



Propuesta de Diseño Web

Boundless: Desarrolladora

de videojuegos

Vivel Couso, Ainhoa

Profesor: García Tahoces, Pablo

Diseño de Aplicaciones Web
Grado en Ingeniería Informática
Escola Técnica Superior de Enxeñaría (ETSE)
Universidade de Santiago de Compostela (USC)

Septiembre 2020

ÍNDICE

1. Abstract	4
2. Presentación de la práctica	4
3. Inventario de contenido	4
4. Arquitectura de Información	6
5. Diseño Gráfico de Interfaces	8
5.1. Sketching	8
5.2. Wireframe	12
5.3. Mockup	19
6. Casos de Uso	24
7. Mapa Web	26
8. Storyboard	27
9. Estructura de Ficheros	28
10. HTML y Mapas de Etiquetas	29
11. CSS y Mapas de clases	34
12. Javascript	38
12.1. Slideshow promocional	38
12.2. Highlight de juegos	42
12.3. Alerta de redirección	44
13. AJAX	46
13.1. XML	47
13.2. JSON	50
14. JQuery	52
14.1. Highlight de juegos	52
14.2. Menú desplegable	55
15. JSP, Java Beans, EL, JSTL y JDBC	56
15.1. Implementación de la base de datos	57
15.2. Login y registro	60
15.3. Tienda	71
15.4. Carro	74
15.5. Pago	79
15.6. Listado de consultas de la base de datos	83

15.6. Información de contacto	85
16. Control de cambios	86
16.1. HTML	86
15.2. CSS	87
16.3. JavaScript	88
16.4. AJAX	88
16.5. JQuery	89
16.6. JSP, Java Beans, EL, JSTL y JDBC	89
17. Referencias	90
17.1. Bibliografía	90
17.2. Recursos complementarios	91

1. Abstract

El presente escrito analiza las necesidades empresariales de Boundless, una sociedad corporativa centrada en el desarrollo de videojuegos, para proponerle un diseño adecuado y profesional para su página web. Así mismo, documenta detalladamente todo el proceso de diseño de la web durante las diferentes fases del mismo, justificando cada una de las decisiones tomadas.

2. Presentación de la práctica

El objetivo de la primera práctica^[1] propuesta es diseñar un sitio web para una empresa ficticia. Este diseño será posteriormente utilizado como base para el resto de prácticas y se le irán añadiendo nuevas funcionalidades a medida que estas sean especificadas. Como se ha recomendado, para la realización del diseño se ha empleado como guía el documento funcional^[2] proporcionado.

En primer lugar, fue necesario establecer un tema principal alrededor del cual se elaboraría la web. Para ello se llevó a cabo una *lluvia de ideas*. Tras considerar varias opciones (entre la que destacan una pastelería, el portafolio de un grafitero, una agencia de viajes y una firma de gafas), se escogió como tema del proyecto una empresa inexistente cuya actividad se basa en el desarrollo y distribución de videojuegos.

Así, se concluye que la empresa *Boundless*, una corporación de nueva creación cuya actividad económica se desenvuelve en el sector del entretenimiento electrónico, ha solicitado una propuesta de diseño para su página web. El primer paso para elaborar un buen diseño consiste en llevar a cabo un *brainstorm* con la intención de encontrar una serie de elementos centrales que definieran el contenido de la web. Dichos elementos y su relación forman el **Inventario de Contenido**^{[3][4]}.

3. Inventario de contenido

El Inventario de Contenido, como se especifica en el guión, *debe incluir los ítems relativos al sitio web que están orientados hacia la audiencia de la página*. Esto quiere decir que en el mismo se deben incluir todos los elementos que un usuario promedio consideraría necesarios o esperaría encontrar en dicha página. Como la empresa se especializa en el sector de los videojuegos, se ha decidido que deben existir secciones relacionadas con información sobre la actividad de la corporativa, formas de contacto, productos, y noticias relacionadas. Adicionalmente, se ha investigado el contenido de las webs de empresas competidoras en su mercado. Las páginas webs consultadas como referencia para determinar el alcance del diseño y mejorarlo son *Electronic Arts*^[5], *Cygames*^[6], *Nintendo*^[7], *Sony Computer Entertainment*^[8] y *Ubisoft*^[9], entre otras. De este modo, se ha obtenido el siguiente Inventario de Contenido¹.

¹ Aunque la página web en condiciones normales debería tener una tienda online a disposición del usuario, al especificarse en el guión de la práctica que la empresa no podía tratarse de una tienda este punto se ha obviado.

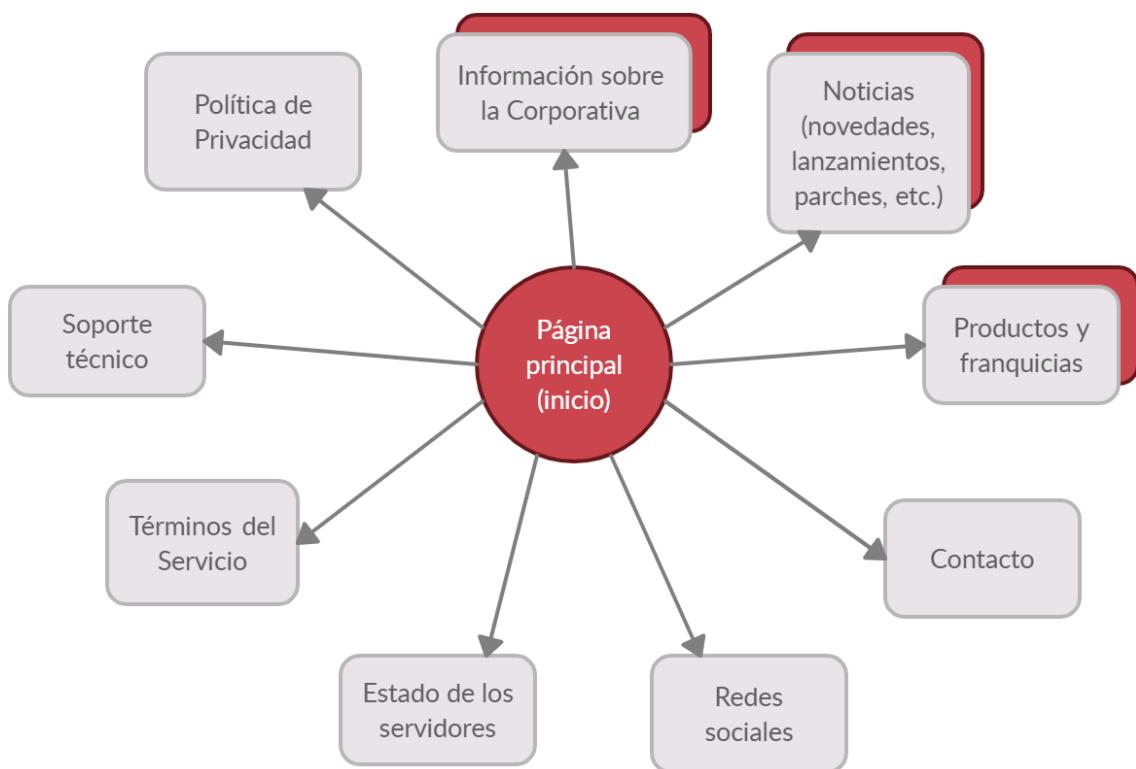


Figura I: Inventario de contenido de *Boundless*

La figura I es el inventario de contenidos propuesto para la página web de *Boundless*. Este contiene todos los elementos a los que un usuario podría acceder. Más detalladamente:

- **Página principal**^[10]. La página de inicio o portada es la parte central de la web. Esta carga cuando se intenta acceder al sitio y desde ella se puede acceder al resto de contenidos. Su finalidad es presentar la web al usuario y brindarle acceso a todo su contenido.
- **Información sobre la Corporativa**. En esta sección del sitio web se incluyen datos diversos sobre la empresa como su detalles de su fundación, localización, objetivos, reconocimientos, empleados, trayectoria y afiliados.
- **Noticias**. Contenido acerca de nuevos lanzamientos, actualizaciones, planes de programación o cualquier otra novedad que le pueda interesar a la audiencia. Toda la información de esta sección está relacionada con su actividad empresarial.
- **Productos y franquicias**. Este elemento engloba toda la información sobre los proyectos de la empresa que han sido finalizados o anunciados. Así, se podrá acceder a datos específicos de cada videojuego.
- **Contacto**. Sección que recoge los diferentes métodos para contactar con algún empleado de la empresa.

