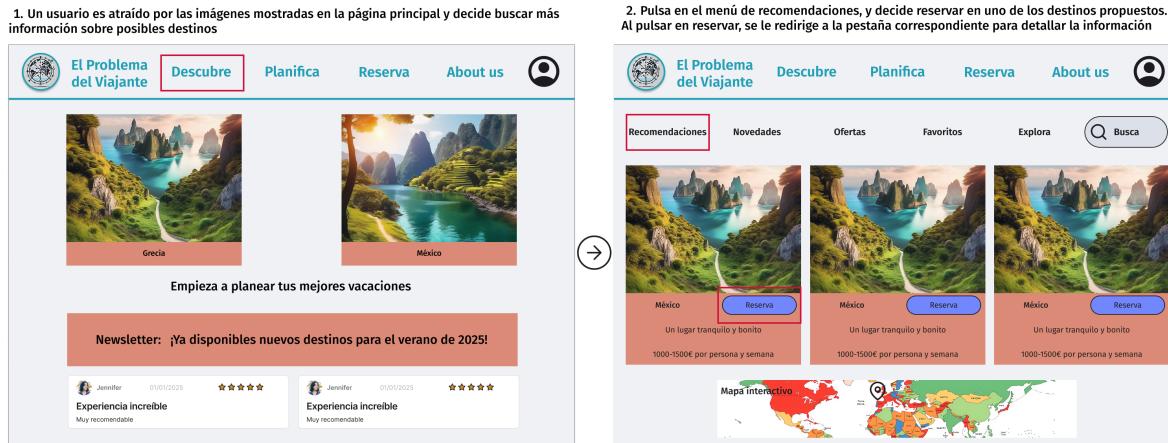


Figura 1.18: Storyboard Descubre

Supongamos que el usuario accede a la página web ya que tiene interés en buscar posibles lugares de visita. Lo intuitivo sería pulsar en la pestaña de **Descubre** del menú de navegación. En ella, hay diferentes opciones en la barra de navegación secundaria, donde la más relevante sea posiblemente **Recomendaciones**, y en esta se mostrarán los destinos propuestos. Al pasar el ratón por encima de uno de ellos, el mapa se situará en el lugar donde se encuentran, para facilitar buscar información sobre sus alrededores. Por último, si decidiese que está interesado en hacer la reserva, al pulsar en el correspondiente botón se le redirigiría a la página web desde la que puede reservar.



Figura 1.19: Storyboard Planifica

La imagen no se ve muy bien, igual hay que pasarlala a vertical

Supongamos ahora que este mismo usuario, tras reservar un viaje y crearse una cuenta en la página web, decide comprobar las fechas de su reserva. Para ello, se dirige a la pestaña de **Planifica**, desde la que puede añadir a las personas que viajarán con él, lo cual facilita la

organización del viaje, la elección de los lugares de visita y el reparto de gastos, además de asegurar que no habrá actividades que se superpongan.

Asimismo, pulsando en el calendario puede cambiar el mes en el que se encuentra y añadir la planificación de cada día según corresponda.



Figura 1.20: Storyboard Reserva

Para el último, pensemos en otro usuario que sí tiene claro el destino al que quiere ir, pero no dónde se va a alojar, por lo que accede a la pestaña de **Reserva**, desde donde podrá ver diferentes ofertas. En la barra de navegación secundaria, selecciona **Alojamiento** e introduce los datos necesarios, como la fecha ida y de vuelta, y el nombre del destino.

Al pulsar en buscar, se le mostrarán diferentes opciones, ordenadas según los filtro seleccionados, los cuales pueden modificarse para mostrar únicamente alojamientos con una puntuación superior a una dada, o que ofrezcan ciertos servicios como piscina o gimnasio. A continuación, si desea realizar la reserva mostrada, solamente tiene que pulsar en el correspondiente botón, el cual le redirigiría a la página web desde la que puede reservar.

1.7. Estructura de ficheros

Para la organización de los ficheros del proyecto, optamos por una estructura simple en directorios en la que el documento principal, 'index.html', está por sí solo en el nivel raíz, pues es lo que espera el servidor web.

Por otra parte, tenemos un directorio con la documentación en el que se incluye este documento y sus imágenes asociadas en su correspondiente subdirectorio, así como otros elementos de documentación que puedan surgir en el futuro.

Para los contenidos en sí de la página web, disponemos de cuatro subdirectorios. Uno está destinado a almacenar todas las imágenes que serán necesarias para la elaboración del sitio web. Por otra parte, disponemos de una carpeta en la que se almacenarán los documentos HTML que darán estructura a las diferentes páginas. En el directorio de estilos se almacenarán los ficheros de CSS necesarios para dotar al sitio web de una apariencia profesional y consistente. Por último en el directorio de scripts se incluirán los ficheros de JavaScript que doten al sitio web de funcionalidades dinámicas.

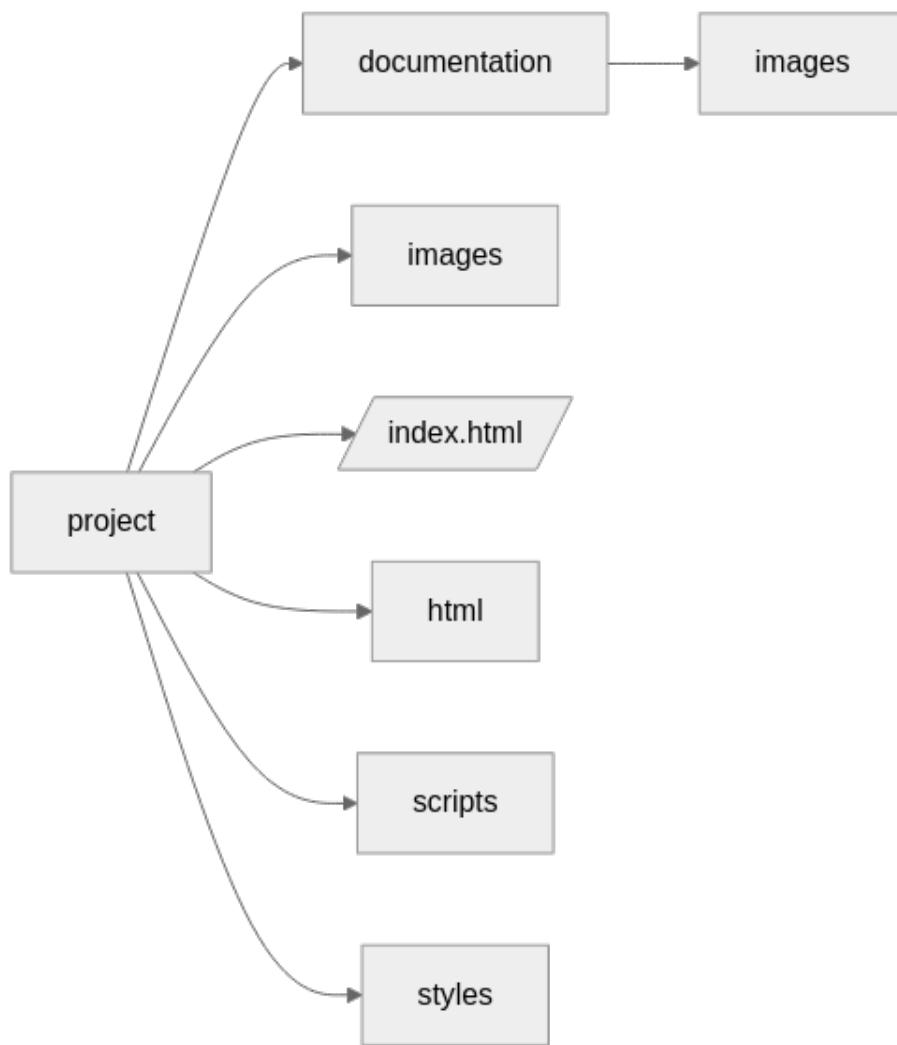


Figura 1.21: Diagrama de la estructura de ficheros

Capítulo 2

Documentación HTML

En la siguiente parte del proyecto, se empieza a codificar el código HTML de las páginas, centrándose en el uso adecuado de las etiquetas HTML, siguiendo los estándares de W3C, y de momento no se codifican características de visualización de la página, pues corresponden a la parte de hojas de estilo.

Para cada una de las cinco página principales, se muestra a continuación los nombres de los elementos más importantes y en algunos casos sus mapas de etiquetas. Como se ha comentado en clase, solo se incluyen los mapas de etiquetas de los elementos con mayor contenido y dificultad, obviando así muchos que están formados únicamente por un encabezado y varios párrafos.

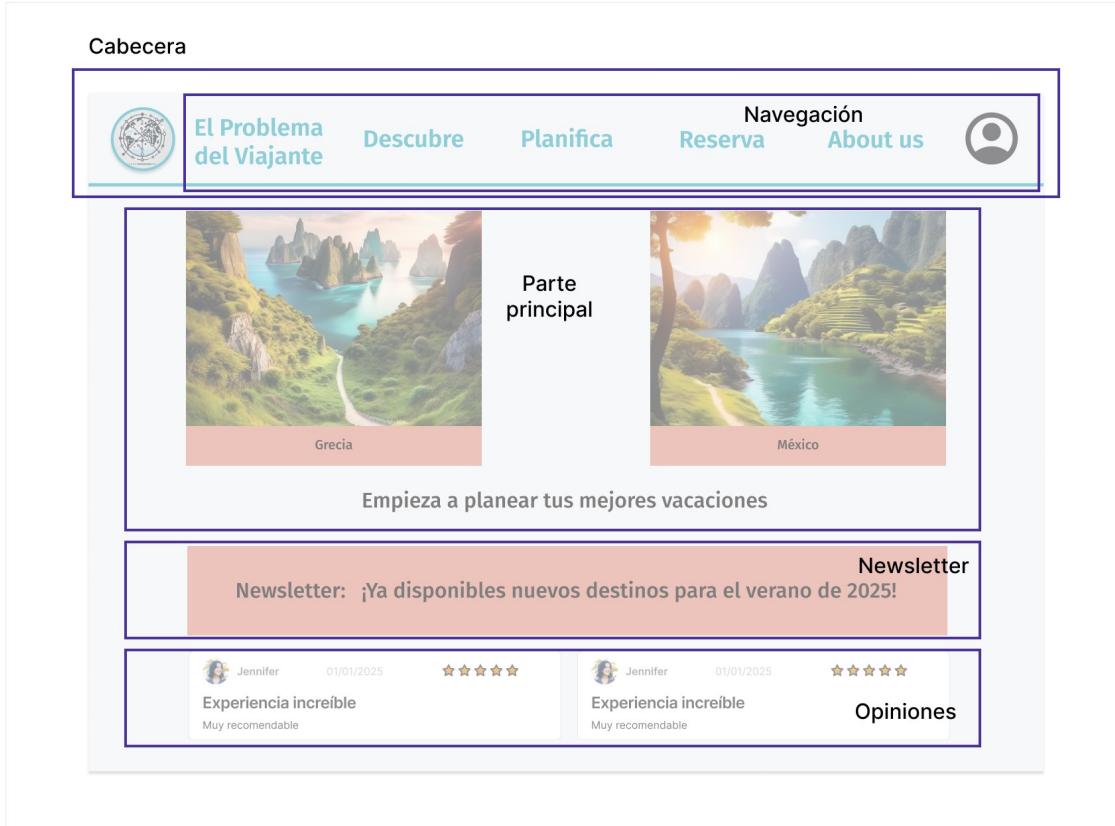


Figura 2.1: Elementos página principal

2.1. Página principal

En primer lugar, se va a comentar la barra de navegación, que es común a todos los elementos. Es sencilla, con el logo de la empresa, y con cinco apartados que nos redirigen precisamente a las cinco páginas principales de la web. Lo más relevante son las imágenes de la zona principal, ya que es lo primero que verá un usuario al ingresar en la página.

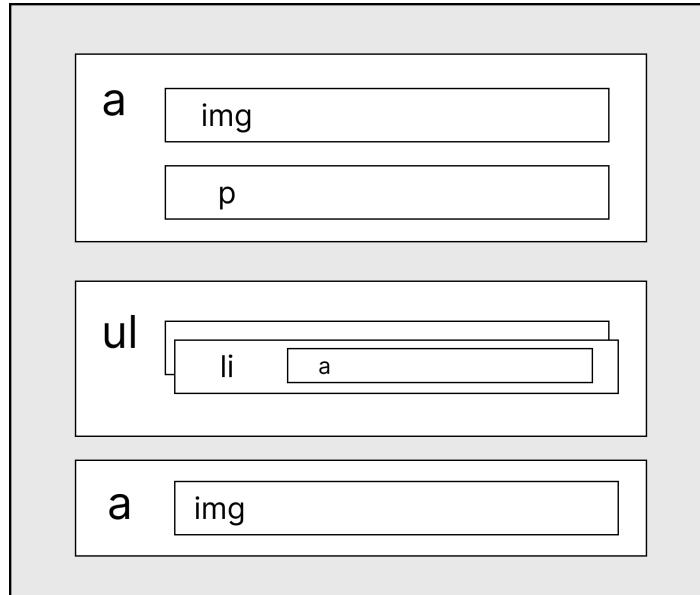


Figura 2.2: Mapa de etiquetas menú de navegación

En el mapa de etiquetas del menú de navegación se entra en más detalles, donde se puede ver que el logo junto al nombre de la empresa son un enlace que lleva a la página principal, la cual se acaba de mostrar. El resto son una lista, a la que se le eliminará el formato, de texto con un hiperenlace, a excepción del botón de *login*, que es de nuevo una imagen.

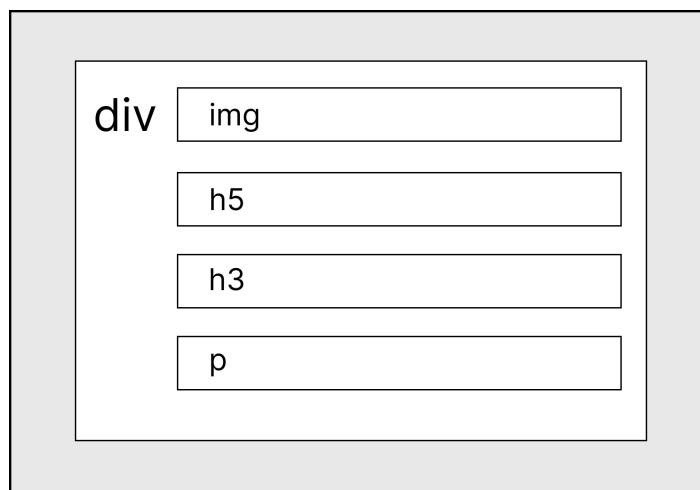


Figura 2.3: Mapa de etiquetas valoración

También se incluyen los elementos de una caja de valoración, pues es relevante cómo se construye. Se ha decidido utilizar un `div` para englobar todos los elementos y poder replicarla con facilidad manteniendo sus propiedades. De esta forma, está compuesta por una imagen,

correspondiente a la foto de perfil del usuario, junto a su nombre y puntuación en el encabezado **h5**. A continuación aparece el título de la reseña y su descripción.

2.2. Descubre

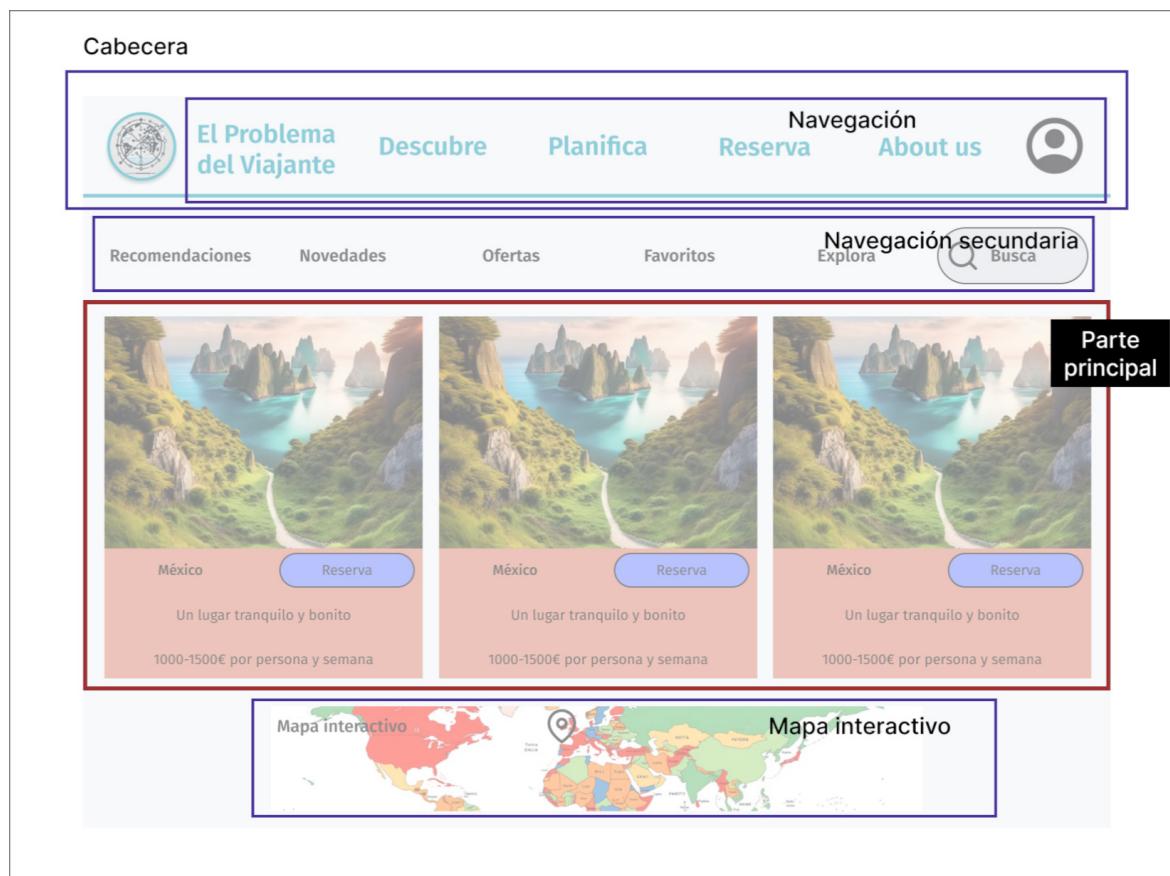


Figura 2.4: Elementos Descubre

En el apartado de descubrir destinos, la página se divide en una barra de navegación secundaria debajo de la navegación principal (esta común a todas las páginas), una lista de tarjetas de destinos con destinos recomendados y una breve descripción de cada uno y un mapa interactivo en el que aparecerán marcados los destinos asociados a las tarjetas y con el que se podrá interactuar directamente o a través de las tarjetas de la parte principal. Se entrará en detalles sobre la barra de navegación secundaria y la sección de las tarjetas de destino, pues la estructura HTML del mapa interactivo poco ilustrativa ya que el comportamiento y la apariencia se consiguen a través de la librería externa Leaflet [11], que hace uso de CSS y JavaScript para conseguir la mayor parte de su funcionalidad.

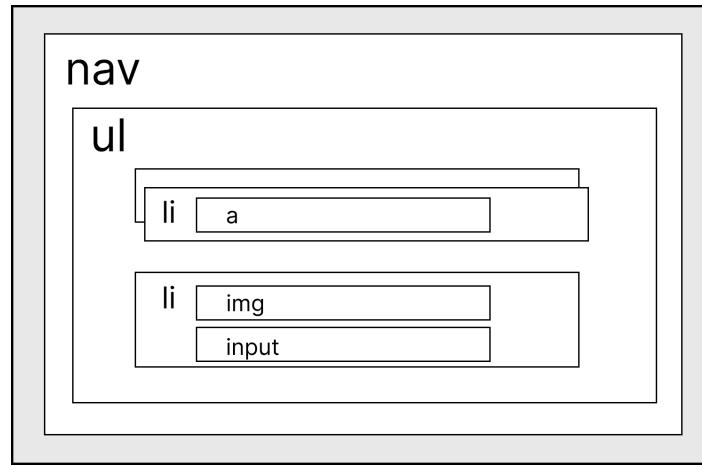


Figura 2.5: Mapa de etiquetas navegación secundaria

Similar a lo que veíamos en el mapa de etiquetas del menú de navegación principal, esta barra de navegación secundaria está compuesta por una lista de enlaces que llevan a las distintas subpáginas de descubrir destinos. Adicionalmente, el elemento de búsqueda también se incluye en la barra de navegación, pero en este caso no está compuesto por un enlace, sino por una imagen para la lupa y un campo de input de texto para poder hacer búsquedas (las cuáles no se implementarán por falta de un servidor que respalde el sitio web, pero así serán reflejadas en el frontend de la página).

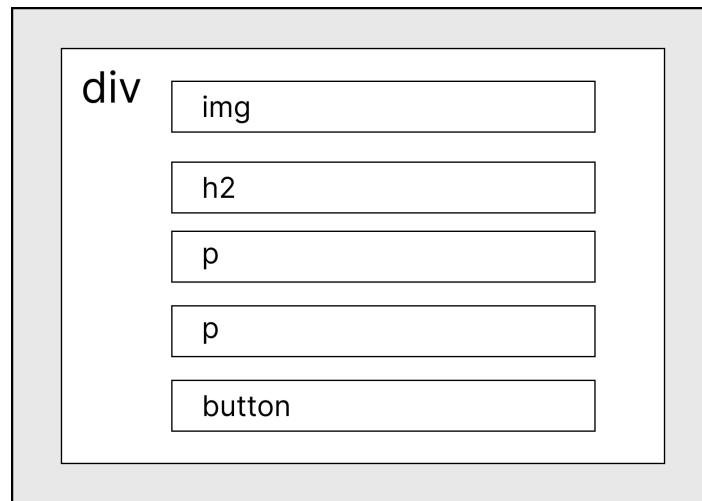


Figura 2.6: Mapa de etiquetas tarjeta de destino

Todas las tarjetas de destino se engloban en una sección del documento HTML, y cada una de ellas se rige por el mapa de etiquetas que se muestra. Se usa un `div` para englobar todos los elementos en una unidad fácilmente replicable. Así, cada tarjeta está compuesta por una imagen, correspondiente a la foto del destino, junto a un título `h2` que indica el nombre del mismo, un par de párrafos con la descripción y el rango de precios y un botón de reserva.

2.3. Planifica

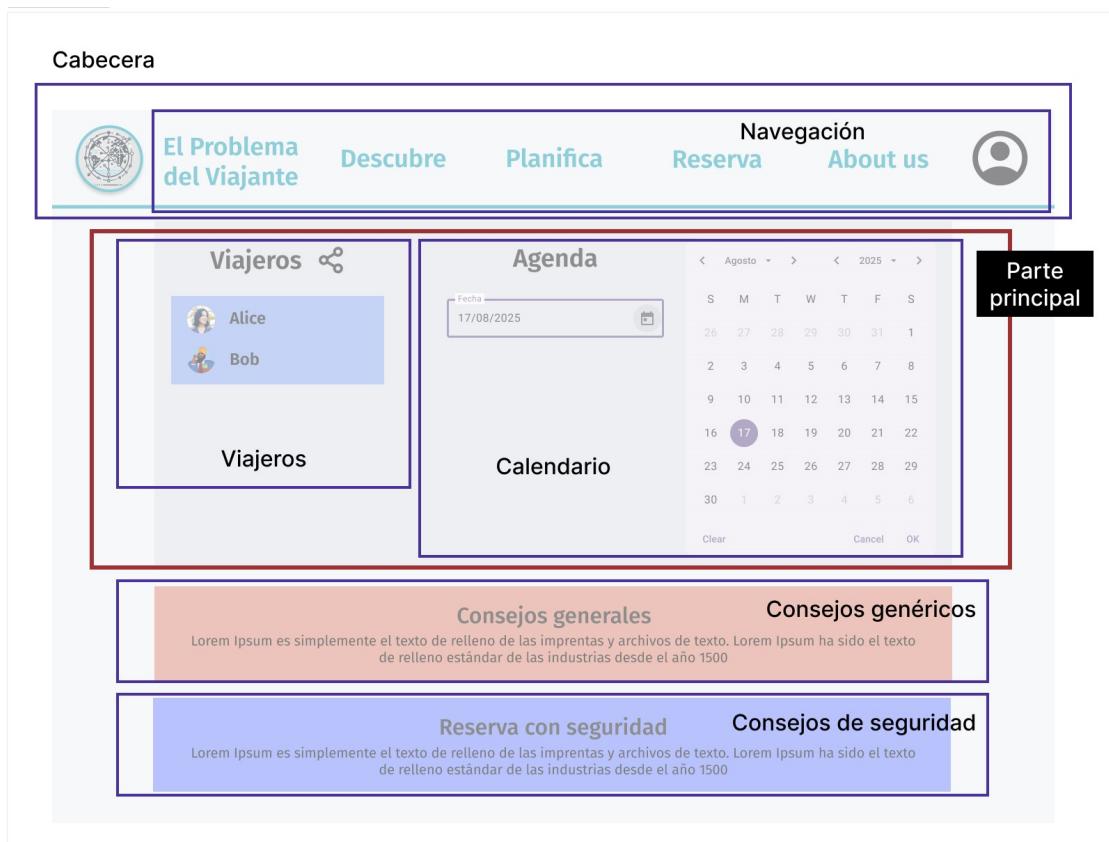


Figura 2.7: Elementos Planifica

En la parte principal de la planificación, se puede ver una lista de personas con los que se comparte el viaje y la organización de las actividades del mismo. Junto a él, el calendario donde propiamente se gestiona la planificación. Se entrará en más detalles en el mapa de etiquetas de cada uno. En la zona inferior, se muestran diferentes consejos que pueden ayudar a la gestión del viaje.

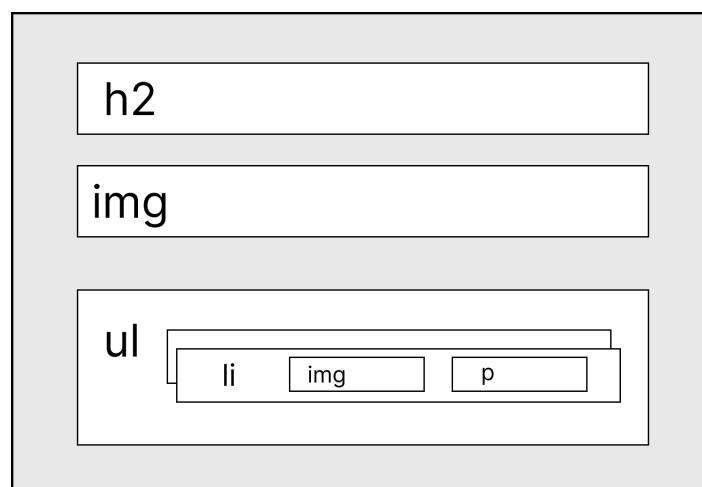


Figura 2.8: Mapa de etiquetas viajeros

En la zona de los viajeros, se muestra el título junto a una imagen, que es un botón para compartir la planificación con otras personas, y así poder añadirlas al viaje fácilmente. A continuación, se muestra una lista de usuarios, que son precisamente los que ya se han incluido en la gestión del viaje y, de forma similar a como aparecía en la caja de valoraciones, aparece su foto de perfil y su nombre.

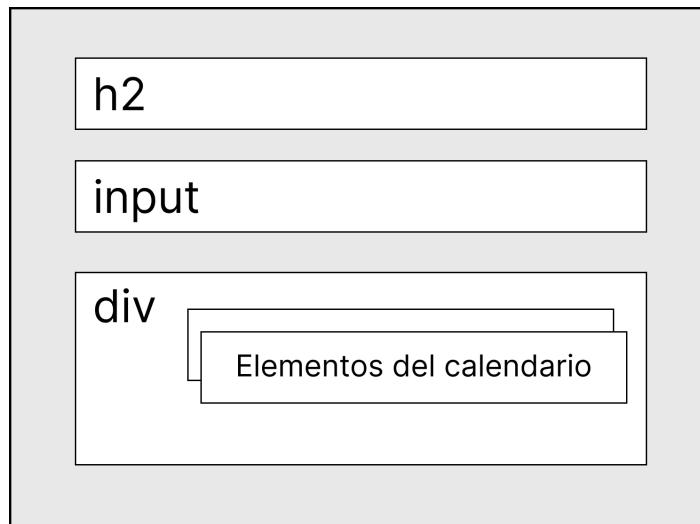


Figura 2.9: Mapa de etiquetas calendario

La caja del calendario es la más importante de todas, pues es la que permitirá realmente la gestión del viaje. En el mapa de etiquetas aparece el título junto a un **input**, desde el cual se puede introducir una fecha, asociada a la posición del calendario. El calendario en sí no está programado con HTML, ya que será necesario utilizar JavaScript para dotarlo de la responsividad que necesita.

2.4. Reserva

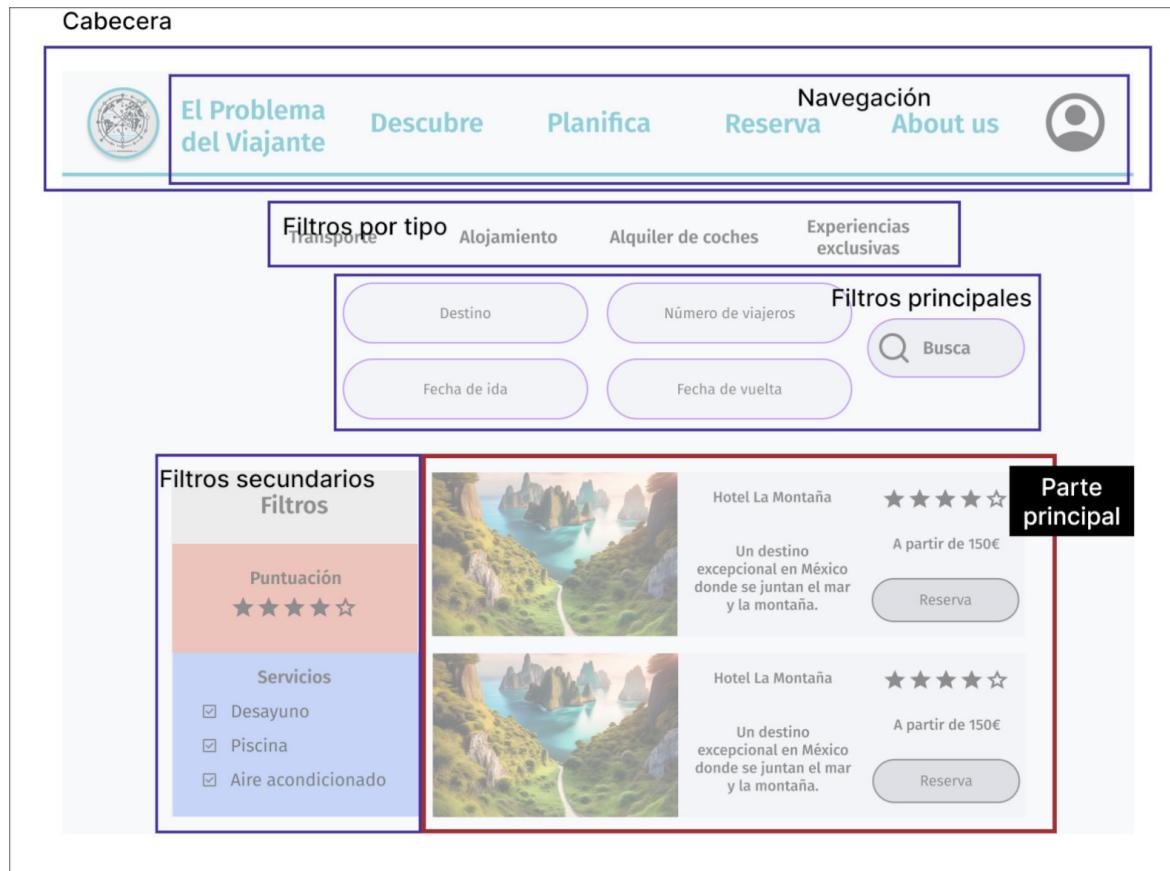


Figura 2.10: Elementos Reserva

La página de reservas muestra una lista de etiquetas de servicios para reservar, desde transporte para llegar al destino, alojamiento, alquiler de coches y otro tipo de experiencias. Se compone de una barra de navegación secundaria con una estructura muy similar a la que veíamos en la página Descubre para filtrar por el tipo de servicio. Abajo, hay una sección con otros filtros principales según el nombre del destino, número de viajeros y fechas del viaje. En la sección inferior, está en un lateral una barra para aplicar más filtros por puntuación y servicios ofrecidos, y en el centro una colección de tarjetas con los servicios similar a como está organizada la página de Descubre.

Cada una de las tarjetas de servicios tiene una estructura análoga a las que había para las tarjetas de destinos, y lo mismo ocurre con la barra de navegación secundaria, por lo que no repetiremos esos mapas de etiquetas. Nos centraremos entonces en mostrar los mapas de etiquetas para los filtros principales y secundarios.

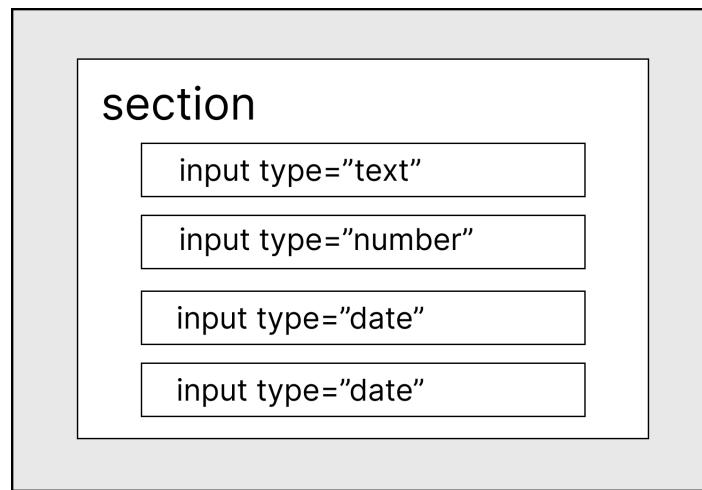


Figura 2.11: Mapa de etiquetas filtros principales

Como se puede apreciar en el mapa de etiquetas, los filtros principales se corresponden con una sección con cuatro campos de input, encargados de filtrar por nombre de destino, número de viajeros y fechas de ida y vuelta, para cada uno de los cuales se usa el tipo de input apropiado. Se decidió eliminar la búsqueda de esta sección de filtros principales, pues no aportaba una mejor capacidad de encontrar el servicio que los filtros ya presentes. Además, se prescindió de englobar todos los inputs en una etiqueta **form**, pues se planea hacer el filtrado dinámicamente utilizando JavaScript, por lo que no hay necesidad de enviar los datos a ningún servidor.

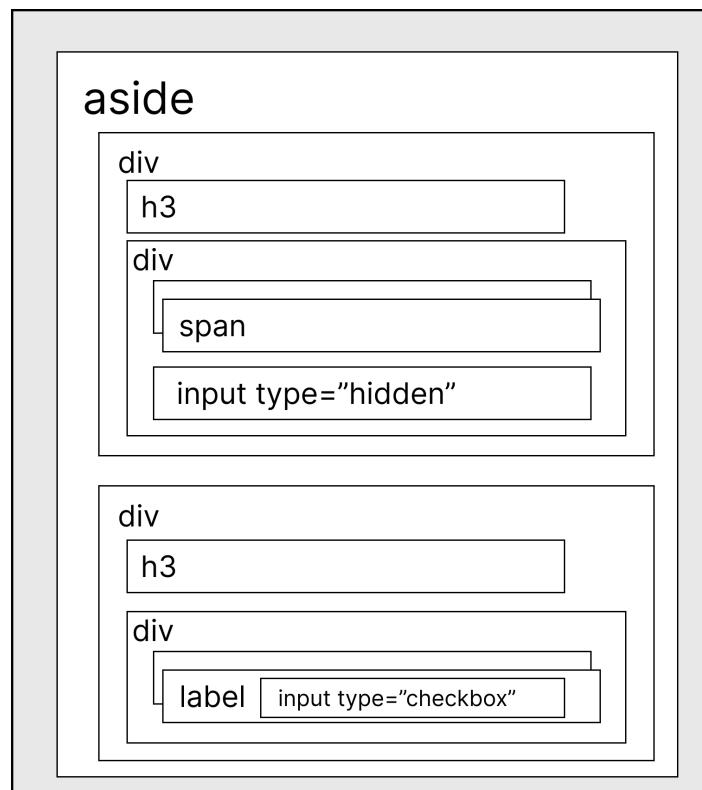


Figura 2.12: Mapa de etiquetas filtros secundarios

Los filtros secundarios están en un **aside**, que indica que aparecerán situados al margen de la página. Están divididos en dos secciones, cada una de ellas en un **div**, con cabeceras

indicando los filtros por puntuación y por servicios. Dentro del filtro por puntuación, hay otro `div` con varios elementos `span`; cada uno de ellos contiene una estrella que el usuario podrá pulsar para seleccionar la puntuación mínima de los servicios a visualizar. Se añade un campo `input` oculto para almacenar el número de estrellas seleccionadas. Los filtros por servicio son simplemente una serie de etiquetas de `input checkbox` metidas en etiquetas `label` para el nombre del servicio a escoger.