- **Redes Sociales.** Sección que recopila todas las redes sociales en las que está disponible la empresa de forma oficial.
- **Estado de los servidores.** Sección que recoge información de la disponibilidad de los servidores de sus juegos, así como su afluencia y reportes de errores relativos a los mismos.
- **Privacidad^[11].** Esta sección incluye información legal acerca de cómo la empresa organización retiene, procesa y maneja los datos del usuario. Las políticas de privacidad se consideran un contrato en el cual la organización responsable promete mantener la información personal del usuario.
- **Términos del Servicio^[12].** Esta sección incluye información legal acerca de las condiciones de uso de sus productos. Los Términos del Servicio especifican las condiciones que debe cumplir y asumir el usuario para poder acceder a los servicios de la empresa.
- **Soporte.** Proporciona asistencia a los usuarios que tengan problemas al utilizar un producto de la empresa.

Los ítems descritos generalizan todo el contenido que se debe incluir en el sitio web. El siguiente paso es descomponerlos y establecer una jerarquía entre ellos. Así, se obtendrá la **Arquitectura de Información** del sitio.

4. Arquitectura de Información

La Arquitectura de Información^[12] se describe como *la disciplina y arte encargada del estudio, análisis, organización, disposición y estructuración de la información en espacios*. Esta abarca además la selección y presentación de dichos datos. Es por tanto una primera aproximación a la estructura del sitio web. Su principal objetivo es facilitar al máximo los procesos de comprensión y asimilación de la información.

La IA permite estimar el número de páginas que se deberán crear y establecer un esquema primitivo de navegación que las relacione. La técnica que se utilizará para realizar la IA de la web de Boundless será la *clasificación por tarjetas*^[13] (*sorting cards*). Esta técnica es útil para realizar una categorización del contenido del sitio web centrada en el usuario. En concreto facilita la toma de decisiones en la etapa de diseño conceptual.

La prueba consiste en, dado un número finito de categorías, solicitar a un usuario que agrupe dichas categorías libremente en el número de conjuntos que considere necesario. Así se obtiene una distribución familiar y comprensible de los contenidos del sitio web para el usuario. Tras llevar a cabo la clasificación por tarjetas y tras analizar la estructura más común en los sitios web de contenidos similares, se ha obtenido el siguiente diagrama.

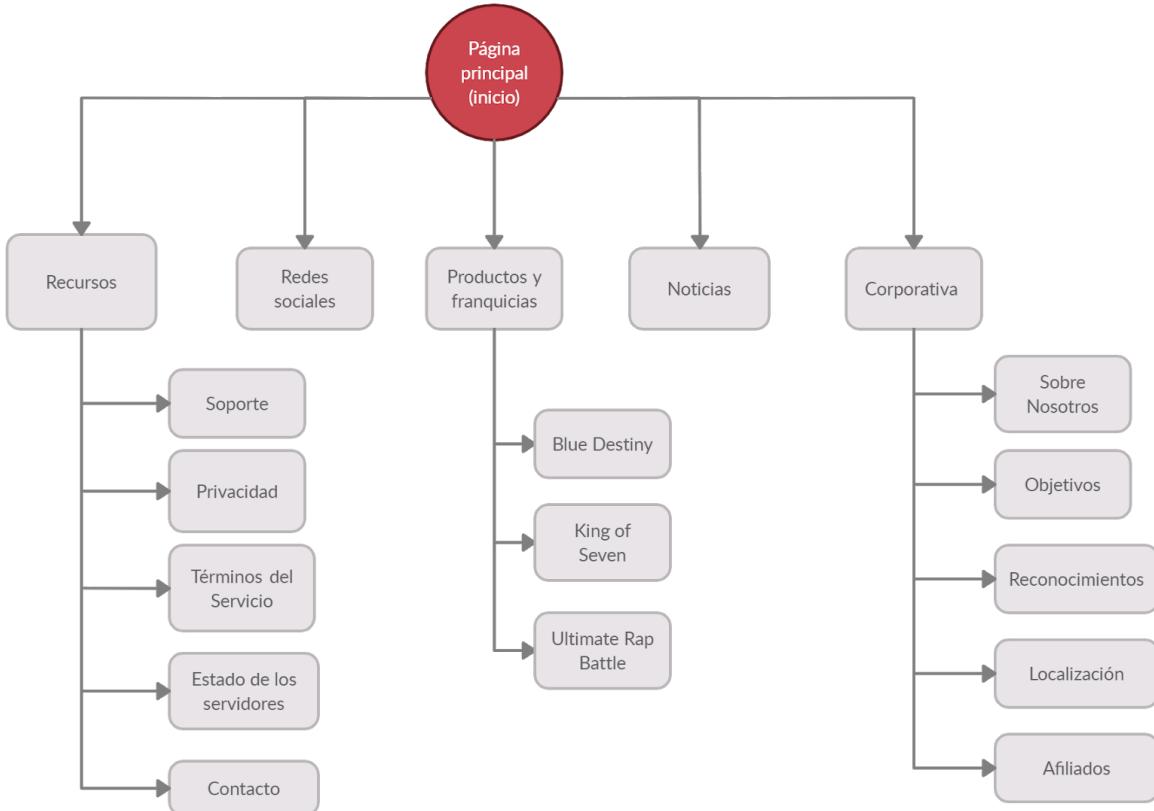


Figura II: Arquitectura de Información de Boundless

La figura II muestra la arquitectura de información propuesta para la web. En el primer nivel de la jerarquía mostrada en la imagen se encuentra la página principal, en el segundo las diferentes categorías o secciones en las que podemos dividir el contenido del sitio web y en el tercer y cuarto, contenido específico. En concreto:

- **Página principal.** El contenido y finalidad de la misma sigue siendo el mismo que el descrito en el apartado 3. *Inventario de contenido*. Se puede acceder a cualquier contenido del sitio partiendo de ella.
- **Recursos.** Conjunto misceláneo de ítems técnicos que resultan de utilidad al usuario. Dichas elementos serían: soporte, políticas de privacidad, términos del servicio, estado de los servidores e información de contacto.
- **Redes Sociales.** Información relacionada con las redes sociales oficiales de la empresa.
- **Productos y franquicias.** Sección que recoge toda la información sobre la temática y jugabilidad de los videojuegos que desarrolla *Boundless*.
- **Noticias.** Conjunto de novedades relacionadas con la empresa o sus productos. Informa al usuario de actualizaciones, parches, nuevos lanzamientos, eventos, etc.
- **Corporativa.** Parte del sitio web destinada a ofrecer información de carácter profesional acerca de la empresa, sus empleados, su trayectoria, localización,

objetivos y otras compañías con las que tenga algún tipo de relación comercial.

Se ha optado por este diseño debido a que ofrece una distribución intuitiva y lógica de contenidos. De este modo, se evita que el usuario se sienta confundido o desorientado buscando información concreta en el sitio web.

Tras haber seleccionado la distribución de los contenidos que se utilizará en el sitio se procederá a elaborar el **diseño de la interfaz de usuario**.

5. Diseño Gráfico de Interfaces

El siguiente paso en el diseño del sitio web consiste en realizar el diseño de las interfaces. Para esto hay que tener en cuenta el sector comercial, visión empresarial e ideales de la marca, entre otros. Esto se debe a que las interfaces ayudan a establecer una comunicación visual con el usuario y le transmiten mensajes específicos. La impresión que causa la web en la audiencia es vital ya que afecta a la imagen que tiene acerca de la marca.