2.5. About Us



Figura 2.13: Elementos About us

Esta es la página más simple de las cinco principales, ya que tiene una zona principal con los diferentes apartados que hablan sobre la empresa, sus características y preguntas frecuentes. Por otro lado, en el lateral hay una lista con diferentes formas de contacto con la empresa y todas sus redes sociales, desde las que se podría acceder a más información sobre ella, y ver más imágenes sobre posibles destinos de vacaciones.

La mayoría de las otras páginas de la web son muy semejantes a las mostradas (por ejemplo, en las que tienen un menú de navegación secundario, las diferencias entre sí son ínfimas), y otras como la pestaña de Política de privacidad o Términos y condiciones solamente tienen contenido de texto a una columna, por lo que no aporta información mostrar su mapa de etiquetas.

Capítulo 3

Documentación CSS

3.1. Flexible Grids

Este es el mecanismo convencional para conseguir la responsividad, mediante el uso *float* y con anchos relativos. Se ha utilizado en la página About us de la siguiente manera: hay una zona principal en la que se comenta la historia de la empresa y los estándares de calidad, y un apartado secundario con información de contacto y redes sociales.

El reparto del espacio no es equitativo, sino que se le da mayor importancia a los elementos principales. Además, se limita el ancho máximo de cada columna con el objetivo de que no se expandan horizontalmente demasiado en pantallas grandes. Se añaden dos puntos de ruptura, uno que reajusta los anchos de cada columna en pantallas de tamaño medio, y otro que eliminar el *float* y convierte el diseño a una columna en pantallas pequeñas. A continuación se muestra un código simplificado del funcionamiento de *flexible grids* y sus correspondientes capturas de pantalla.

```
1 .column1 {  
2     width: 65%;  
3     max-width: 800px;  
4     float: left;  
5 }  
6 .column2 {  
7     width: 25%;  
8     max-width: 300px;  
9     float: right;  
10 }  
11  
12 @media screen and (max-width: 900px) {  
13     .column1 {  
14         width: 90%;  
15         float: none;  
16     }  
17     .column2 {  
18         width: 90%;  
19         float: none;  
20     }  
21 }
```



Figura 3.1: Página About us - Ancho 1920px

La figura anterior muestra cómo se vería en un ordenador actual, o en una pantalla de tamaño grande. El contenido aparece centrado, y la información de contacto y redes sociales hacia el lateral.



Figura 3.2: Página About us - Ancho 1024px

Así se mostraría el contenido en una pantalla de tamaño mediano. La letra de los párrafos es ligeramente más pequeña, y cada columna es un poco menos ancha, para ajustarse a una pantalla más estrecha.



Figura 3.3: Página About us - Ancho 768px

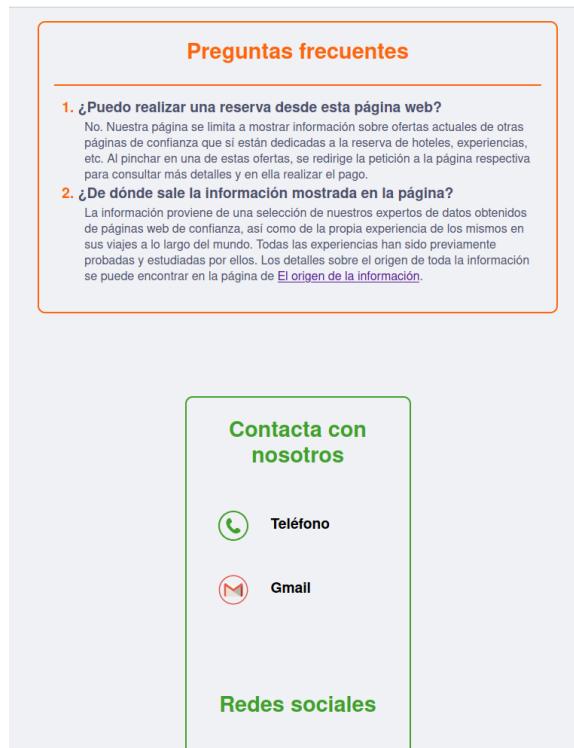


Figura 3.4: Página About us - Ancho 768px

De esta forma se muestra en un móvil, eliminando el diseño anterior y pasando a una columna. La sección de contacto y redes sociales ahora se mostraría más abajo, pero centrada. También se ajusta la letra y el interlineado consecuentemente.

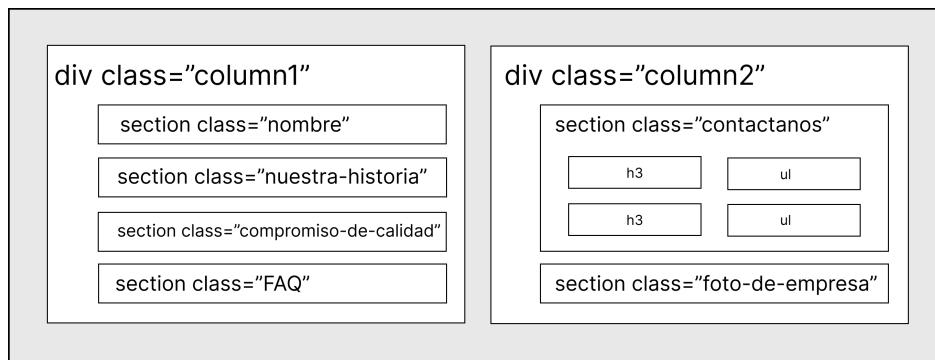


Figura 3.5: Esquema About us

Se muestra en la imagen el esquema que sigue la página, que es bastante sencillo como se ha comentado. Tiene dos columnas, sobre las que se aplica el efecto, la primera está dividida en secciones, y están compuestas todas ellas por un encabezado y párrafos. La segunda columna tiene la sección de contacto y redes sociales, con dos listas, y finalmente aparece la foto de empresa.

3.2. CSS Multicol

Este es el más antiguo dentro de los métodos de layout modernos, que asumen ya que el diseño que se va a crear es flexible. Se basa en indicarle al navegador en cuántas columnas se quiere dividir un contenedor, dejando que sea el propio navegador el que haga los cálculos en función del tamaño del viewport. Esta técnica es sencilla de aplicar a priori, pero los resultados que proporciona no se adaptan del todo bien al diseño responsive y se vuelve complicado conseguir una correcta disposición de los elementos dentro de cada columna.

Se ha utilizado en la página Descubre para dividir las tarjetas de destino en tres, dos o una columnas en función del tamaño del viewport. No se aplicó a la barra de navegación principal, que ya estaba creada y es la misma para todas las páginas, ni para la barra de navegación secundaria, que ya se había creado haciendo uso de *flex* para la página Reserva, resultando en un comportamiento más fácilmente calibrable. Así mismo, como el mapa interactivo aparece en una fila por su cuenta debajo de las tarjetas, no tiene sentido aplicar *multicol* para este elemento.

Se establecieron, así, tres puntos de ruptura para pantallas grandes, medianas y pequeñas, en los que las tarjetas aparecen en 3, 2 o 1 columna, respectivamente. A continuación, se muestra un código con el funcionamiento de *multicol*.

```

1 @media screen and (min-width: 900px) {
2   #destinations {
3     column-count: 2;
4     column-gap: 20px;
5     max-width: 900px;
6   }
7 }
8
9 @media screen and (min-width: 1400px) {
10   #destinations {
11     column-count: 3;
12     column-gap: 30px;
13     max-width: 1400px;
14   }
}
  
```

```

15 }
16
17 @media screen and (max-width: 900px) {
18     #destinations {
19         column-count: 1;
20         max-width: 450px;
21     }
22 }
```

La propiedad más relevante del código es `column-count`, que establece el número de columnas en las que se va a dividir el contenedor; en este caso, para pantallas con un ancho no inferior a 1400 píxeles se posicionarán los elementos a 3 columnas, para pantallas con ancho entre los 900 y los 1400 píxeles se dispondrán en dos columnas y para pantallas de hasta 900 píxeles aparecerán a una única columna.

Se muestran a continuación algunas capturas de pantalla de la página para diferentes anchos del navegador.

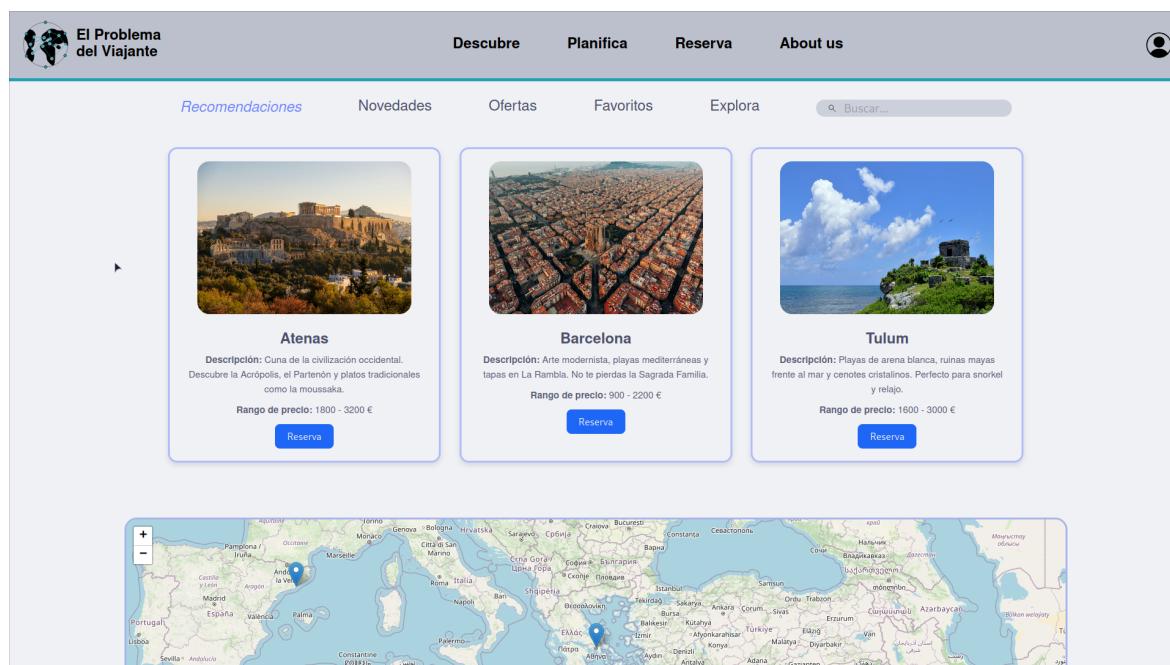


Figura 3.6: Página Descubre - Ancho 1920px

La figura anterior muestra cómo se vería en un ordenador actual, o en una pantalla de tamaño grande. El contenido aparece centrado y las tarjetas aparecen a 3 columnas. La barra de navegación secundaria ocupa una única fila (aunque, de nuevo, esto se consigue mediante `flex` y no con `multicol`).

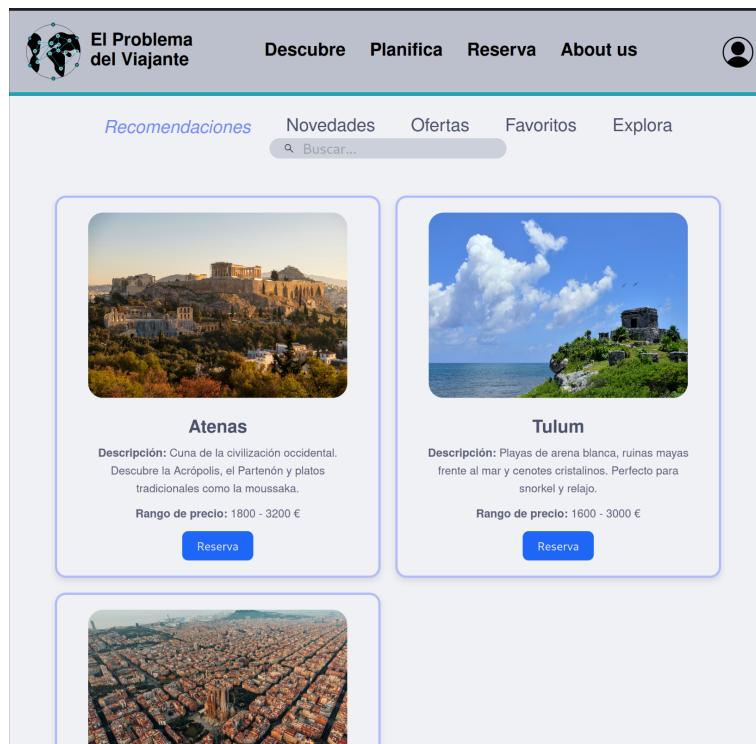


Figura 3.7: Página Descubre - Ancho 1024px

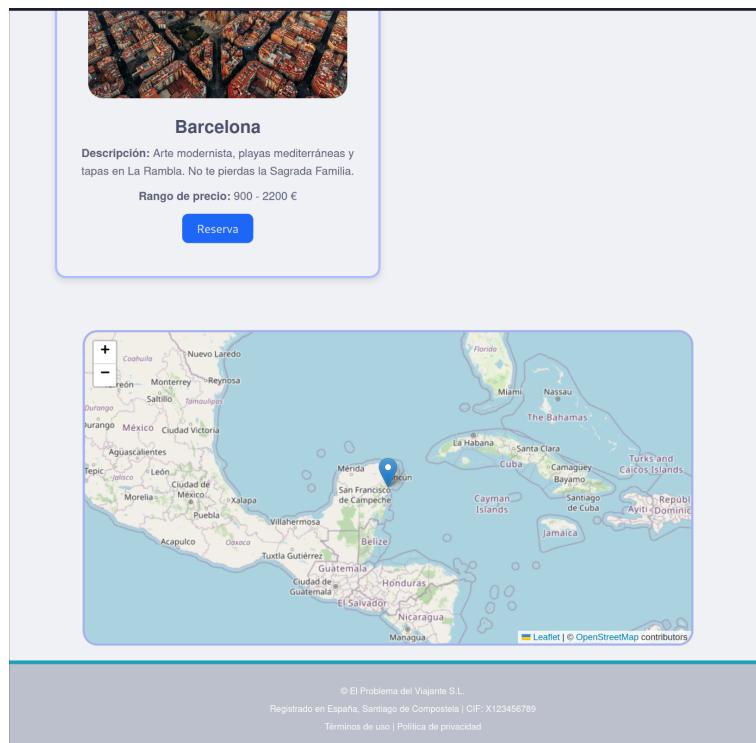


Figura 3.8: Página Descubre - Ancho 1024px

En pantallas de tamaño medio, se pasa al diseño de dos columnas. El mapa se rescala para ocupar el mismo espacio proporcional de la página y la barra de búsqueda se sitúa en una fila inferior. Las tarjetas, que son las afectadas por *multicol*, conservan su tamaño original, pero

ahora se distribuyen en dos columnas para caber en la pantalla.

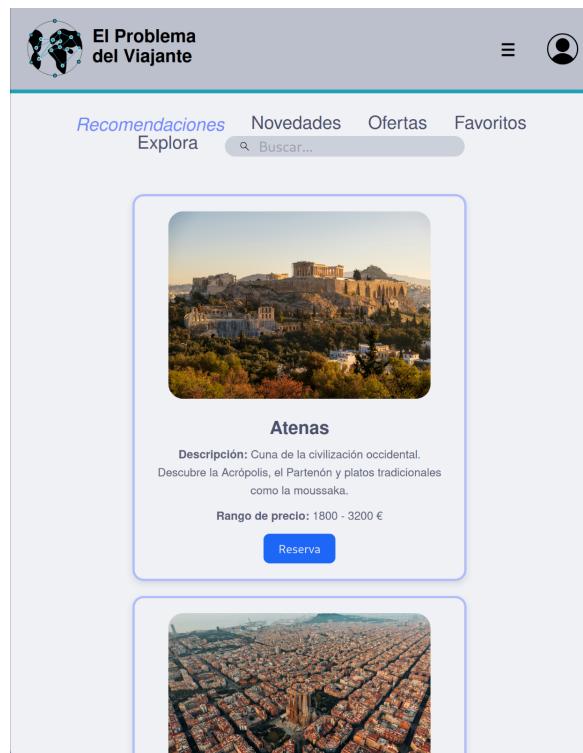


Figura 3.9: Página Descubre - Ancho 768px

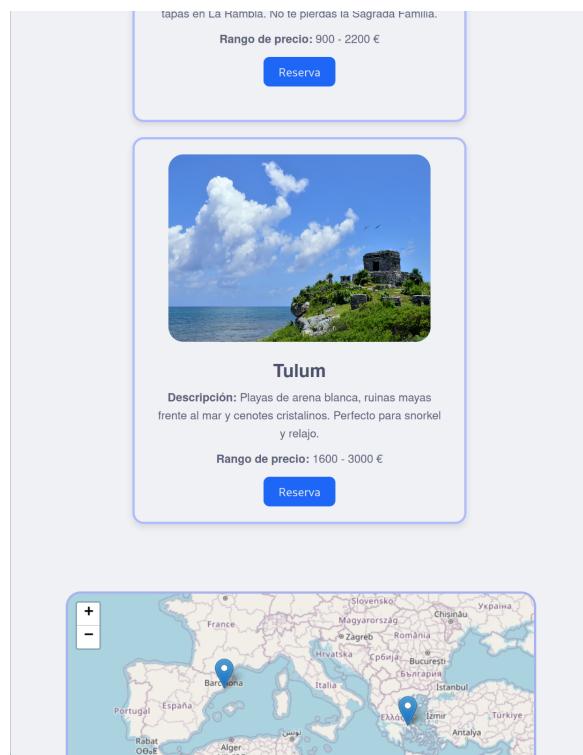


Figura 3.10: Página Descubre - Ancho 768px

Por último, vemos como el diseño a una columna se ajusta correctamente a pantallas pequeñas,

de forma similar al anterior.

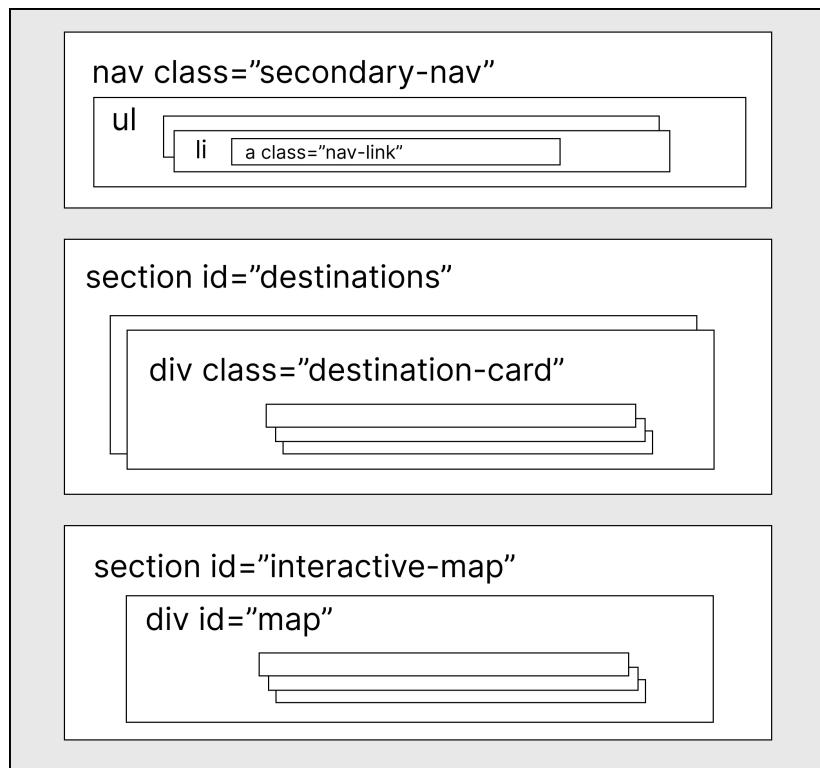


Figura 3.11: Esquema Descubre

Como ya se comentó, el uso de *CSS Multicol* se usó para la sección que en el esquema aparece con el identificador *destinations*. El uso de *multicol* es sencillo, pero cuando se quiere utilizar para configurar la disposición de elementos que no sean puramente texto se vuelve complicado obtener resultados con una apariencia satisfactoria, por la dificultad para centrar los elementos o asegurar que cada parte va a la columna que le corresponda. Se incluyen en el diagrama las clases de los elementos más relevantes y que afectan a la disposición de la página.

3.3. Flex Container

El mecanismo *Flex Container* permite crear con facilidad un contenedor flexible para facilitar la disposición de elementos en una dimensión, bien sea en filas o en columnas. Este diseño se aplicó a la página Reserva y en este caso a todos sus elementos, por su facilidad de uso, versatilidad y simplicidad, que lo convierten en una muy buena opción por defecto para el posicionamiento de los elementos en una página responsive, si bien puede refinarse mediante el uso de *CSS grid* en caso de necesitar mayor control en dos dimensiones.

El uso de *flex* simplifica el uso de los puntos de ruptura, pues en este caso los elementos ya se distribuyen automáticamente ocupando el espacio que necesiten, de forma que el uso de los puntos de ruptura se puede limitar a cambios más significativos en la apariencia de la página, a diferencia de lo que pasaba con *multicol*. A continuación se muestra un código CSS simplificado en el que se aprecia el uso de este mecanismo.

```

1 | #main-filters {
2 |   display: flex;
3 |   flex-wrap: wrap;
4 |   justify-content: center;
  
```

```
5 }
6
7 .reservations-container {
8     display: flex;
9     align-items: flex-start;
10    overflow-x: hidden;
11 }
12
13 .checkbox-group {
14     display: flex;
15     flex-direction: column;
16     overflow-y: auto;
17 }
18
19 @media screen and (max-width: 1400px) {
20     .reservations-container {
21         width: 90%;
22     }
23 }
24
25 @media screen and (max-width: 900px) {
26     .filter-input {
27         flex: 1 1 100%;
28         max-width: 80%;
29     }
30
31     .reservations-container {
32         flex-direction: column;
33     }
34
35     #sidebar-filters {
36         max-height: none;
37         width: 80%;
38         margin: auto;
39     }
40
41     #service-cards {
42         max-height: none;
43         justify-content: center;
44     }
45 }
```

Como vemos, este mecanismo se caracteriza por el uso de `display: flex`, y gracias a este se pueden utilizar múltiples propiedades como `flex-direction` para controlar si la disposición es por filas o por columnas o `justify-content` y `align-items` para justificar y alinear el contenido de la página de una forma sencilla.

Se muestran ahora capturas de pantalla de esta página para diferentes tamaños del navegador.

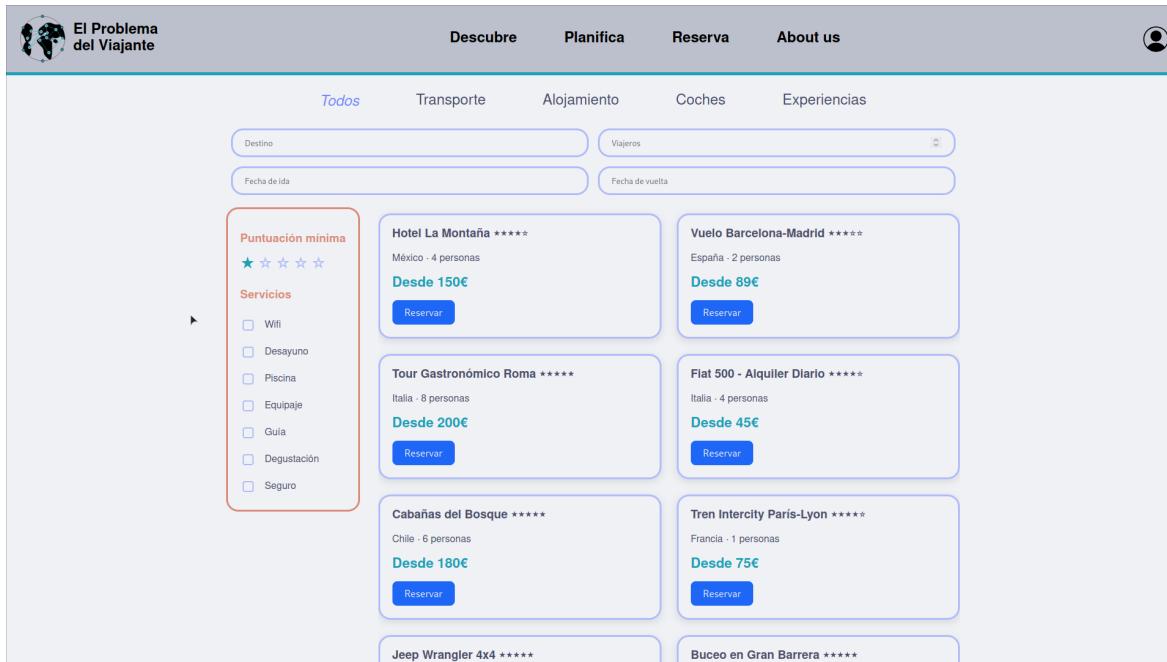


Figura 3.12: Página Reserva - Ancho 1920px

En pantallas grandes, el contenido se ve de esta forma. Como vemos, el menú de navegación principal aparece en dos filas y dos columnas, hay una barra lateral para los filtros secundarios y las tarjetas de servicios aparecen en dos columnas al lado de los filtros secundarios. Cabe mencionar las diferencias respecto al diseño original, en especial la eliminación de las imágenes de las tarjetas para no sobrecargar la página, como se explica en la sección de Cambios 5.2.

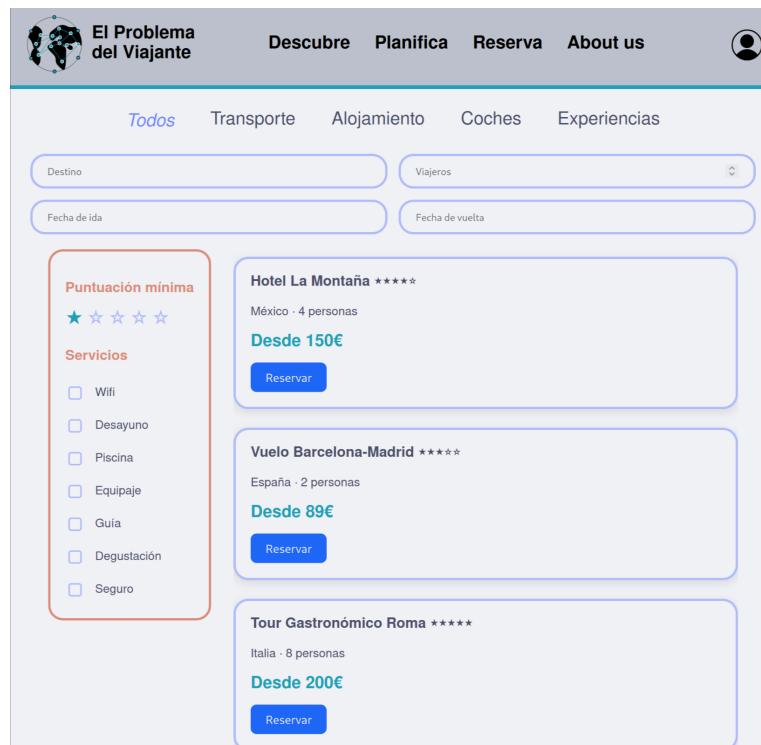


Figura 3.13: Página Reserva - Ancho 1024px

En pantallas medianas, los filtros principales siguen apareciendo en dos columnas y dos filas y la barra de filtros secundarios a la izquierda de las tarjetas, pero en este caso las tarjetas se muestran por fila de una en una. Como se puede ver en el código, la única regla @media que añadimos para este tamaño simplemente reduce un poco el contenedor donde están las tarjetas y la barra lateral para que aparezcan más centrados. El hecho de que se muestren ahora las tarjetas a una columna en lugar de a dos es consecuencia automática de utilizar *flex* para su disposición, de nuevo dejando patente la versatilidad de este método.

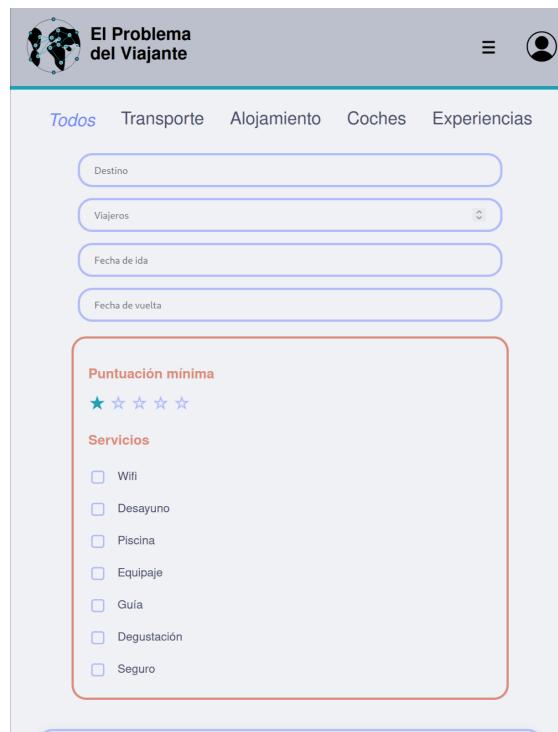


Figura 3.14: Página Reserva - Ancho 768px

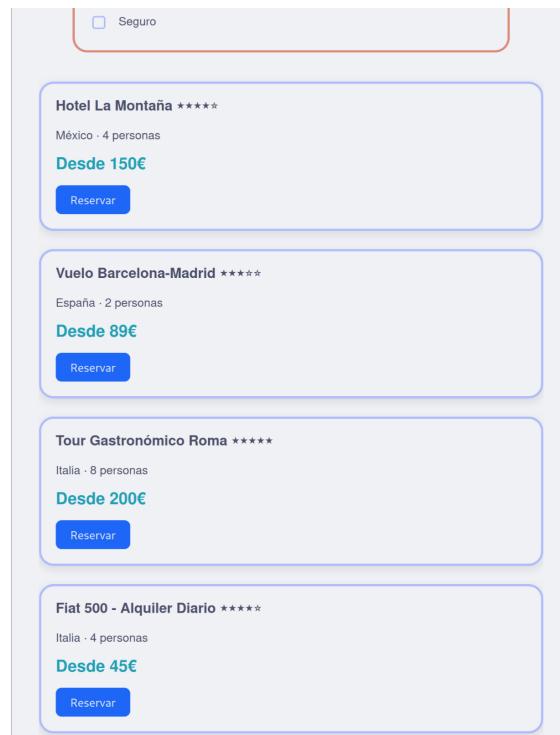


Figura 3.15: Página Reserva - Ancho 768px

Por último, para pantallas pequeñas vemos cómo los filtros principales se disponen cada uno en una columna. Igualmente, ahora la barra de navegación lateral se sitúa por sí misma ocupando todo el ancho de la página y las tarjetas de los servicios aparecen debajo de esta, también a una sola columna.

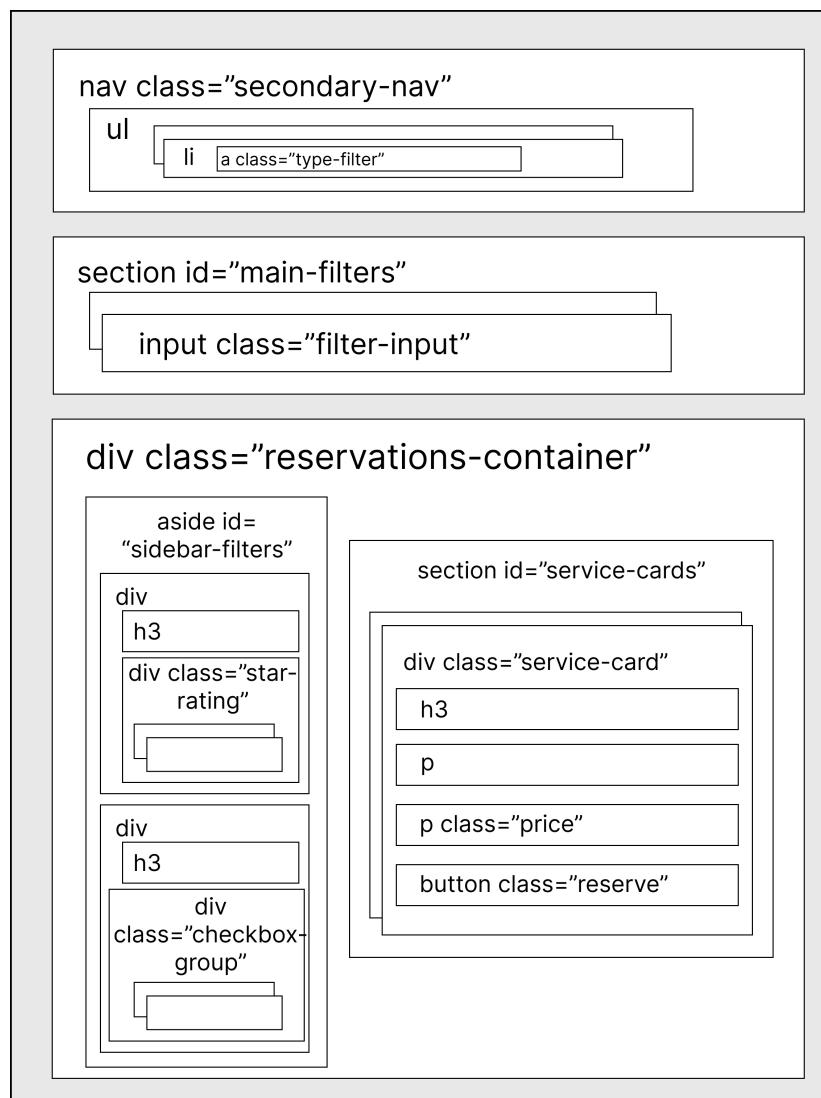


Figura 3.16: Esquema Reserva

Para el esquema de esta página se ha decidido usar la misma disposición que aparece en la página real, con el objetivo de facilitar la compresión de la posición de cada elemento. Se muestran las clases o identificadores de los elementos más relevantes para obtener el aspecto deseado mediante *flex*.

3.4. CSS Grid

El mecanismo *CSS Grid* permite ajustar con precisión elementos en dos dimensiones, alineando con facilidad tanto las filas como las columnas. Ha resultado muy útil su uso en la página Planifica, ya que hay muchos elementos dispuestos de forma horizontal, y que van cambiando hacia un diseño más vertical según se reduce el ancho de la pantalla.

De esta forma, se parte de un diseño en el que aparecen en la primera fila tanto los viajeros como el calendario de eventos. Un primer punto de ruptura acerca el contenido manteniendo su estructura, y el segundo punto de ruptura pasa a un diseño en el que se muestra todo en una columna, tal y como se ve en el código simplificado siguiente.

```
1 | .grid-container {
```

```

2   display: grid;
3   grid-template-columns: 1fr 1fr 1fr;
4   grid-template-rows: auto;
5   grid-template-areas:
6     "travellers-title calendar-title calendar-title"
7     "travellers         calendar-events calendar"
8     "advices           advices           advices"
9     "security          security          security"
10 }
11
12 @media screen and (max-width: 1100px){
13   .grid-container {
14     grid-template-areas:
15       "travellers-title"
16       "travellers"
17       "calendar-title"
18       "calendar"
19       "calendar-events"
20       "advices"
21       "security"
22     }
23 }
```

La propiedad más relevante del código es `grid-template-aread`, la cual, como indica su nombre, crea una plantilla de la cuadrícula, por lo que se puede ajustar fácilmente la disposición de los elementos asignándole a cada uno su área. Se pasa así de un diseño en tres columnas para pantallas más anchas, a uno de una columna en dispositivos más pequeños.