Como *Boundless* es una empresa del sector tecnológico el diseño de la misma debe ser de acabado moderno y sencillo. Como es tendencia² en este tipo de webs, también debería considerarse la utilización de temas de alto contraste para destacar el contenido que promocionan. Esto en concreto es de gran importancia puesto que sus productos están destinados a una audiencia principalmente conformada por gente joven. Por tanto, se busca un diseño que permita llamar su atención y así inviertan en la marca. Para ello, se incorporarán animaciones o grandes imágenes que causen impacto y aporten dinamismo. No obstante, el diseño será escueto para proporcionar un acabado elegante. Así, se debe encontrar un equilibrio entre lo sensacional y lo profesional que permita mostrar la seriedad de *Boundless* como empresa pero que resulte altamente atractivo para los usuarios.

5.1. Sketching

Una vez establecidos los objetivos del diseño se procedió a la elaboración de *sketches*^[14] de las principales páginas que compondrán el sitio web. Estos primeros prototipos se realizaron a mano a partir de la estructura definida en la IA. Dichos bocetos permiten plasmar varias ideas en papel con gran rapidez, lo que facilita la existencia de gran diversidad de diseños y la selección de aquellos que se ajusten más a las necesidades del cliente. De todos los bocetos realizados, los escogidos como base para el diseño de la web fueron los siguientes³.

² Véanse las páginas referenciadas 5,6,7,8 y 9.

³ Los bocetos realizados estarán sujetos a cambios en las siguientes etapas del diseño.

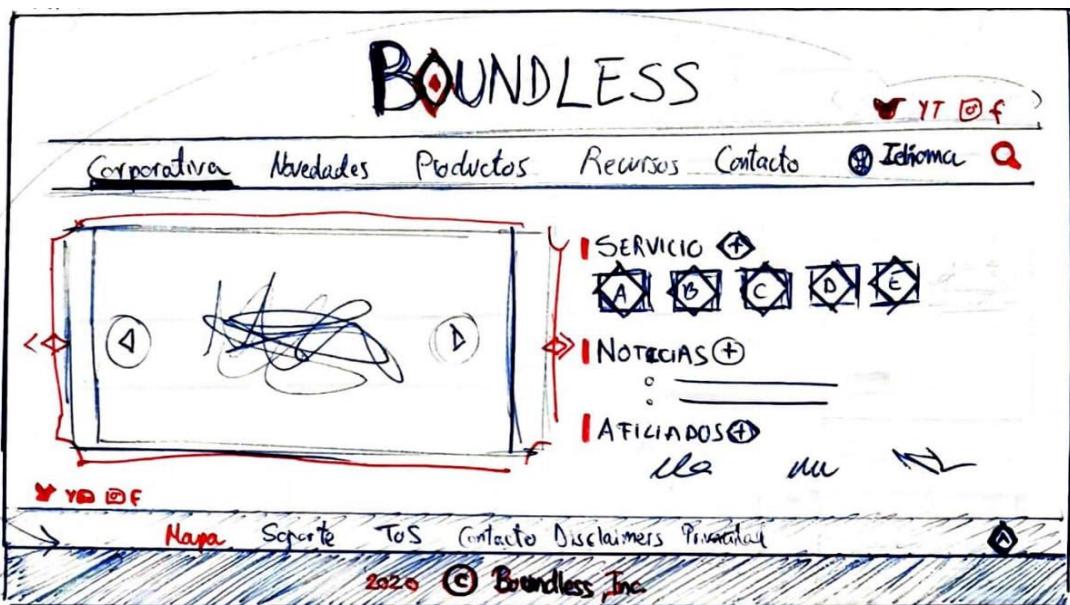


Figura III: Boceto de la página de inicio de Boundless

La página principal, mostrada en la figura III, se puede considerar la más importante. Es la imagen directa de la marca y además será, en la mayor parte de ocasiones, lo primero que verá el usuario del sitio web. Por tanto, se ha decidido colocar el logo de la marca en la parte superior. Debajo de este se incluye el menú de acceso a otras páginas importantes como las novedades o productos. Entre ambos, situado en la esquina superior derecha se colocarían enlaces a redes sociales. En el centro se encontraría el contenido de la página de inicio. Ocupando la mitad del espacio habría una presentación con publicidad con productos, eventos, etc. A la derecha del mismo, se proporcionaría acceso rápido a los servicios de la empresa, noticias recientes y afiliados. En la parte inferior se recogería toda la información legal relacionada con la empresa, soporte técnico, contacto y un mapa de la web. De este modo se sintetiza el contenido completo del sitio web. No se descarta la posibilidad de incluir imágenes e información adicional bajo la zona mostrada en el boceto.

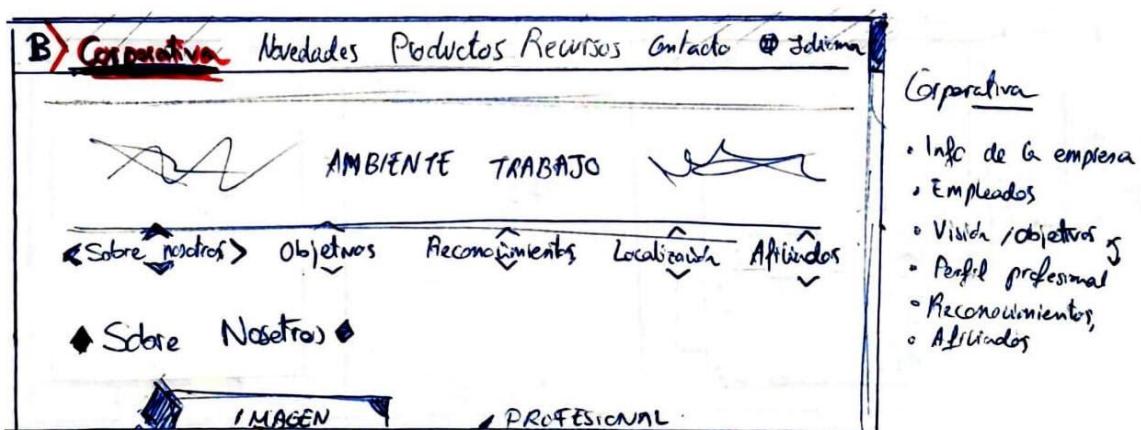


Figura IV: Boceto de la página dedicada a la Corporativa

En contraposición con la página principal, la página de la Corporativa mostrada en la figura IV se ha diseñado para que resulte más escueta, elegante y profesional puesto que está íntegramente dedicada a contener información sobre la propia empresa, su trayectoria, empleados, objetivos, reconocimiento, etc. Todas estas secciones se recogen enlazadas en un menú sencillo bajo una imagen de presentación del entorno de trabajo de la empresa. Para evitar la navegación constante entre las diferentes secciones (y como se espera que los contenidos de las mismas no sean demasiado extensos) se ha decidido agruparlas todas en la misma página web.