Figura 3.17: Página Planifica - Ancho 1920px

En pantallas grandes, se muestra el `grid` de esta forma. Como se ha comentado, aparece el calendario a la derecha, y los viajeros a la izquierda

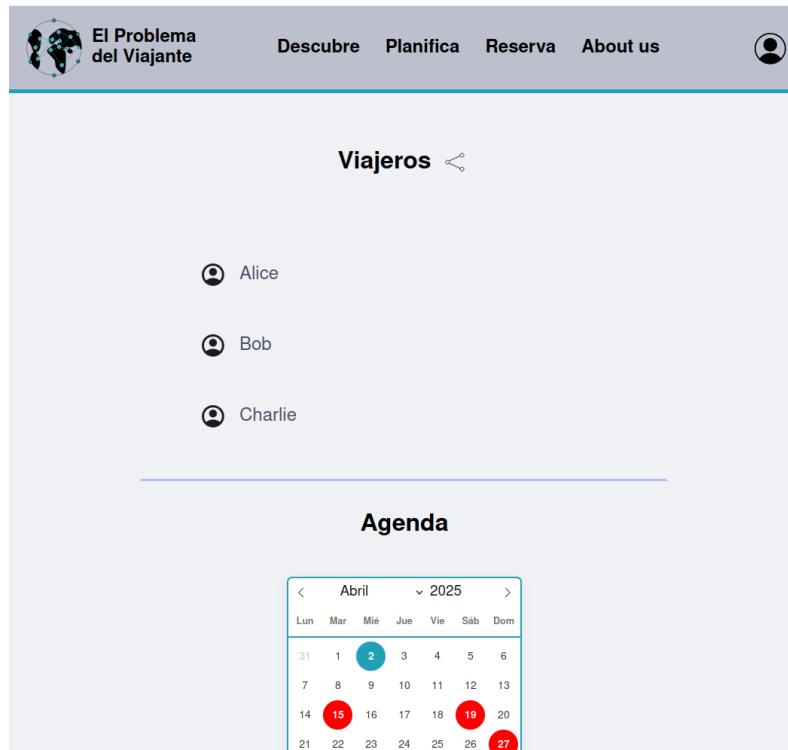


Figura 3.18: Página Planifica - Ancho 1024px

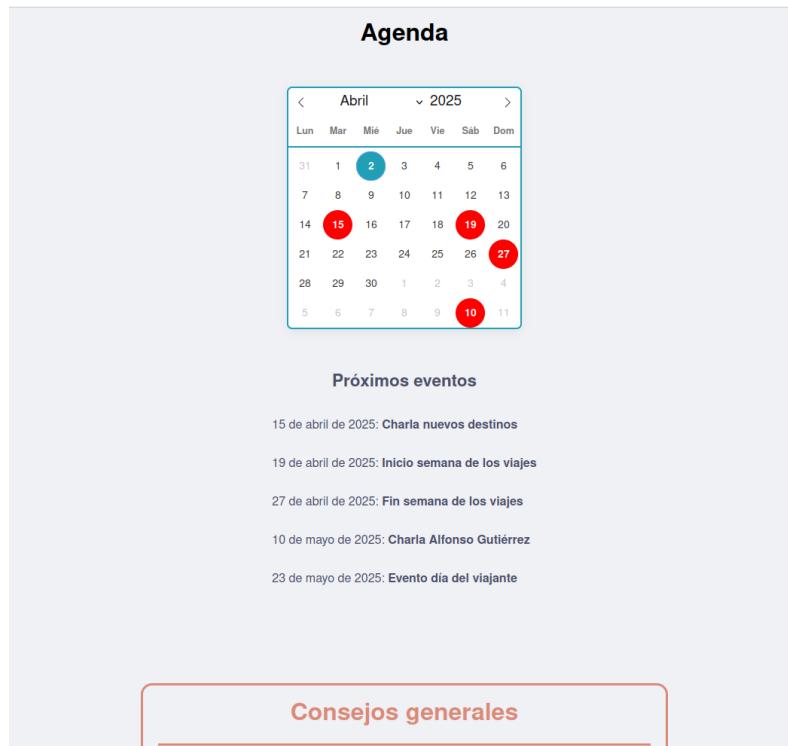


Figura 3.19: Página Planifica - Ancho 1024px

En pantallas de tamaño medio, se pasa al diseño de una columna. La ventaja de usar *CSS Grid* es que sería sencillo realizar cambios para que en un determinado lugar hubiese dos columnas, si así se quisiese.

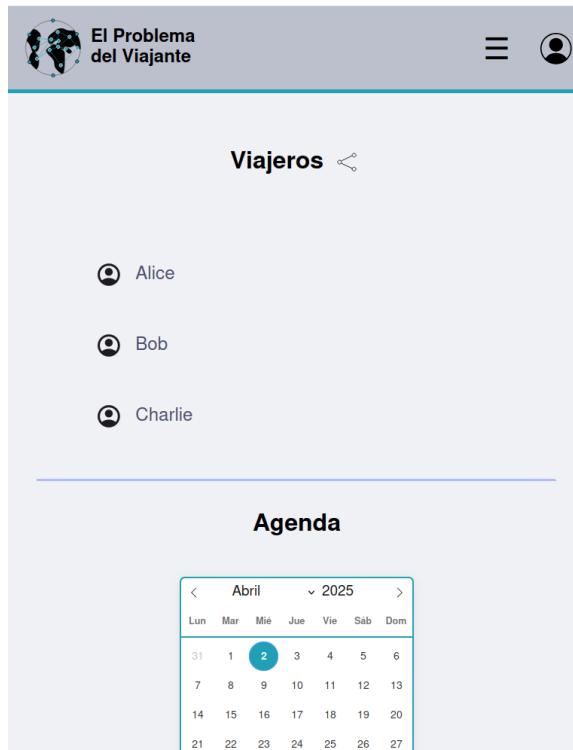


Figura 3.20: Página Planifica - Ancho 768px



Figura 3.21: Página Planifica - Ancho 768px

Por último, el diseño se ajusta correctamente a pantallas pequeñas, de forma similar al anterior.

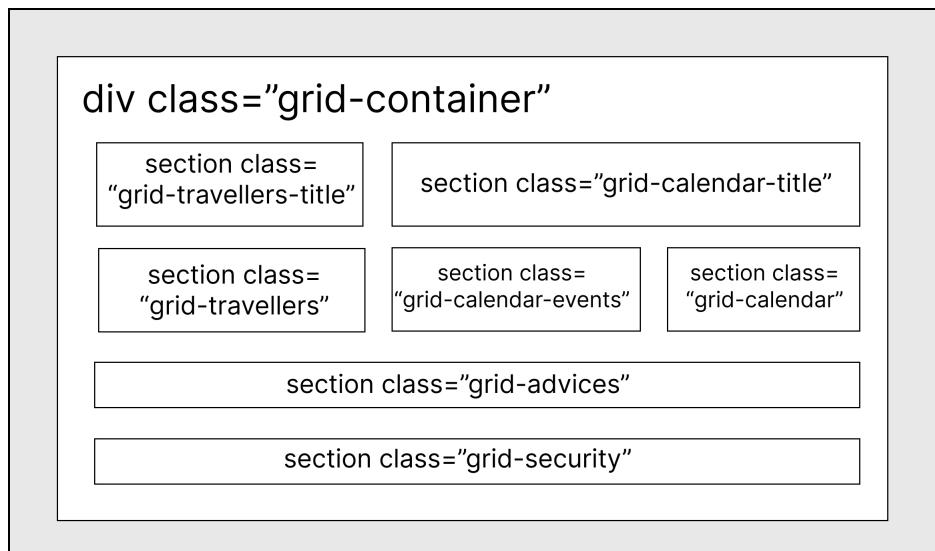


Figura 3.22: Esquema Planifica

Para el esquema de esta página se ha decidido usar la misma disposición que aparece en el *grid*, con el objetivo de facilitar la compresión de la posición de cada elemento en la página. Las clases de cada uno se corresponden precisamente con los nombres de la plantilla de *grid* que se muestra en el código, junto a la palabra *grid*, y que están asignados en el archivo CSS.

3.5. Bootstrap

Bootstrap es una librería de reglas CSS que permite reducir significativamente la cantidad de código propio necesario para desarrollar la página, a cambio de utilizar ciertas características predeterminadas. También permite añadir más variabilidad a los diseños, y en este caso se ha realizado un cambio respecto al diseño inicial en la página principal, pues se ha incluido un carrusel de imágenes (más detalles en el Anexo).

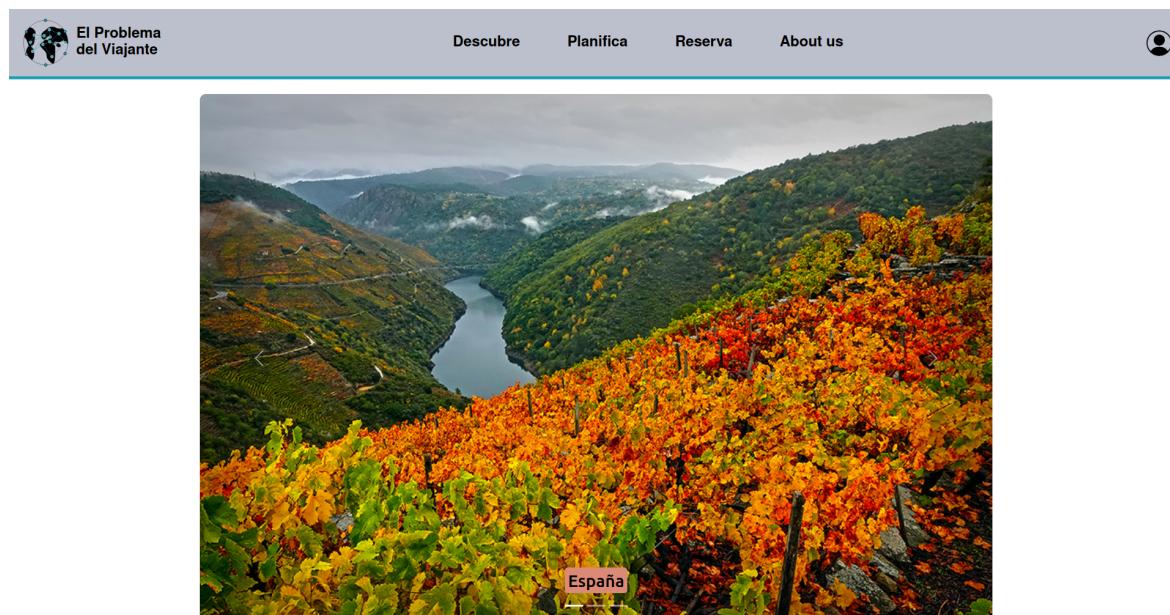


Figura 3.23: Página principal - Ancho 1920px

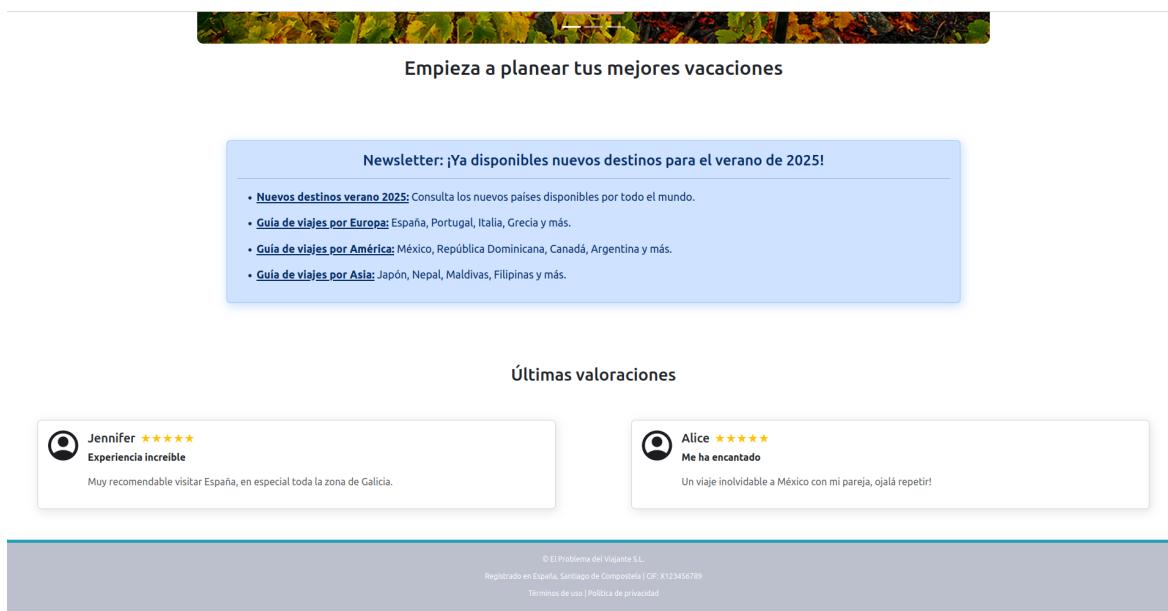


Figura 3.24: Página principal - Ancho 1920px

Se muestra el carrusel de fotos como elemento principal, lo cual es una característica muy atractiva para el usuario, con el objetivo de que muestre interés en el contenido del resto de la página. Más abajo se encuentra un recuadro de noticias recientes y un apartado de valoraciones ficticias de otros usuarios de la página.

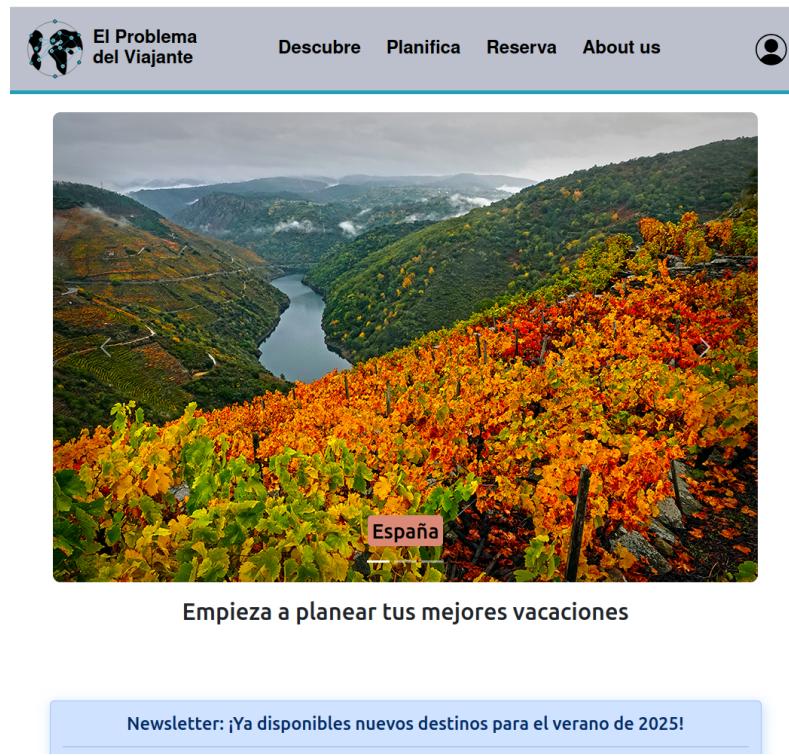


Figura 3.25: Página principal - Ancho 1024px

Otro aspecto positivo de utilizar *Bootstrap* es que él mismo gestiona los puntos de ruptura, lo cual también facilita el formateo de la página. Otro elemento destacable es que se han

incluido imágenes de diferentes tamaños con el objetivo de minimizar el ancho de banda utilizado, especialmente pensado para dispositivos móviles, los cuales suelen tener conexiones más débiles e inestables.



Figura 3.26: Página principal - Ancho 768px

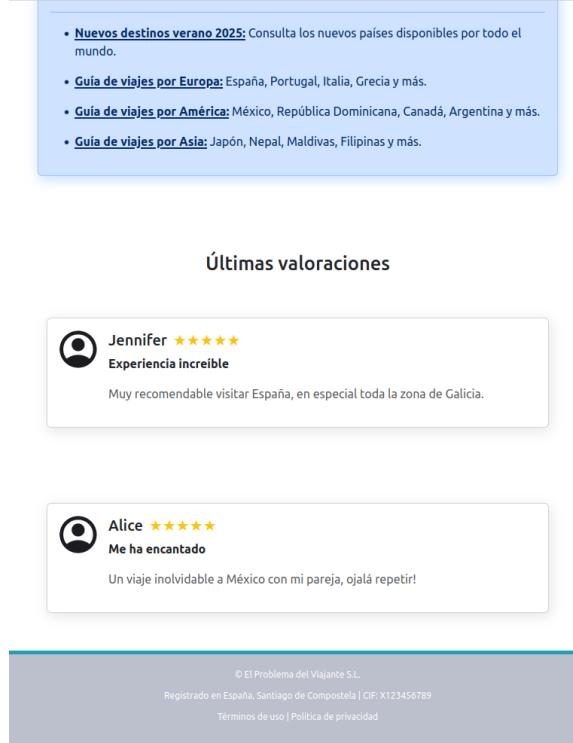


Figura 3.27: Página principal - Ancho 768px

Finalmente se muestra la versión de móvil, con su respectiva imagen del carrusel redimensionada y con un diseño a una columna de las valoraciones de usuarios.

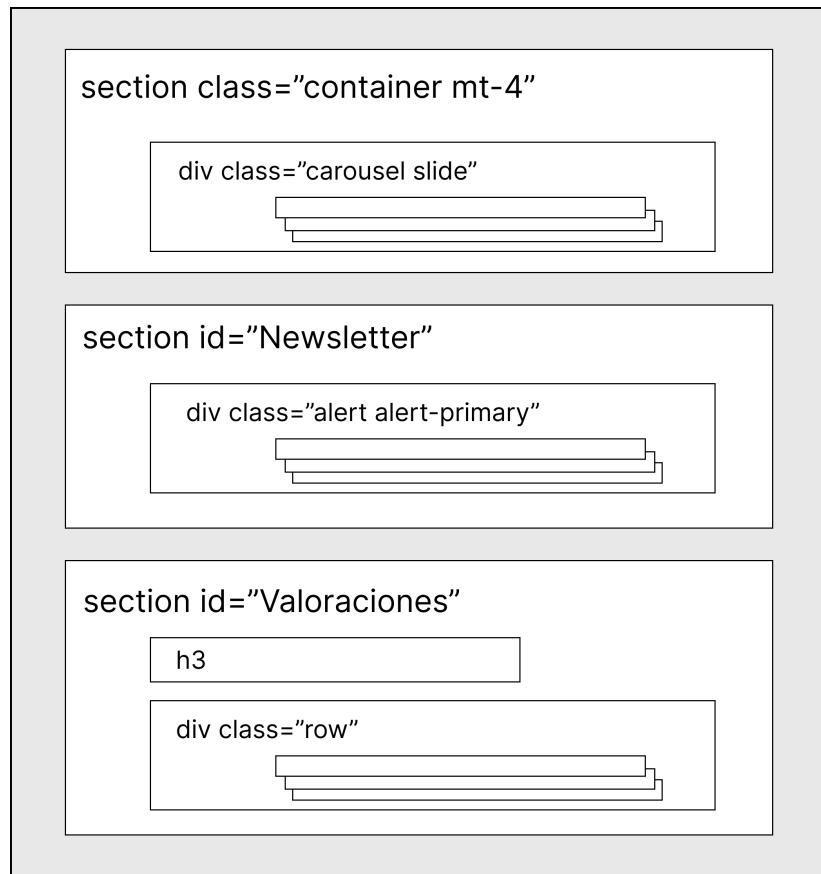


Figura 3.28: Esquema Página principal

Una de las mayores desventajas que se ha encontrado con el uso de *Bootstrap* es la gran cantidad de **divs** necesarios para organizar el contenido, debido a que hay que utilizar las clases predeterminadas de la librería. Solo se incluyen en este caso los **divs** más exteriores de cada elemento relevante para facilitar la comprensión de la imagen, los cuales son el carrusel de fotos, el recuadro de noticias y la sección de valoraciones.

3.6. Otros elementos de CSS

Aunque no se mencionen explícitamente, hay otros muchos elementos de la página que son también responsivos con el objetivo de mejorar la apariencia visual y de ayudar al usuario a reconocer en qué lugar está situado el ratón. Es por eso que en muchas páginas se pueden encontrar:

- Elementos que al situar el ratón encima cambian de color. Por ejemplo, en el menú de navegación o en el calendario.
- Elementos que resaltan la caja que los contiene, como en la sección de redes sociales de About us y en las tarjetas de Reserva.
- Elementos que se mueven ligeramente hacia arriba al situar el ratón encima y aumentan la sombra de su caja, de forma que parezca que el elemento sobresale de la pantalla,

destacándolo.

Las páginas no principales incluyen un formateo sencillo, pues son en su mayoría bloques de texto. En ellas, solo se ajusta que aparezca el contenido en el centro de la página, y se adapte al ancho de la pantalla junto al tamaño de letra y al interlineado.

Capítulo 4

Documentación JavaScript

Se ha incluido código JavaScript en tres elementos principales, los cuales se comentarán ahora para posteriormente entrar en detalles específicos en cada uno de los apartados posteriores.

- **Inclusión dinámica del menú de navegación y del pie de página:** con el objetivo de evitar copiar y pegar constantemente estos elementos, con el riesgo de realizarlo de forma incorrecta, en especial si se realiza un cambio, se ha decidido añadirlo mediante un código JavaScript. Este calcula dinámicamente la dirección del directorio raíz a partir de la dirección de la página en la que se encuentra, y funciona tanto con un servidor como sin él, sin importar dónde esté ubicada la raíz.
- **Calendario responsive:** se ha utilizado el calendario proporcionado por la página web flatpickr para añadir sobre él eventos cargados desde un fichero, los cuales también se muestran en la pestaña Planifica.
- **Generación de tarjetas:** se ha adaptado el código HTML de las páginas Descubre y Reserva, cuyo contenido principal se basa en la presencia de tarjetas con los destinos y servicios recomendados, para que ahora dichas tarjetas se generen dinámicamente a partir de archivos de datos en formato XML y JSON, respectivamente. Además, el uso e integración del JavaScript y las facilidades que proporciona la librería de jQuery permiten dotar a estas páginas de una gran interactividad, posibilitando la interacción entre las tarjetas y el mapa generado mediante la librería Leaflet [11], así como la aplicación de filtros sobre las tarjetas a mostrar en la página de Reserva.

4.1. Inclusión de JavaScript

4.1.1. Acceso al DOM

Se enumeran los accesos al DOM producidos en la inclusión dinámica del menú de navegación:

```
1 | document.addEventListener("DOMContentLoaded", function () {  
2 |     // Funcion que inserta el nav  
3 | }
```

En primer lugar, se crea un *listener* con una función que se ejecutará una vez que el HTML haya sido totalmente cargado y procesado por el navegador. Este tipo de acceso a eventos es la primera línea de código de todos los scripts, pues asegura que el script no se ejecuta habiendo elementos sin cargar, lo que podría causar errores.

```
1 let header = document.querySelector("header");
2 if (!header) {
3     header = document.createElement("header");
4     document.body.insertBefore(header, document.body.firstChild);
5 }
```

En segundo lugar, se busca la primera etiqueta **header** de la página, donde debe ir el menú de navegación. En caso de no encontrarla, se crea llamando a la función `createElement()`, la cual permite insertar un nuevo nodo en el DOM. Y luego se accede a un elemento del DOM, en concreto al primer nodo del cuerpo del documento, que es donde debe insertarse el menú de navegación. Por tanto, hay tres nuevos accesos al DOM en esta zona del código.

```
1 const navContainer = document.createElement("nav");
2 navContainer.className = "main-nav";
3 navContainer.innerHTML = `...`;
4
5 document.querySelector("header").appendChild(navContainer);
```

A continuación, se crea el nodo correspondiente al menú de navegación, se inserta el contenido y se realiza un nuevo acceso a elementos del DOM, insertando mediante `appendChild()` un nuevo hijo al **header**, que ya se sabe que se encuentra en el código HTML por el apartado anterior.



Figura 4.1: Página principal sin el menú de navegación

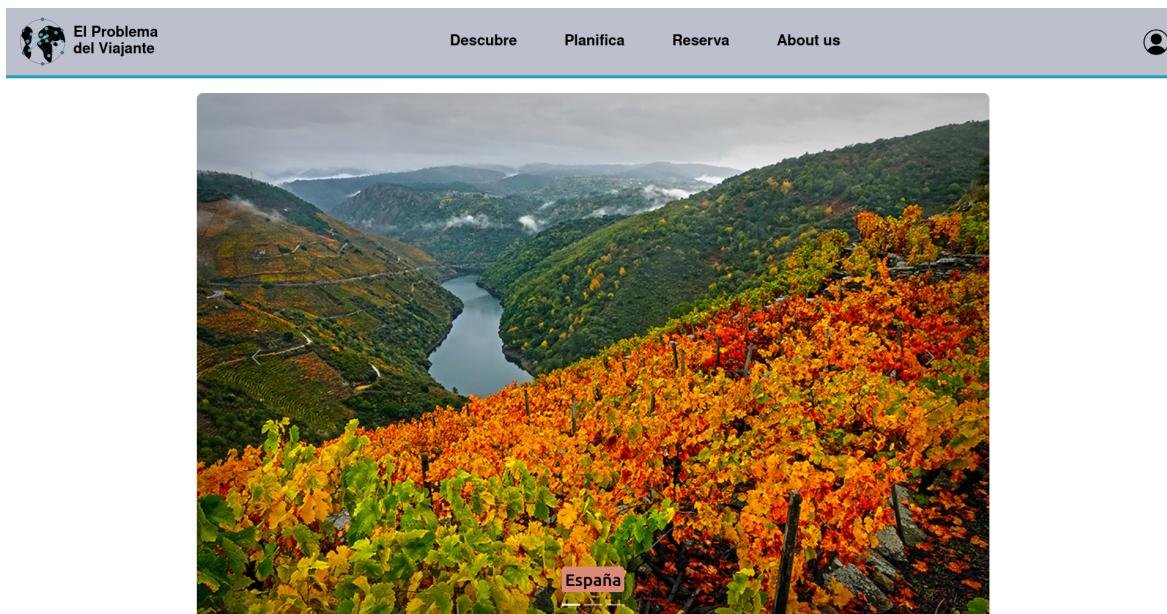


Figura 4.2: Página principal con el menú de navegación

En este caso, no hay elementos que cambien dinámicamente, por lo que solamente se pueden mostrar dos imágenes de las grandes diferencias que produce en el aspecto de la página un menú de navegación visualmente interesante respecto de no tenerlo.

```
1 | const mobileMenu = document.getElementById("mobile-menu");
2 | const closeButton = document.getElementById("close-menu");
```

Para finalizar, se incluyen dos accesos a elementos del DOM que devuelven el nodo cuyo id coincide con el nombre indicado. Estos se utilizan para la gestión del menú de navegación para móviles, que se crea mediante una barra lateral. Solamente en el código del menú de navegación hay muchas más accesos al DOM de todo tipo, pero aquí solo se incluyen algunos de los más relevantes para la comprensión del código.



Empieza a planear tus mejores vacaciones

Newsletter: ¡Ya disponibles nuevos destinos para el verano de 2025!

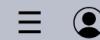
- [Nuevos destinos verano 2025](#): Consulta los nuevos países disponibles por todo el mundo.
- [Guía de viajes por Europa](#): España, Portugal, Italia, Grecia y más.
- [Guía de viajes por América](#): México, República Dominicana, Canadá, Argentina y más.
- [Guía de viajes por Asia](#): Japón, Nepal, Maldivas, Filipinas y más.

Figura 4.3: Página principal sin el menú de móvil

A mobile menu icon consisting of three horizontal lines.



El Problema
del Viajante



Empieza a planear tus mejores vacaciones

Newsletter: ¡Ya disponibles nuevos destinos para el verano de 2025!

- [Nuevos destinos verano 2025](#): Consulta los nuevos países disponibles por todo el mundo.
- [Guía de viajes por Europa](#): España, Portugal, Italia, Grecia y más.
- [Guía de viajes por América](#): México, República Dominicana, Canadá, Argentina y más.
- [Guía de viajes por Asia](#): Japón, Nepal, Maldivas, Filipinas y más.

Figura 4.4: Página principal con el menú de móvil

4.1.2. Respuesta a eventos

En relación con el menú de móvil, también hay que comentar la respuesta a eventos, la cual es necesaria para el funcionamiento correcto del mismo, ya que requiere desplazar la barra lateral a una zona visible de la pantalla, y que desaparezca otra vez cuando se cierra.

```
1 | mobileMenu.addEventListener("click", function () {  
2 |     menuSidebar.classList.add("open"); // Muestra el menu  
3 |     body.classList.toggle("open"); // Ajuste del ancho  
4 |});
```

En el elemento `mobileMenu` que se ha mencionado anteriormente, el cual se obtiene mediante un acceso al DOM, se añade un *listener* que ejecuta la función mostrada cuando se hace click sobre el botón del menú. Cuando se pulsa con el ratón, se muestra el menú mediante una transición realizada con CSS al añadirle la clase `open`, que tiene su propio formateo. Además, ajusta la clase `open` del cuerpo, alternando entre el estado abierto y cerrado, lo cual permite activar y desactivar estilos dinámicamente. El botón de cierre tiene un funcionamiento similar, por lo que no se incluye.



Figura 4.5: Menú de móvil cerrado

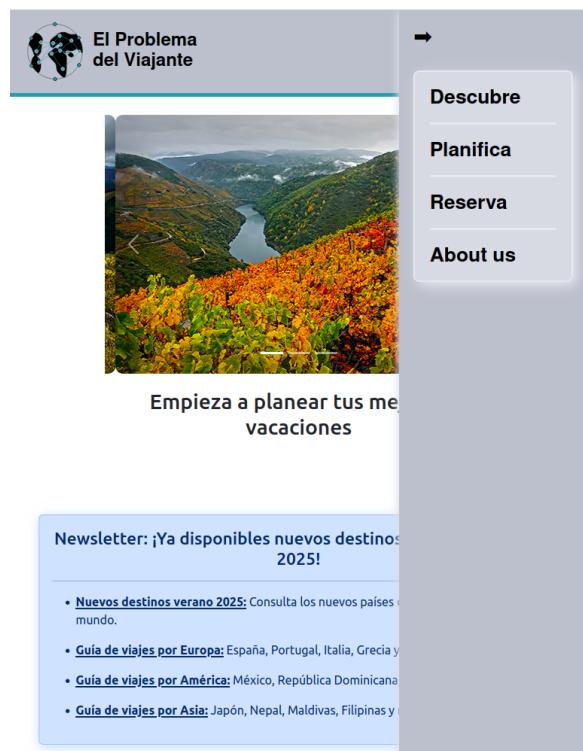


Figura 4.6: Menú de móvil abierto

En las dos imágenes anteriores se muestra el funcionamiento del menú de móvil aunque, debido a que las fotos son estáticas, se pierde la información relativa a la responsividad de los botones y las transiciones creadas con CSS que otorgan una mejor experiencia de usuario.

```

1 | function checkScreenSize() {
2 |   menuLinks.style.display = window.innerWidth > 900 ? "flex" : "none";
3 |   if (window.innerWidth > 900) menuSidebar.classList.remove("open");
4 |
5 |
6 | window.addEventListener("resize", checkScreenSize);

```

Otro evento incluido en este script está relacionado con los cambios en el tamaño de la pantalla. Cuando se redimensiona la pantalla, se comprueba el ancho de esta, lo cual permite saber cuándo mostrar y ocultar el menú de móvil. Además, en caso de que esté abierta la barra lateral (`menuLinks`) cuando se amplíe la pantalla a un ancho superior a 900px, esta se cierra automáticamente, con una transición suave creada con CSS.

Este efecto no se puede mostrar con nuevas imágenes ya que es totalmente dinámico, y ya se han mostrado en múltiples ocasiones fotos con el menú de navegación de ordenador y de móvil, por lo que no se incluyen de nuevo.

En relación a la respuesta a eventos, destacan sobre todo los filtros aplicados en la página de Reserva, pues esta se programó con un enfoque orientado a eventos, de forma que los cambios en la página ocurren a medida que el usuario interactúa con los distintos elementos de esta. Se muestra a continuación una parte del código encargado de la respuesta a algunos de estos eventos. Cabe destacar que este código emplea jQuery y muchos elementos de JES6, en los que se pondrá más énfasis en la siguiente sección, pues simplifican mucho la sintaxis de acceso a los elementos del DOM.

```

1 | $('.type-filter').on('click', function(e) {
2 |   e.preventDefault();

```

```

3   currentFilters.type = $(this).data('type');
4   $('.type-filter').removeClass('active');
5   $(this).addClass('active');
6   applyFilters();
7 });
8
9 $('#destination-filter').on('input', function() {
10   currentFilters.destination = $(this).val();
11   applyFilters();
12 });
13
14 $('.service-checkbox').on('change', function() {
15   const service = $(this).data('service');
16   currentFilters.services = $('.service-checkbox:checked')
17     .map((i, el) => $(el).data('service')).get();
18   applyFilters();
19 });

```

Los eventos a los que se responde en los ejemplos que se recogen aquí son:

- **click**, que ocurrirá cuando se pulse en algún elemento de la barra de navegación secundaria de la página Reserva, lo que producirá que se filtren los servicios de las tarjetas por tipo, además de marcar el tipo marcado como activo, lo que cambiará su estilo.
- **input**, que se lanzará cada vez que se modifique el texto en el filtro de destino de los filtros principales, y lo mismo ocurrirá con el número de viajantes (aunque no se muestra en el código aquí recogido por brevedad). Así, cada vez que se modifique el campo, se filtrarán las tarjetas de servicios a aquellas cuyo destino contenga parte del campo introducido y una capacidad suficiente para el número de viajantes. Gracias a este evento, no es necesario pulsar un botón ni ninguna otra acción especial para aplicar los filtros, sino que estos se irán aplicando a medida que el usuario escriba.
- **change**, que se activará cada vez que se seleccione o deseccione un *checkbox* de los filtros secundarios en la barra lateral. Si un servicio está marcado, se filtrarán las tarjetas para solo mostrar aquellas que lo incluyan. Además, el resto de filtros sobre los servicios también cambiarán para solamente mostrar posibles filtros que apliquen sobre las tarjetas que se estén visualizando en ese momento.

Se muestra a continuación el efecto que produce la aplicación de estos filtros en respuesta a los eventos propiciados por el usuario.

Antes de aplicar filtros:

El sitio web 'El Problema del Viajante' muestra una lista de reservas para diferentes destinos:

- Hotel La Montaña ******: México - 4 personas. Precio: Desde 150€. Botón 'Reservar'.
- Vuelo Barcelona-Madrid *******: España - 2 personas. Precio: Desde 89€. Botón 'Reservar'.
- Tour Gastronómico Roma *******: Italia - 8 personas. Precio: Desde 200€. Botón 'Reservar'.

A la izquierda, un cuadro resaltado con un cuadro rojo muestra:

- Puntuación mínima**: 5 estrellas.
- Servicios** (checkboxes):
 - Wifi
 - Desayuno
 - Piscina
 - Equipaje
 - Guía
 - Degustación
 - Seguro

Después de aplicar filtros:

La lista de reservas se ha filtrado para mostrar solo el Hotel La Montaña:

- Hotel La Montaña ******: México - 4 personas. Precio: Desde 150€. Botón 'Reservar'.