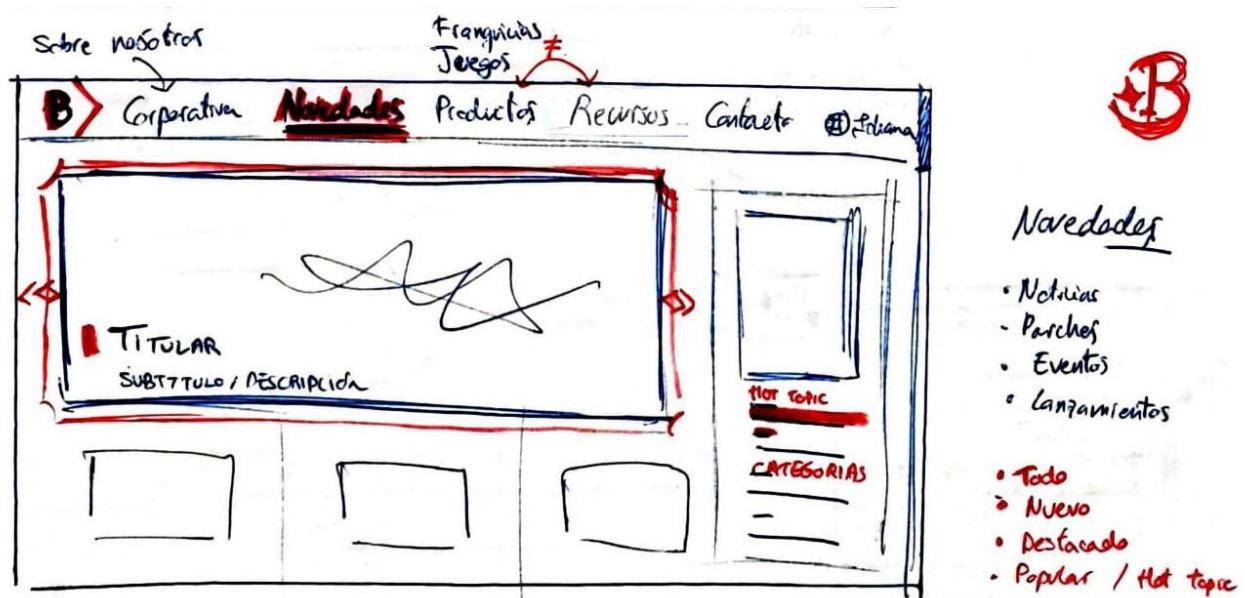


Figura V: Boceto de la página de noticias

La figura V muestra el diseño provisional de la página de noticias. Esta contiene una presentación con las novedades destacadas y más recientes en la parte superior derecha. En su lateral hay una sección que se extiende por el lateral derecho donde se incluye las categorías de las diferentes noticias, los temas populares, etc. Bajo la presentación se pueden previsualizar varias noticias. El resto de la página, que no se muestra en el boceto, incluye un listado de noticias con una portada, titular, fecha, categoría y descripción. Esta lista tiene un número de entradas finitas. Las entradas más antiguas que las mostradas se sitúan en listas diferentes dependiendo de su antigüedad. A estas se podrá acceder desde la página de noticias.

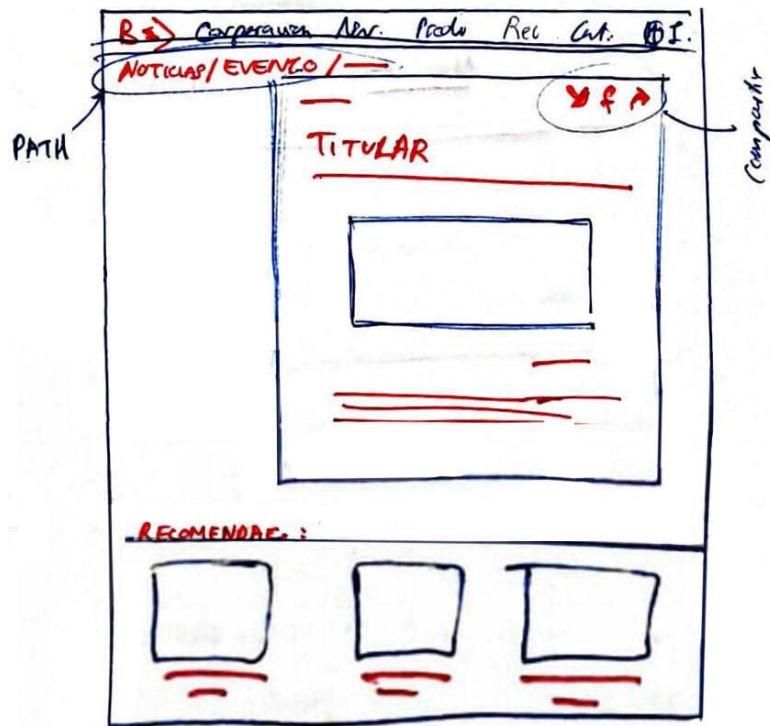


Figura VI: Boceto de la página de un artículo

En el caso de que un usuario solicite ver una noticia en concreto, el diseño de la página que verá será similar al de la figura VI. En este, la información de dicha noticia se muestra en un panel que ocupa más de la mitad derecha de la página. Este debe contener el titular, cuerpo, path, fecha y autor de la noticias. Adicionalmente puede contener alguna imagen que ilustre el contenido de la misma. Bajo el panel descrito se podrán encontrar otras noticias relacionadas con el tema de la mostrada.

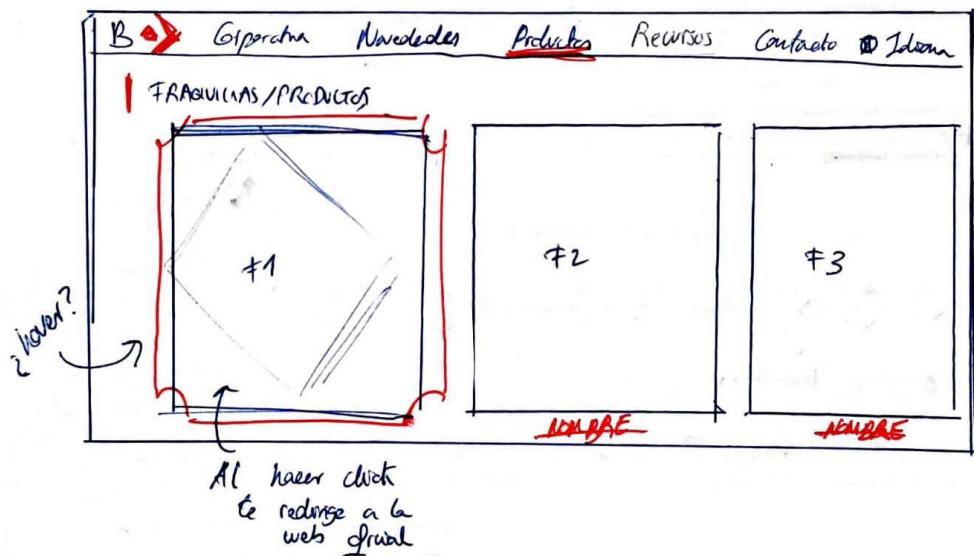


Figura VII: Boceto de la página Productos

La figura VII muestra el diseño que se le dará a la página dedicada a mostrar los productos de la empresa. En grande se muestran las portadas de los videojuegos. En este caso, esta página es la más susceptible a cambios. En caso de que el número de juegos aumente o disminuya se deberán realizar cambios en las dimensiones de las imágenes, el número de filas y columnas.

Con todos los bocetos realizados se podrá proceder a la elaboración del esqueleto digital.

5.2. Wireframe

El *wireframe*^[15], también denominado esquema de página, es una guía visual que representa la estructura visual de un sitio web. Este esqueleto ayuda a distribuir los elementos dentro de las páginas y a establecer las proporciones adecuadas entre ellos. A diferencia del boceto, este esquemas es mucho más claro. El wireframe permite establecer una jerarquía clara entre elementos basada en los tamaños y posiciones. Además, resultan extremadamente útiles para detectar y corregir errores de diseño^[16]. Al ser sencillos y rápidos de realizar te permiten exponerlos rápidamente a feedback y resolver problemas básicos relacionados con la usabilidad y funcionalidades propuestas.

En el caso de *Boundless*, se detectaron varios errores en tamaños y disposición de los elementos en los bocetos. El wireframe resultante es el siguiente.

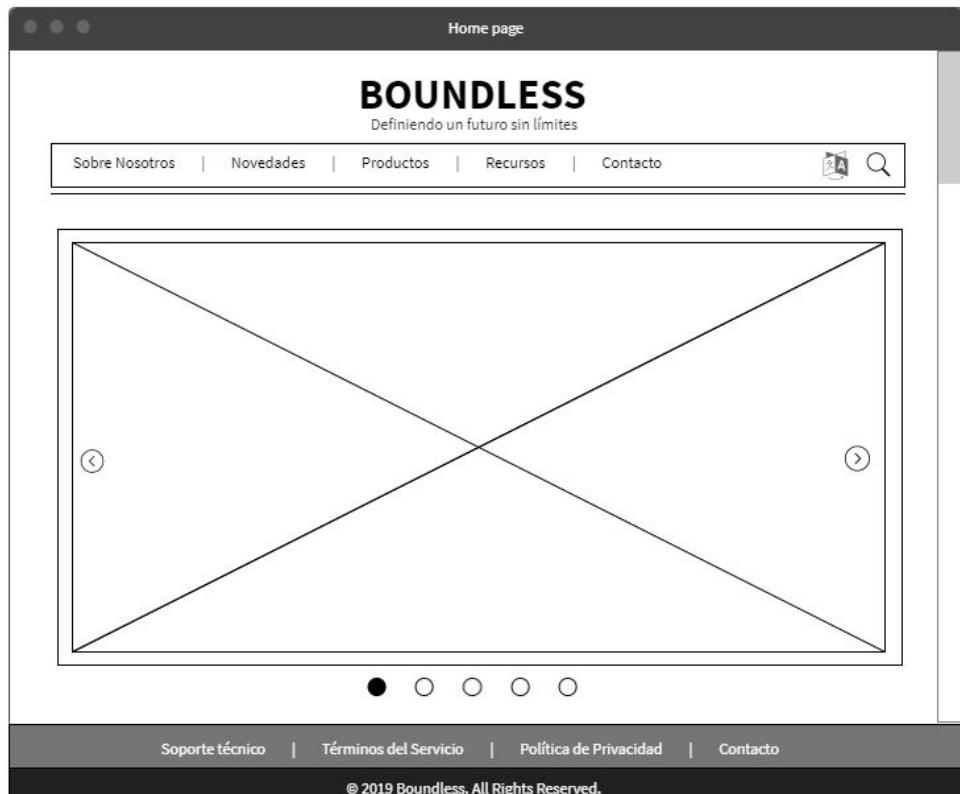


Figura VIII: Wireframe de la parte superior de la página principal

La figura VIII muestra el diseño corregido de la parte superior⁴ de la página de inicio. En él, se llevó a cabo una importante redistribución del contenido. En primer lugar, la presentación de novedades fue sustituida por una presentación que ocupa la pantalla completa del usuario. En ella se seguirá promocionando el mismo contenido, pero se ha modificado su tamaño y posición para darle más importancia en el diseño. Adicionalmente, redujeron el número de elementos listados en el pie de la página.

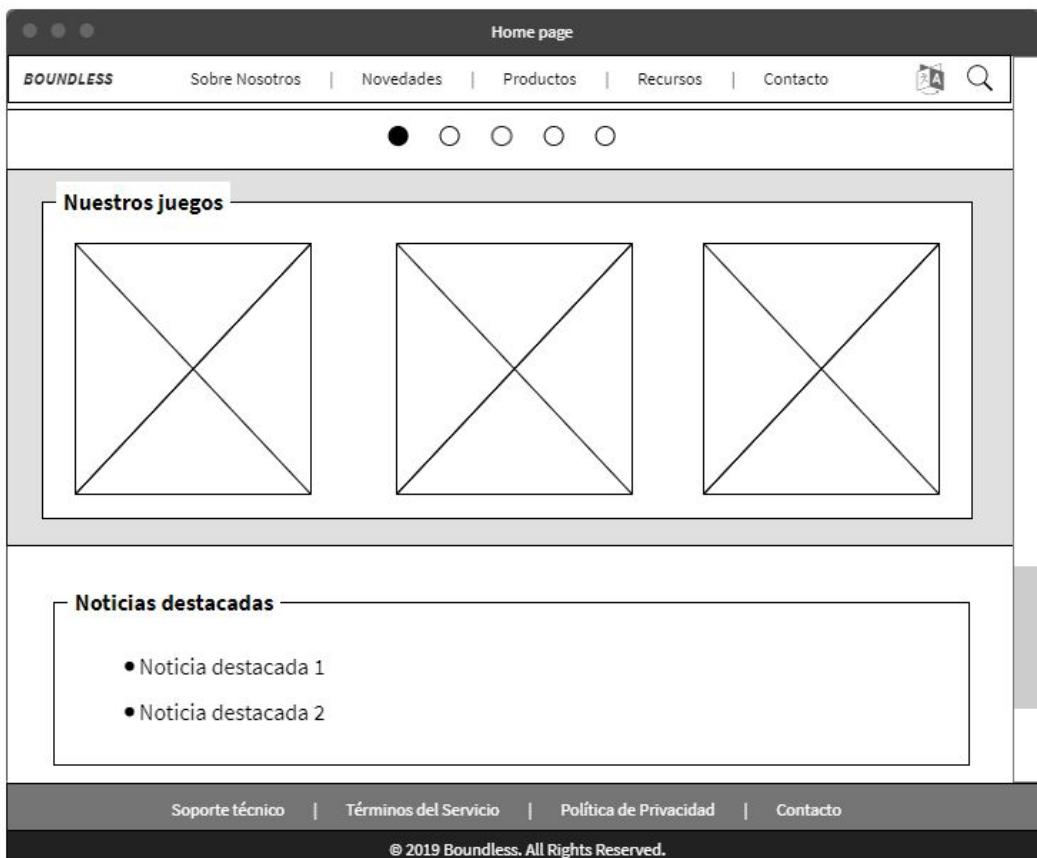


Figura IX: Wireframe de la parte inferior de la página principal

La parte inferior de la página de inicio se muestra en la figura IX. Con respecto al boceto, también se desplazaron y redimensionaron los apartados de *servicios* y *noticias*, y el apartado de *afiliados* fue eliminado de la portada. Esto último se debe a que su presencia en esta página no se consideró necesaria al existir una réplica en la sección *Sobre Nosotros*. Así, en este nuevo diseño se divide el espacio disponible entre las portadas de los juegos de la compañía y una sección con los titulares más destacados.

⁴El wireframe de la mayoría de las páginas se presenta dividido en dos partes por comodidad. El lienzo que permitía utilizar la web para realizarlo era limitado y la mayor parte de las páginas presentadas no eran visibles por completo.