A la izquierda, el cuadro resaltado muestra los mismos criterios de filtro, pero con cambios en las selecciones:

- Puntuación mínima**: 5 estrellas.
- Servicios** (checkboxes):
 - Wifi (selecciónada)
 - Desayuno
 - Piscina (selecciónada)

En la parte inferior, se incluyen los términos y condiciones:

- © El Problema del Viajante S.L.
- Registrado en España, Santiago de Compostela | CIF: X123456789
- Términos de uso | Política de privacidad

Figura 4.7: Comparativa antes y después de aplicar filtros en la página Reserva

Como se puede ver, de la gran cantidad de tarjetas de servicios que se mostraban al principio, tras filtrar el tipo de servicio a *Alojamiento*, filtrar por destino con *México*, el número de viajeros a 3, la puntuación mínima a 4 estrellas y seleccionar como servicios necesarios *Wifi* y *Piscina*, obtenemos una única tarjeta de destino que cumpla con nuestras especificaciones. Además, se puede ver cómo la lista de filtros de servicios en la barra lateral también se redujo para solo incluir aquellos posibles filtros relevantes, es decir, los que producirían alguna coincidencia.

4.2. Uso de jQuery y JES6

Para ilustrar el uso de objetos jQuery, vale por ejemplo el código de la sección anterior relacionado con la respuesta a eventos para filtros, pero a continuación mostramos otro distinto que también emplea esta librería. Este efecto es el relacionado con la interactividad entre las tarjetas de destinos y el mapa.

```

1 let card = $('<div>', { class: 'destination-card' });
2 let marker = L.marker([latitude, longitude], { riseOnHover: true }).addTo(
    markerGroup).bindPopup(name);
3
4 /* Guardar referencias para hacerlo interactivo */
5 card.data('marker', marker);
6 marker.card = card;
7
8 /* Eventos en la carta */
9 card.hover(
10     function() {
11         marker.openPopup();
12         marker.setZIndexOffset(1000);
13         map.setView([latitude, longitude], 5);
14     },
15     function() {
16         marker.closePopup();
17         marker.setZIndexOffset(0);
18     }
19 );
20 card.on('click', function() {
21     map.setView([latitude, longitude], 12);
22 });
23
24 /* Eventos en el mapa */
25 marker.on('mouseover', function() {
26     this.openPopup();
27     this.setZIndexOffset(1000);
28     this.card.addClass('highlight');
29 });
30 marker.on('mouseout', function() {
31     this.closePopup();
32     this.setZIndexOffset(0);
33     this.card.removeClass('highlight');
34 });
35 marker.on('click', function() {
36     map.setView([latitude, longitude], 12);
37 });

```

En este código, se genera mediante jQuery una carta de destino dinámicamente (esa parte se omite) y un marcador en el mapa mediante la librería Leaflet ya mencionada. Se guardan referencias cruzadas entre ambos elementos y se añaden eventos en la carta de tal forma que al posicionarse sobre ella o pulsarla se haga zoom en el mapa en la ubicación correspondiente.

De igual forma, cuando se pase el botón sobre el marcador en el mapa o se pulse, se destacará la carta asociada. Este comportamiento se puede apreciar en el siguiente pantallazo (aunque el comportamiento no se aprecia del todo bien debido a las limitaciones intrínsecas de intentar hacer una captura de pantalla a un comportamiento dinámico).

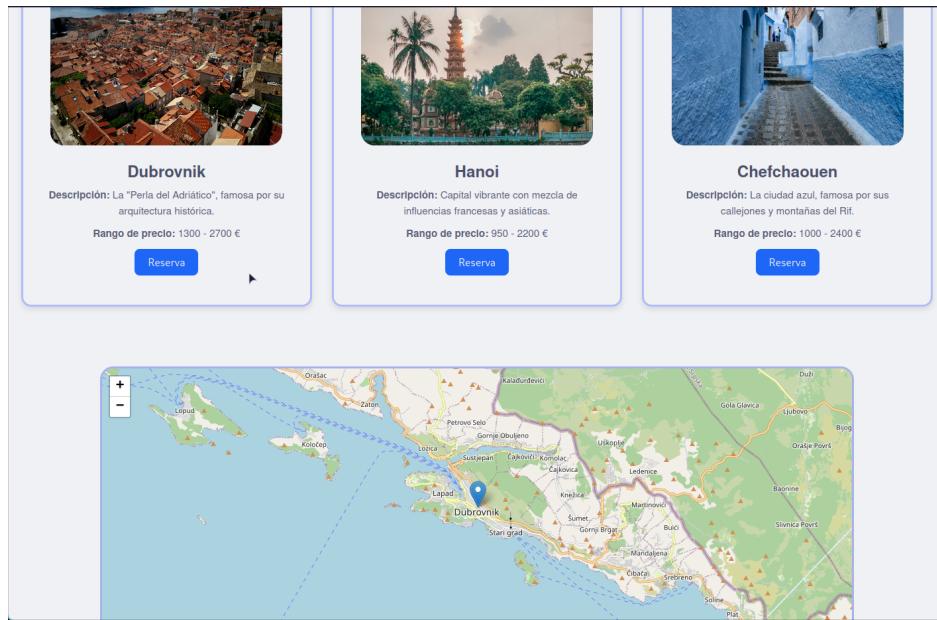


Figura 4.8: Interactividad en el mapa de Descubre gracias a jQuery

En relación con JES6, este se utiliza constatemente en los códigos ya mostrados, por ejemplo en el uso de `let` y `const` en la declaración de variables, la utilización de funciones arrow o el uso de `async/await` en la lectura de archivos.

Por ejemplo, se puede ver en el siguiente código, usado para la actualización de los filtros de servicios relevantes en la página Reserva, un uso constante de elementos de JES6, entremezclado también con jQuery, por ejemplo, en el uso de funciones arrow para filtrar los servicios utilizando un estilo funcional, o el uso de `const` en lugar de `var` para declarar datos que no van a variar en toda la función.

```
1 | function updateServiceFilters(visibleServices) {
2 |   const allServices = visibleServices
3 |     .flatMap(service => service.servicios)
4 |     .filter((value, index, self) => self.indexOf(value) === index); // Servicios unicos
5 |
6 |   const container = $('.checkbox-group');
7 |   container.empty();
8 |
9 |   allServices.forEach(service => {
10 |     const label = service.charAt(0).toUpperCase() + service.slice(1).toLowerCase();
11 |     container.append(`\n      <label>\n        <input type="checkbox"\n          class="service-checkbox"\n          data-service="${service}"\n          ${currentFilters.services.includes(service) ? 'checked' : ''}>\n        ${label}\n      </label>\n    `);
12 |   });
13 | }
```

```

20     });
21
22     // Re-asignar eventos
23     $('.service-checkbox').off('change').on('change', function() {
24         currentFilters.services = $('.service-checkbox:checked')
25             .map((i, el) => $(el).data('service')).get();
26         applyFilters();
27     });
28 }

```

4.3. Carga de contenido

En esta sección se mostrarán dos posibles formas de cargar contenido, una utilizando API Fetch para leer un archivo JSON, en el que se encuentran los eventos que luego se muestran en el calendario, y otra que usa AJAX mediante jQuery para cargar un archivo XML a partir del que generar más fácilmente las tarjetas de destinos en la página Descubre y permitir su interacción dinámica con el mapa de esa página.

4.3.1. Formato XML

Para cargar el archivo XML con la información de las tarjetas de destinos, se utilizó la API de AJAX integrada en jQuery, que resulta muy moderna y fácil de utilizar. Se optó por trasladar la estructura y contenido de las tarjetas desde el HTML de la página Descubre a un archivo XML, lo que permite desacoplar los datos de los destinos de la estructura y presentación del documento, y facilita la interactividad con el mapa de esa misma página.

Esta decisión llevó de forma natural a adoptar un diseño *Single Page Application* (SPA), de forma que lo que antes eran cinco páginas HTML distintas con cada una de las variantes de la página Descubre, accesibles mediante la barra de navegación secundaria, ahora se transformaron en una página única, y la barra de navegación secundaria pasa a utilizarse ahora para indicarle al código JavaScript de qué archivo XML obtener los datos. Este diseño es mucho más limpio y fácil de mantener, evitando tener que propagar los cambios a múltiples páginas cada vez que se quiera actualizar la web; es también más eficiente, pues en vez de recargar toda la página, incluido el mapa que es costoso, solo se tienen que recargar las tarjetas y los marcadores. Una estrategia similar se empleó en la página Reserva, donde antes los filtros de tipo de servicio de la barra de navegación secundaria estaban pensados para enlazar páginas distintas y se transformaron en simplemente un filtro más, agilizando el desarrollo y facilitando el mantenimiento.

A continuación se muestra el formato utilizado para una tarjeta de destino en XML.

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <destinations>
3      <destination>
4          <image>
5              <src>descubre/1-Greece.jpg</src>
6              <alt>Grecia</alt>
7              <width>350</width>
8              <height>250</height>
9          </image>
10         <details>
11             <name>Atenas</name>
12             <description>
13                 Cuna de la civilización occidental. Descubre la Acropolis,
14                 el Partenón y platos tradicionales como la moussaka.

```

```

15         </description>
16         <price>
17             <lower>1800</lower>
18             <upper>3200</upper>
19         </price>
20     </details>
21     <location>
22         <latitude>37.9838</latitude>
23         <longitude>23.7275</longitude>
24     </location>
25   </destination>
26   <!-- ... -->
27 </destinations>
```

Este archivo se carga mediante el siguiente código que emplea jQuery, y a partir de él se crea cada una de las tarjetas que se muestran en la página.

```

1 $.ajax({
2     url: rootPath + "/data/descubre/" + xmlFile,
3     type: "GET",
4     dataType: "xml",
5     success: function(xml) {
6         /* Eliminar el contenido previo */
7         $('#destinations').empty();
8
9         /* Procesar cada destino del archivo XML */
10        $(xml).find('destination').each(function() {
11            let imageSrc = $(this).find('image > src').text().trim();
12            let imageAlt = $(this).find('image > alt').text().trim();
13            let width = $(this).find('image > width').text().trim();
14            let height = $(this).find('image > height').text().trim();
15            let name = $(this).find('details > name').text().trim();
16            let description = $(this).find('details > description').text()
17                .trim();
18            let lowerPrice = $(this).find('details > price > lower').text
19                .trim();
20            let upperPrice = $(this).find('details > price > upper').text
21                .trim();
22            let latitude = parseFloat($(this).find('location > latitude')
23                .text().trim());
24            let longitude = parseFloat($(this).find('location > longitude')
25                .text().trim());
26
27        // ...
28    });
29 },
30 error: function() {
31     console.error('Error cargando el archivo XML');
32 }
33});
```

Como vemos, a través del método de jQuery `$.ajax` se puede procesar un archivo XML indicando su dirección, el tipo de método HTTP empleado para obtenerlo, el tipo del archivo, y las funciones a ejecutar en caso de éxito y de fracaso. En la función de éxito, que es donde va la mayor parte del código, se puede acceder al contenido del archivo XML utilizando la sintaxis estándar de jQuery. Así, mediante el método `find()`, se puede acceder a los distintos campos del XML por su nombre, que en este caso guardamos en sendas variables para posteriormente poder construir las tarjetas dinámicamente.

El efecto obtenido gracias a este código es el de una página SPA altamente interactiva y fácilmente modificable en cuanto a contenido. Para ilustrarlo, se muestra el cambio en la página tras pulsar en un elemento de la barra de navegación.

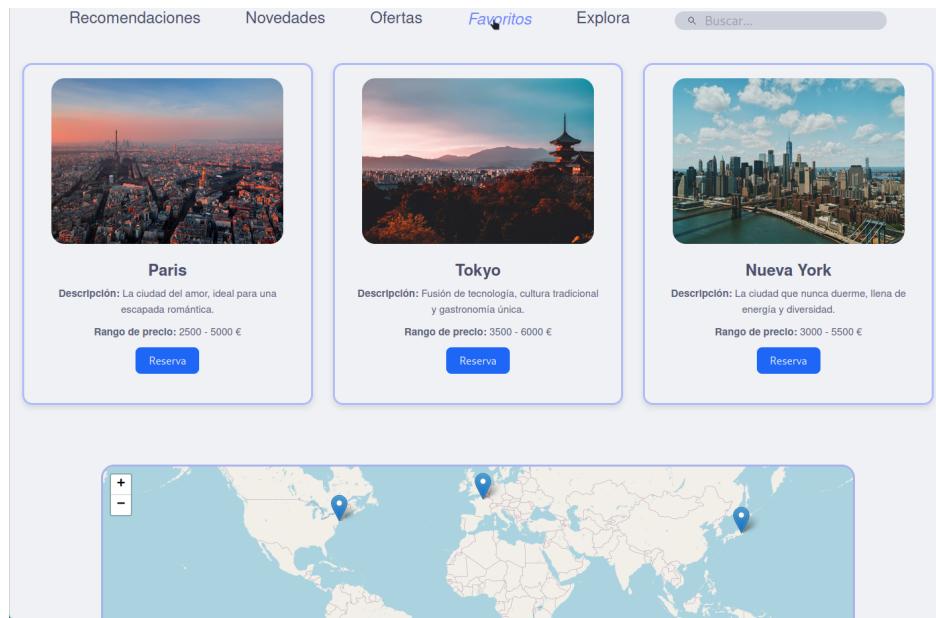


Figura 4.9: Cambio de las tarjetas y marcadores en Descubre a partir de datos en XML

4.3.2. Formato JSON

Como se ha mencionado, se carga un archivo JSON sencillo que incluye los próximos eventos relevantes para el usuario, los cuales se muestran en el calendario señalando los días en los que ocurren, y junto a él en una lista donde aparece la fecha en formato de texto y el nombre del evento. Se utiliza también el formato JSON para cargar la información de las tarjetas de servicios en la página Reserva, pero como esta funcionalidad es análoga a la anterior, explicaremos en detalle la empleada para el calendario. El formato utilizado en el archivo JSON es

```

1 { "events": [
2   { "date": "fecha1", "title": "Evento1" },
3   { "date": "fecha2", "title": "Evento2" },
4   { ... }
5 ]}
```

Este archivo se carga en el script del calendario de la siguiente forma.

```

1 async function loadEvents() {
2   try {
3     const response = await fetch(rootPath + "/data/events.json", {
4       cache: "no-cache" });
5     const data = await response.json();
6     /* ... Gestión de errores, no se incluye ... */
7     return data.events;
8   } catch (error) {
9     console.error("Error cargando eventos:", error);
10    return [];
11  }
12}
13 loadEvents().then(function (events) {
14   initCalendar(events);
15});
```

Este código lleva al cliente el archivo indicado en la función `fetch()`, y lo lee, devolviendo un array con los eventos en caso de éxito, es decir, si el archivo ha sido procesado correctamente.

En ese caso, se ejecuta la función `initCalendar()`, la cual no se incluye ya que principalmente se encarga de configurar el calendario para mostrarlo de forma adecuada (en español, que el primer día de la semana sea el lunes, etc.). Sí que es interesante comentar una de las funciones a las que llama `initCalendar()`, que se muestra a continuación.