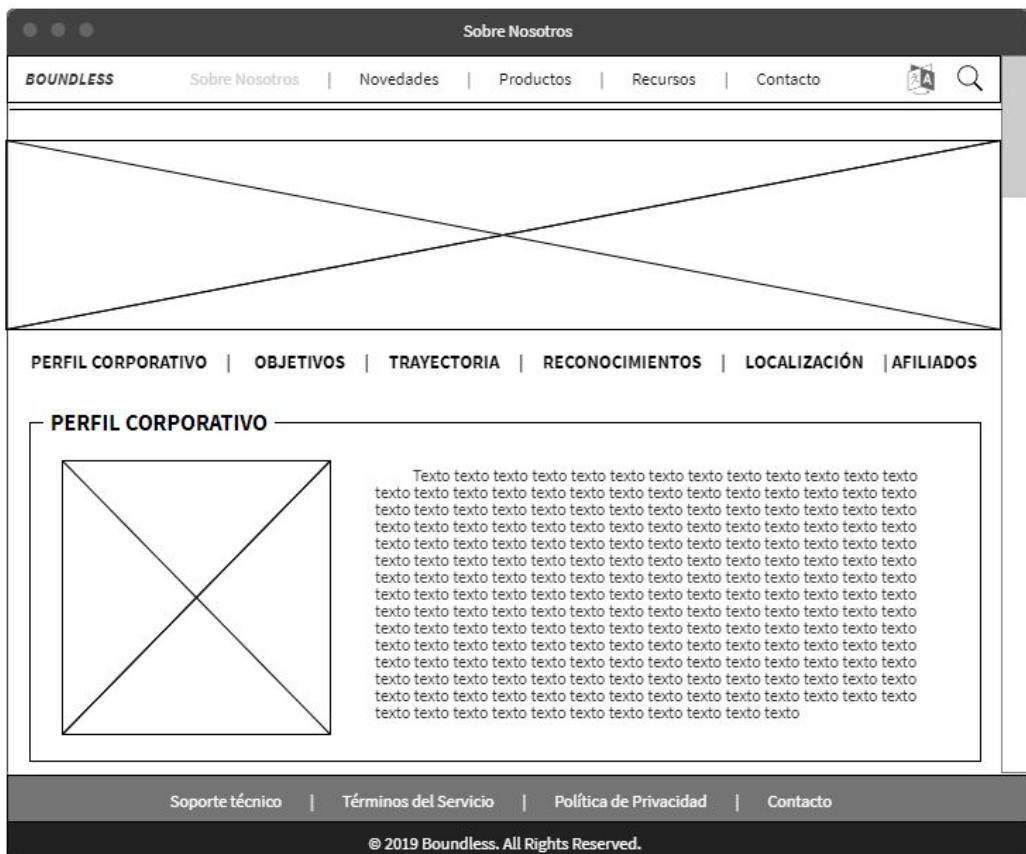


Figura X: Wireframe de la parte superior de la página Sobre Nosotros

Otra página que sufrió cambios significativos fue la de *Corporativa*. La parte superior del nuevo diseño se muestra en la figura X. Se añadieron nuevas secciones a la página y las antiguas se mantuvieron aunque algunas fueron renombradas. Adicionalmente, todas ellas fueron separadas en cuadros independientes en lugar de sucederse una tras otra. En la imagen se puede ver la sección *Perfil Corporativo*, antigua *Sobre Nosotros*. Por otra parte, los iconos presentes en el boceto previo como adornos fueron eliminados. Finalmente, la página fue renombrada como *Sobre Nosotros* para dar una impresión más afable al usuario.

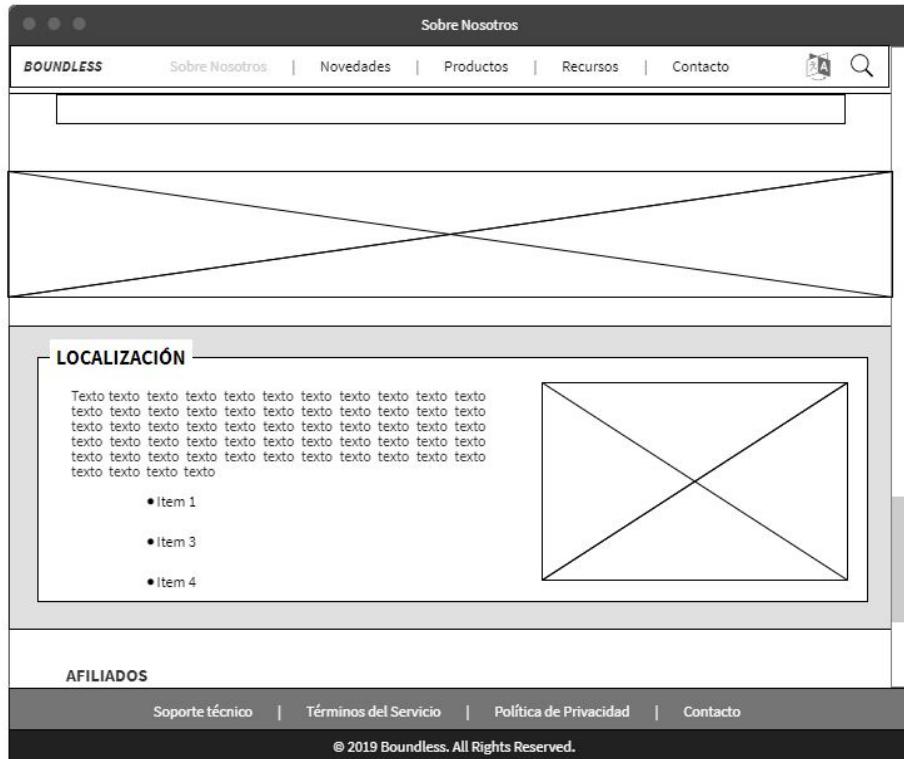


Figura XI: Wireframe de la parte inferior de la página Sobre Nosotros

En la figura XI se puede observar como la parte inferior de la página *Sobre Nosotros* tiene el mismo patrón de diseño que la superior.

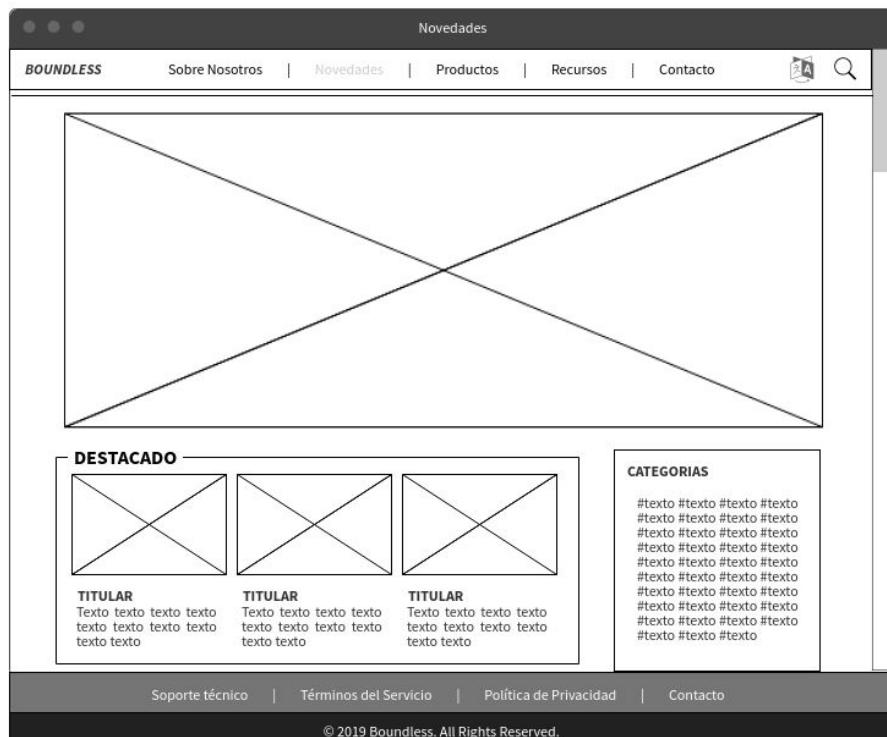


Figura XII: Wireframe de la parte superior de la página Noticias

La siguiente página modificada fue *Noticias*. El nuevo diseño, visible en la figura XII, reestructura el contenido de la página con respecto al boceto previo. En este caso, la presentación de las noticias más destacadas fue redimensionada para ocupar de forma exclusiva toda la sección superior de la vista. Por tanto, la sección adyacente donde se muestran las categorías de las noticias fue movida hacia abajo. La sección inferior de noticias destacadas se mantuvo.

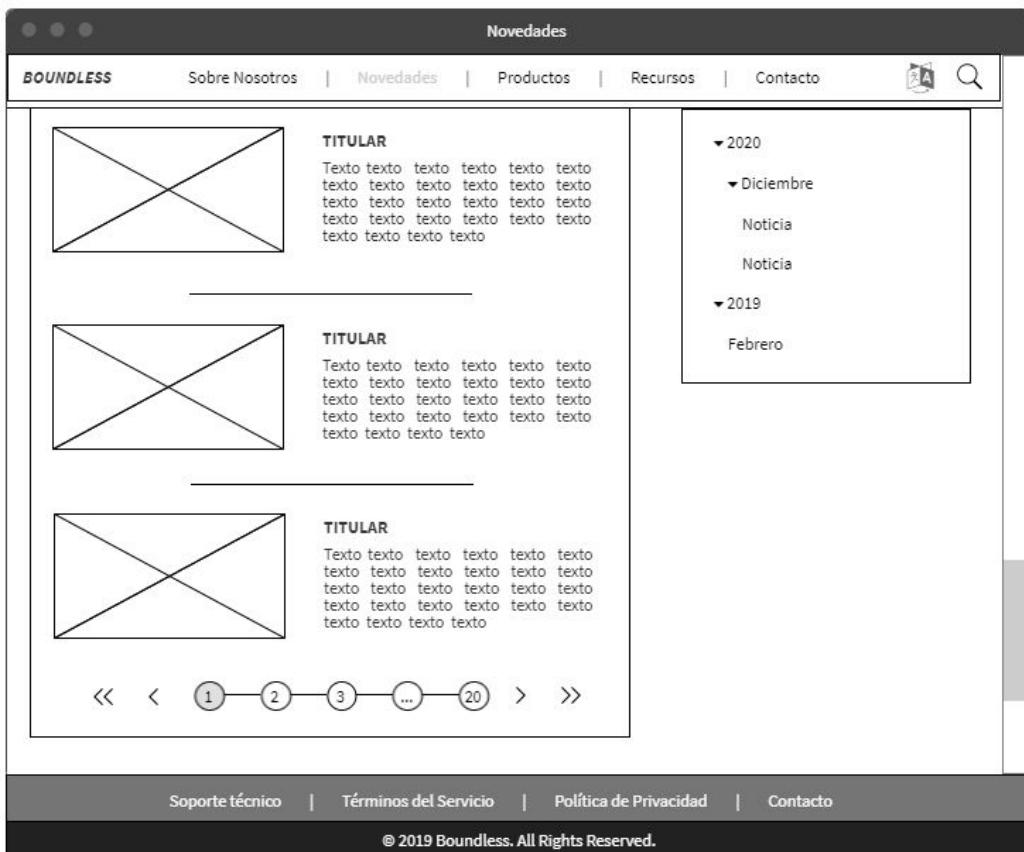


Figura XIII: Wireframe de la parte inferior de la página *Noticias*

La figura XIII precisa el diseño de la parte inferior de la página, el cual fue descrito en el apartado 5.1. Este diseño presenta dos partes claramente diferenciadas. La sección de la izquierda contiene un listado ordenado cronológicamente de las noticias que se publican en la web. Cada ítem de la lista cuenta con una imagen, titular, fecha y descripción. Adicionalmente puede contener etiquetas, las cuales se utilizan para categorizar las noticias. El menú numérico situado al final del listado permite acceder a otras páginas con noticias anteriores o posteriores a las que se muestran. La sección derecha es la continuación del elemento visto en la parte superior y que contiene también las categorías. En esta imagen se ve la parte del archivo. Este permite acceder a las noticias que se publicaron en un año y/o mes específico.

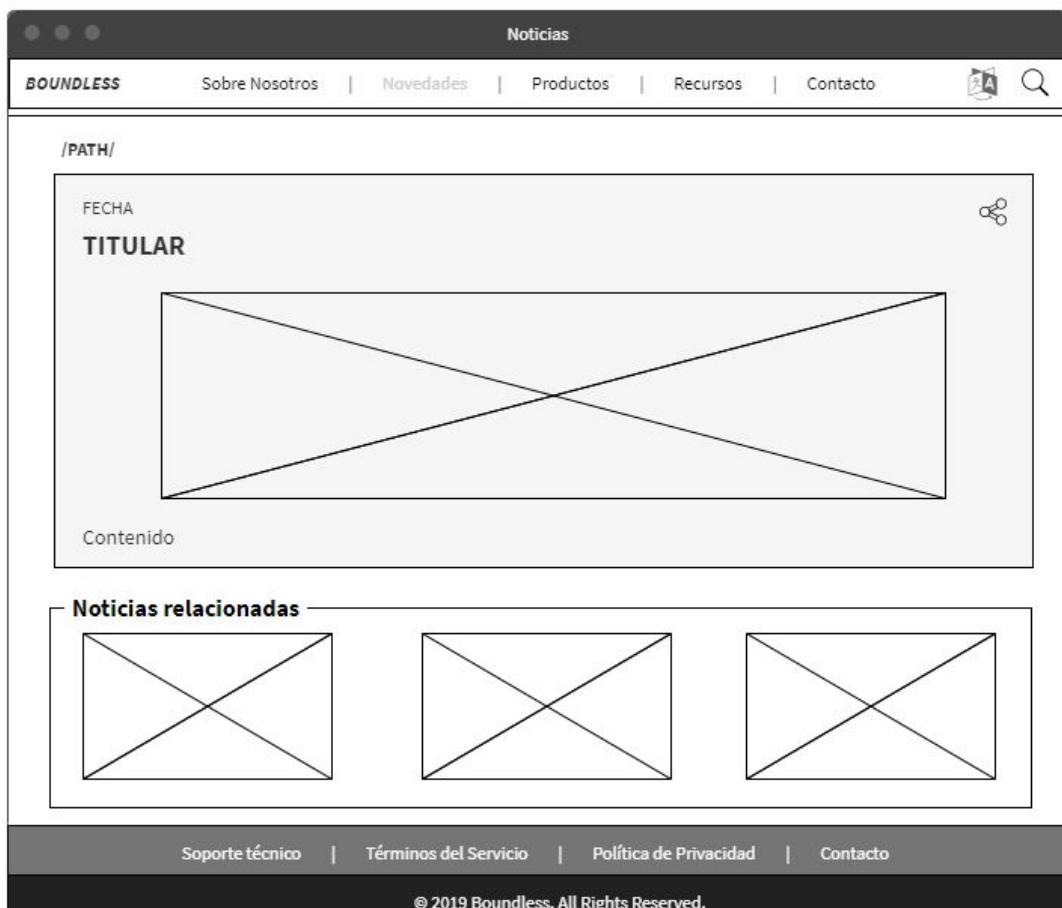


Figura XIV: Wireframe de un artículo

Las noticias que se publican en la web utilizarán el diseño mostrado en la figura XIV. Este también ha sido modificado con respecto al boceto. Se mantiene el *path* de la noticia, la estructura de fecha, titular, imagen y contenido y la sección de recomendaciones. La principal diferencia es que se ha aumentado considerablemente el ancho que ocupa la noticia y las opciones para compartir el enlace a la misma se han unificado en uno único. Se debe tener en cuenta que la longitud de la página dependerá del contenido del artículo. Al final de la página se encuentran una serie de artículos de temática similar.