```

1 function highlightEvents(instance, events) {
2     setTimeout(function () {
3         let days = instance.calendarContainer.querySelectorAll(".flatpickr-day");
4
5         days.forEach(function (day) {
6             let date = formatLocalDate(day.dateObj); // Formato YYYY-MM-DD
7             // Comprobar si la fecha esta en la lista de eventos
8             if (markedDates.includes(date)) {
9                 day.classList.add("event-day");
10            }
11        });
12    }, 100);
13 }

```

Esta función se encarga de resaltar los días en los que hay algún evento. Para ello, se accede al DOM para obtener todos los nodos que son un día, los cuales fueron creados al instanciar el calendario, y para cada uno se comprueba si existe algún evento cuya fecha coincida con la del propio día. En caso afirmativo, se añade la clase `event-day` al mencionado día, la cual lo muestra de color rojo. Los días marcados también se añaden en una lista al lado del calendario junto con el nombre del evento, aunque este código no se muestra debido a su simplicidad.



Figura 4.10: Calendario sin eventos



Figura 4.11: Calendario con eventos

En las dos imágenes anteriores se muestran las diferencias entre el calendario proporcionado por flatpickr y las modificaciones añadidas con JavaScript. De nuevo, hay detalles que no se muestran en las imágenes como la responsividad al pasar el ratón por encima de los días del calendario, o por encima de la lista de eventos.

Capítulo 5

Anexo

5.1. Tormenta de ideas

Todos los elementos obtenidos durante la tormenta de ideas son:

- Presentación de la página
- Destinos de moda/recomendados
- Exploración (lugares menos conocidos, o aleatorios)
- About us: quiénes somos, nuestra historia, qué nos diferencia
- Búsqueda de hoteles
- Búsqueda de medios de transporte (avión, tren, autobús, etc.)
- Lugares de visita, ocio, restauración, etc.
- Alquiler de coches
- Experiencias exclusivas
- Calendario para la planificación del viaje
- Mapa de lugares favoritos
- Newsletter con novedades
- FAQ
- Mecanismos de reserva con seguridad
- Seguro de viajes
- Consejos generales para viajes al extranjero
- Nuestro sistema de puntuación
- Opiniones de expertos
- Certificados de calidad, premios
- Redes sociales
- Términos de uso, política de cookies, política de privacidad

5.2. Cambios

En esta sección se incluye una lista de cambios realizados a lo largo del proyecto, que difieren respecto al diseño original que se planteó. Cada uno incluye una breve descripción y una justificación de por qué se ha hecho.

5.2.1. Diseño Página principal

En la página principal se ha cambiado la zona principal en la que se pretendían mostrar dos imágenes en el diseño inicial. Después de investigar todas las posibilidades que ofrece la librería *Bootstrap*, se decidió incorporar un carrusel de fotos, que dota a la página de un aspecto más profesional y trabajado.

Otro cambio menos relevante es que ha optado por un color azul para el fondo de la sección de noticias, el cual es más suave que el color flamenco inicial, y encaja mejor en el diseño de la página. El apartado de valoraciones se mantiene sin cambios. Cabe destacar también el cambio en el logo a una versión más minimalista, creado vectorialmente con SVG, que permite un uso más eficiente del mismo a la hora de incorporarlo a la pestaña de navegación y en otros lugares de la web.

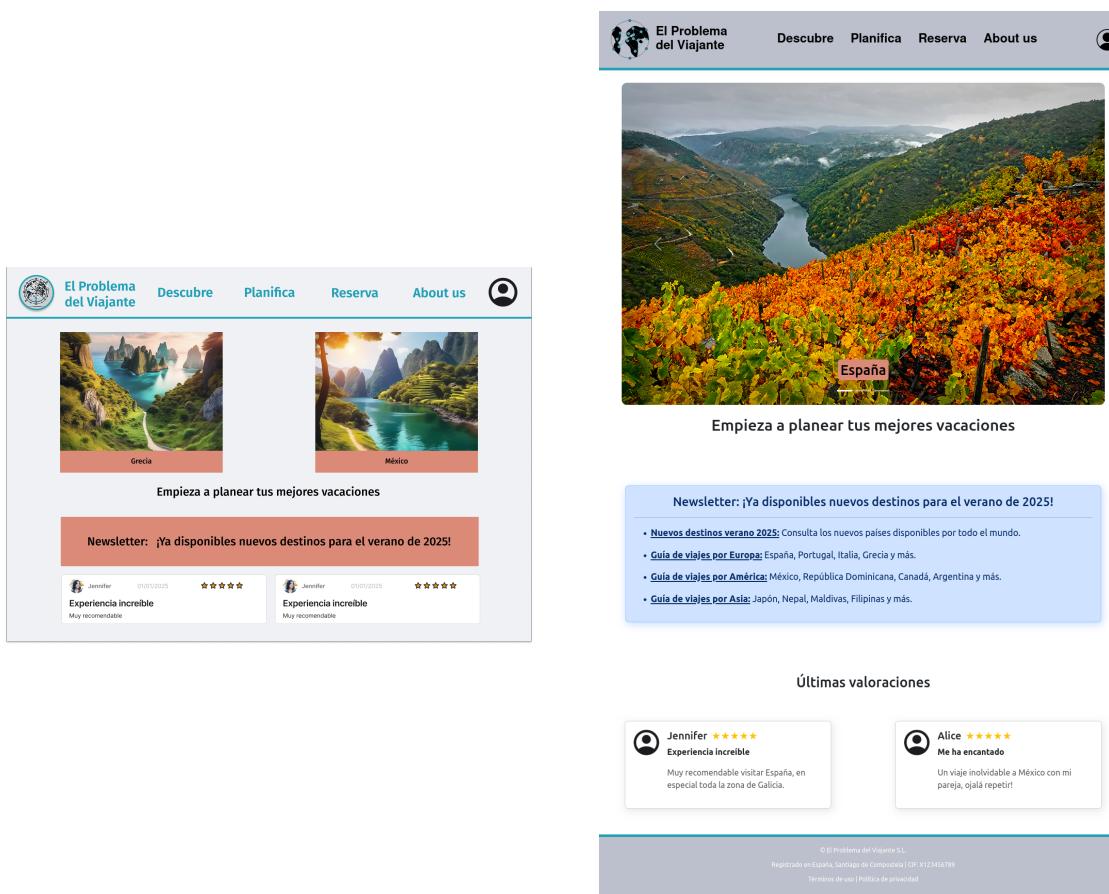


Figura 5.1: Comparativa diseño inicial y final de la página Principal

5.2.2. Diseño Descubre

En la página Descubre se mantuvo un diseño bastante similar al que estaba planeado desde un principio, con cambios estéticos que afectan principalmente a un diseño menos recargado para las tarjetas de destino. Así, estas pasaron de tener un fondo colorido a un blanco neutro, con un ligero borde que se destaca al interactuar con ellas. También se optó por el uso de bordes más redondeados y se cambió ligeramente la disposición de los componentes de la tarjeta, consiguiendo así una estética más pulida y un diseño más coherente con el resto del sitio web. Por supuesto, también hay cambios apreciables con respecto a la apariencia del mapa, pero estos ya eran esperados, pues en el diseño inicial se utilizó una imagen de stock que ya se esperaba que fuera sustituida por un mapa realista e interactivo con la apariencia estándar que estos tienen.

Otro cambio significativo, ya no en el diseño sino en la estructura, es el ya mencionado uso de un diseño *Single Page Application*. Así, de un diseño inicial en el que cada uno de los subapartados de la página se correspondería con un archivo HTML diferente, se pasó a una única página en el que las tarjetas y marcadores se actualizarían a partir de datos leídos de un archivo XML. Este diseño resulta mucho más escalable, eficiente y sencillo de mantener y desarrollar, desacoplando los datos de la estructura y la apariencia.

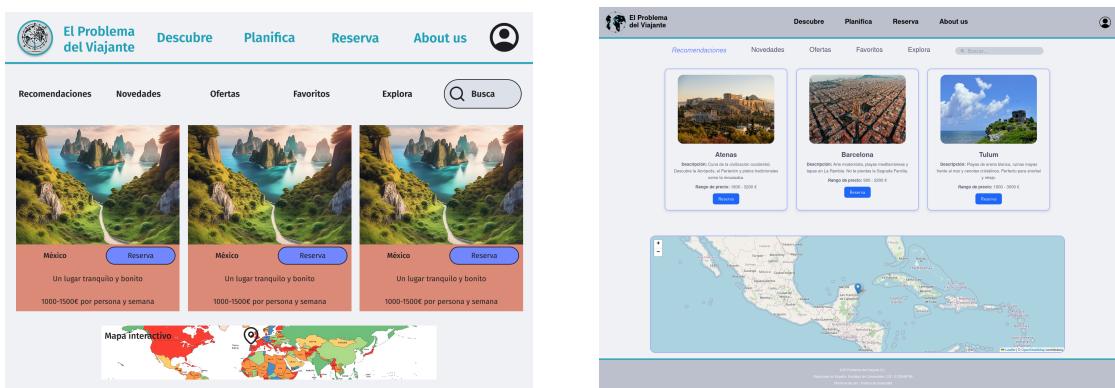


Figura 5.2: Comparativa diseño inicial y final de la página Descubre

5.2.3. Diseño Planifica

En la página Planifica, se ha optado por un diseño ligeramente menos recargado, en el que los recuadros mostrados no tienen el fondo de colores diferentes, sino que solo el borde, delimitando así cada zona. También se ha añadido una línea que separa la sección de Viajeros de la Agenda, y se han destacado los títulos de las secciones de Consejos generales y Reserva con Seguridad, para mostrarlos de una forma más elegante.

La diferencia más importante respecto al alcance del diseño inicial, aunque no tanto en el aspecto, se encuentra en el calendario. Se ha eliminado la entrada de una fecha mediante teclado, optando por realizar todas las acciones desde el propio calendario. Y no ha sido posible realizar todo lo que se pretendía de sincronizar el calendario con la cuenta del usuario ni que el calendario fuese compartido por varios usuarios, debido a las limitaciones de no tener un servidor que aloje la página web. Es por ello que se ha optado por limitarse a mostrar ciertos eventos organizados por la empresa ficticia, y que estos aparezcan marcados en el calendario.

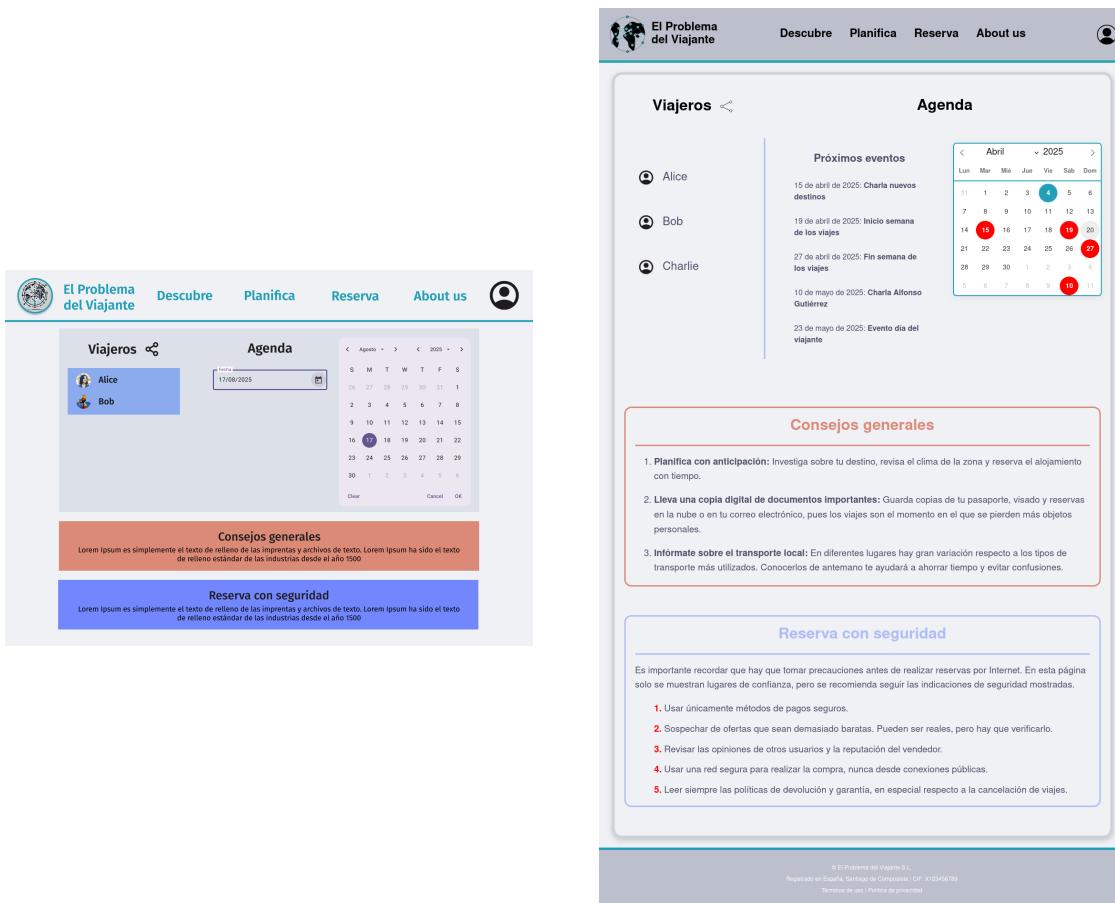


Figura 5.3: Comparativa diseño inicial y final de la página Planifica

5.2.4. Diseño Reserva

Para la página Reserva, los cambios son sustanciales. En primer lugar, como ocurría en Descubre, de un diseño inicial que utilizaba varias páginas para cada uno de los tipos de servicios, se pasó a un diseño SPA en el que la barra de navegación secundaria se transformó en una capa adicional de filtros. Por este motivo, se incluyó la opción *Todos* para poder optar por no filtrar por tipo de servicio.

En lo que respecta a los filtros principales por destino, número de viajeros y fechas, se decidió eliminar la opción de búsqueda, que desequilibraba el diseño de la página y se consideró prescindible al disponer ya de numerosos filtros. En cuanto a la barra lateral de filtros secundarios, se mantuvo su propósito y su estructura, pero se adaptó su diseño por uno mucho menos recargado, en línea con el diseño del resto del sitio web, pasando a usar un fondo neutro y reservando el color para los márgenes y los títulos; también se adaptó el estilo de las estrellas de puntuación y los *checkbox* para adaptarlo a los colores que se utilizaron para la interactividad en el resto de páginas.

Otro cambio muy destacable es el referente a las tarjetas de servicios. Como en todas las páginas, se optó por un diseño más minimalista y simplificado, eliminando las imágenes para no saturar la página y poder apreciar mejor el efecto de los filtros que se estaban aplicando, además de facilitar su aplicación en tiempo real, pues se ahorra el coste de renderizar las imágenes. En general, como se viene mencionando, se optó por un diseño más limpio y pulido que proporcionase una estética más profesional y cohesionada a toda la página.

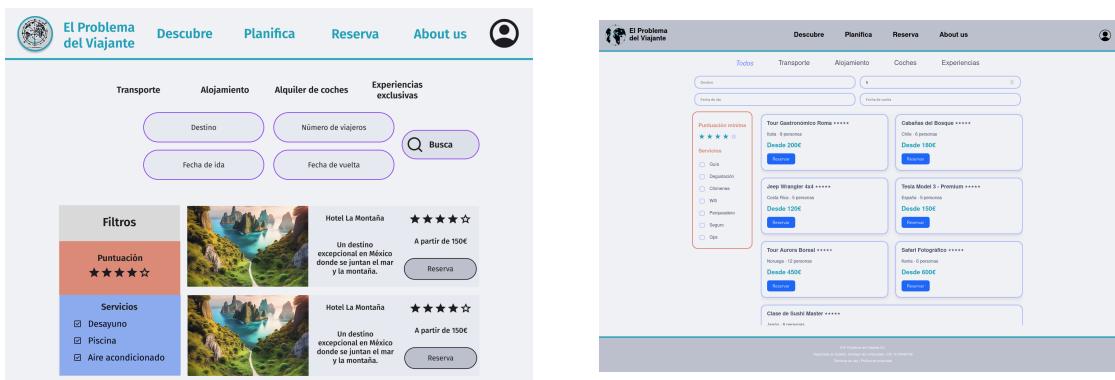


Figura 5.4: Comparativa diseño inicial y final de la página Reserva

5.2.5. Diseño About us

De forma similar a lo que ocurría con la pestaña de Planifica, en la página About us se han eliminado los fondos de colores de los recuadros, dejando únicamente el borde que separa cada sección, aunque el diseño se ha mantenido semejante. Los colores de los bordes también se han cambiado ligeramente, con el objetivo de mejorar el aspecto de la página. Finalmente, el borde de la sección de Preguntas frecuentes es naranja, y el de Contacto verde, lo cual dota a la página de un aspecto más vivo.

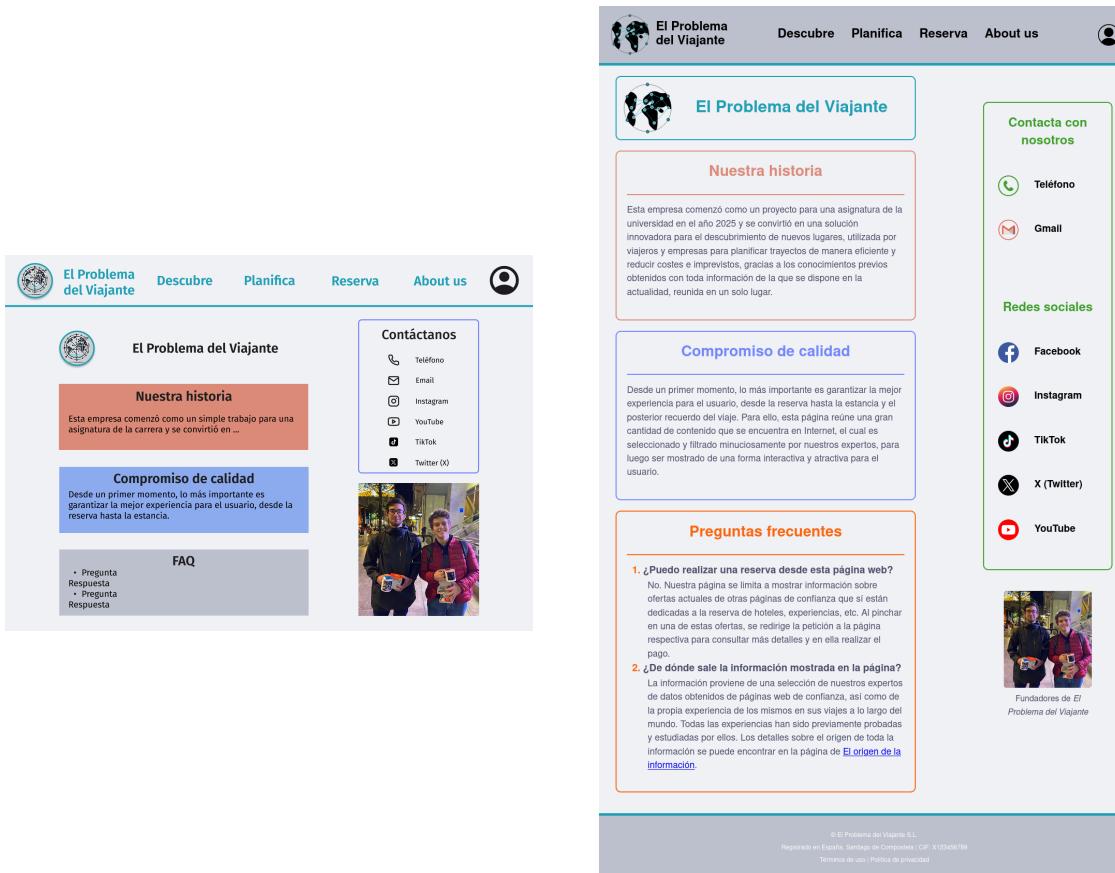


Figura 5.5: Comparativa diseño inicial y final de la página About us

5.3. Estructura de ficheros

La estructura de ficheros y directorios general se ha mantenido a lo largo del proyecto, aunque se ha añadido algún subdirectorio a los que se mencionan en la sección 1.7. Concretamente, cada una de las cinco páginas principales tiene su propio directorio dentro de /html e /images (para las que emplean imágenes), con el objetivo de facilitar la gestión de grandes cantidades de contenido.

Destaca también la creación del directorio /data, para guardar los archivos XML y JSON utilizados en la generación dinámica del contenido de las páginas, que también se estructuraron en su propio subdirectorio en el caso de los archivos XML de la página Descubre para imitar la estructura original que tenía originalmente esta página y que se conseguía antes a través de archivos HTML.

El directorio /documentation sigue presente para albergar todo lo relacionado con la redacción de esta memoria, pero se excluye aquí y en la entrega pues no aporta nada interesante y su único propósito es servir de soporte para la creación de este documento. Se muestra a continuación la estructura de directorios y ficheros final del proyecto.



Figura 5.6: Estructura final de ficheros

Bibliografía

- [1] Experience travel group
- [2] Booking
- [3] Tripadvisor
- [4] Skyscanner
- [5] Google Calendar
- [6] W3C (Estándares HTML5 y revisor de código HTML, CSS y JavaScript)
- [7] W3Schools (Guías generales de uso de HTML y CSS)
- [8] Campus Virtual de la asignatura
- [9] Bootstrap (Librería de CSS y JavaScript)
- [10] Flatpickr (Calendario responsivo con CSS y JavaScript)
- [11] Leaflet (Mapa interactivo con CSS y JavaScript)