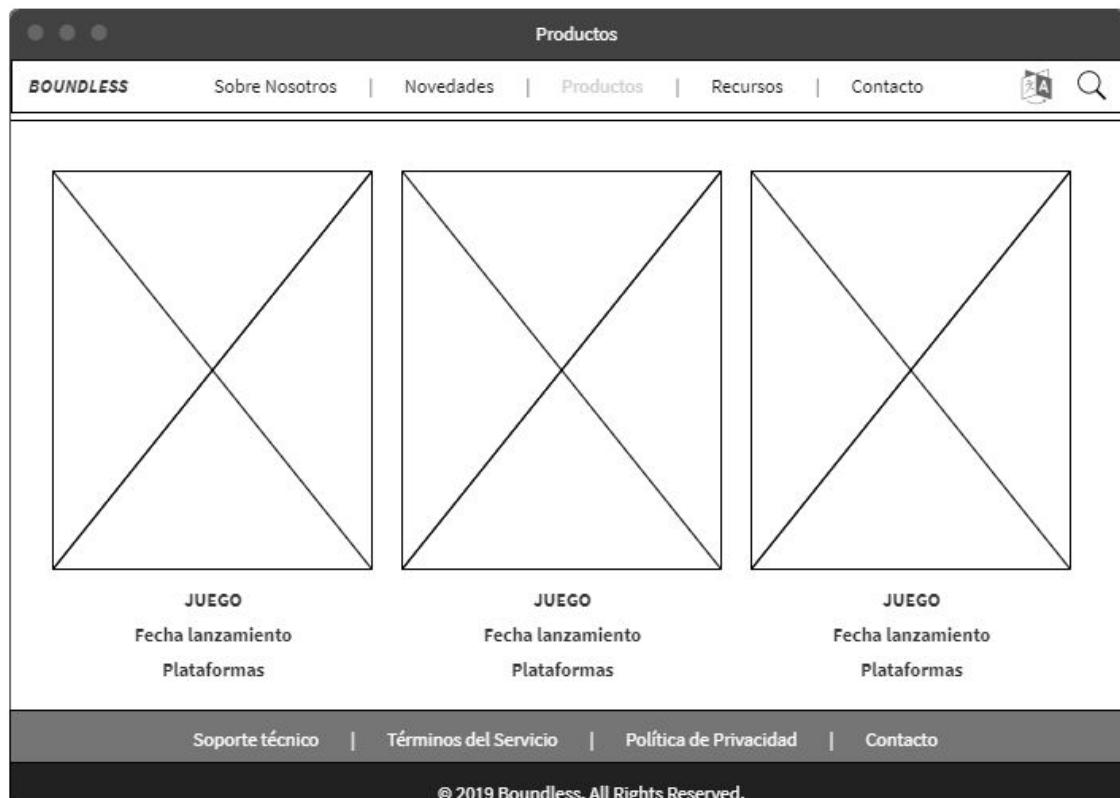


Figura XV: Wireframe de la página de Productos

La figura XV muestra el diseño de la página destinada a los productos de la empresa. En este caso, no ha habido variaciones significativas con respecto al sketch. La principal diferencia es que se han añadido bajo la portada de cada juego su nombre, fecha de lanzamiento y plataformas en las que está o estará disponible.

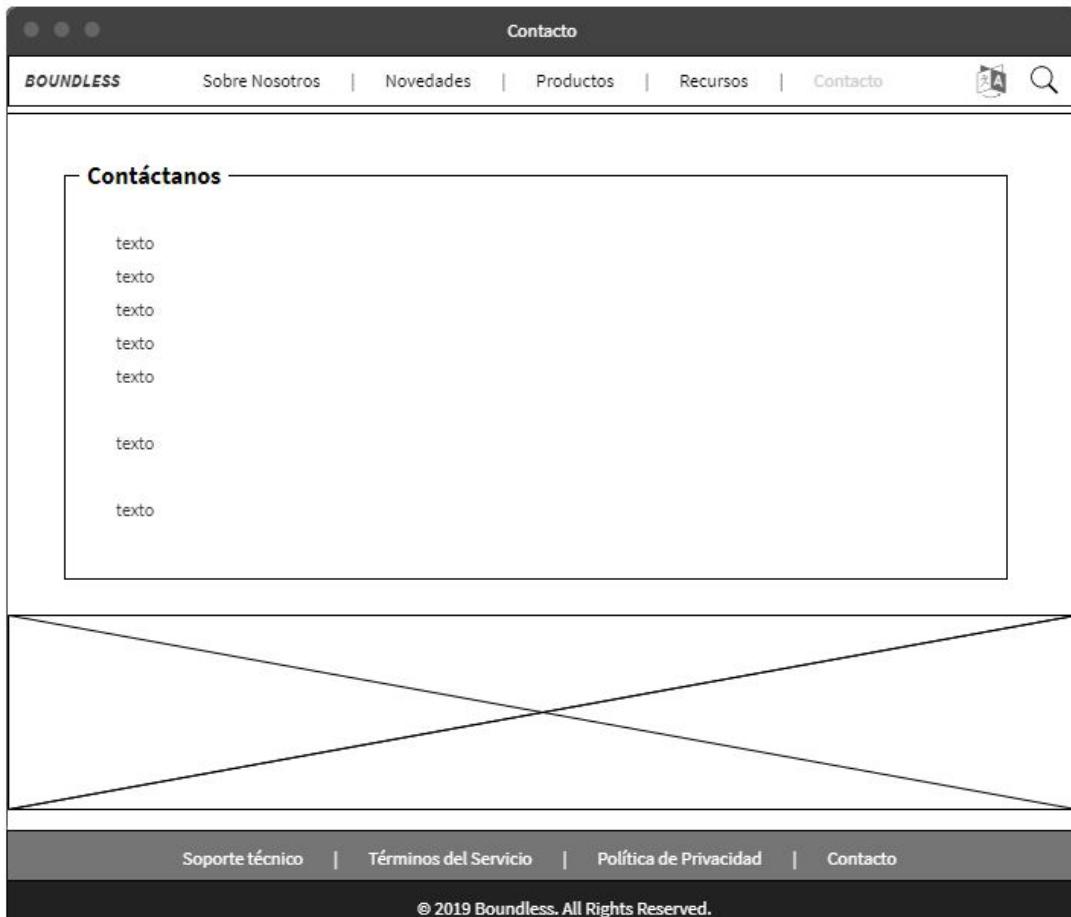


Figura XVI: Wireframe de la página de Contacto

La página de contacto tiene un diseño muy simple. Esta contiene toda la información de contacto de la empresa en sección superior, como se puede apreciar en la figura XVI. Debajo hay una imagen de adorno⁵. Este diseño se utilizará también para las páginas *Soporte técnico*, *Términos del Servicio*, *Política de Privacidad*, *Estado de los Servidores* y *Recursos*.

5.3. Mockup

La última parte del diseño de interfaces consiste en realizar el *mockup* del sitio. El *mockup*^[17] o maqueta es un modelo completo del diseño que se utiliza para mostrar y evaluar el resultado final. El maquetado para la web de *Boundless* utiliza como fuente *Calibri*. La razón principal de esto es que dicha fuente facilita la lectura, lo que hace la web más accesible. Para el diseño final se ha escogido una paleta de colores por lo general poco saturada que contrasta con las vívidas imágenes y los paneles rojo carmesí. El resultado es el siguiente.

⁵ La posición de la imagen puede variar en la versión final.



Figura XVII: Mockup de la página principal

The mockup displays a professional website layout. At the top, a navigation bar features a logo (a stylized 'B' with a play button icon), followed by menu items: SOBRE NOSOTROS, NOVEDADES ▾, PRODUCTOS, RECURSOS ▾, and CONTACTO. A search icon is also present. Below the header is a large, vibrant anime-style illustration of a girl with long white hair and blue eyes, surrounded by falling petals. A decorative border of dark diamonds frames the top of the main content area. The main content is organized into sections: 'PERFIL CORPORATIVO' (with sub-links to OBJETIVOS, TRAYECTORIA, RECONOCIMIENTOS, LOCALIZACIÓN, and AFILIADOS), 'PERFIL CORPORATIVO' (containing company details like name, address, president, foundation date, capital, and industry), and 'OBJETIVOS' (with the tagline 'Definiendo un futuro sin límites'). A photograph of a modern, multi-story office building under construction is included. The footer contains links for Soporte Técnico, Términos del Servicio, Política de Privacidad, Contacto, and a copyright notice: © 2019 Boundless. All Rights Reserved.

PERFIL CORPORATIVO

Nombre oficial:
Boundless, Inc.

Oficina:
Calle Maestro Don Marciano, 1,
33011 Oviedo, Asturias

Presidente:
Andrea Martínez Paradela

Fundación:
Julio 2012

Capital:
1.758.000 €

Industria:
Planificación y desarrollo de juegos

OBJETIVOS

Definiendo un futuro sin límites

Soporte Técnico | Términos del Servicio | Política de Privacidad | Contacto

© 2019 Boundless. All Rights Reserved.

Figura XVIII: Mockup de la página Sobre Nosotros

B SOBRE NOSOTROS | NOVEDADES ▾ | PRODUCTOS | RECURSOS ▾ | CONTACTO

NEW PATCH NIGHTBLOOM

ILLUST BY KANIKAMARI & HENTITIE

NOTICIAS DESTACADAS

[ULTIMATE RAP BATTLE]
¡Nuevas canciones!

[KING OF SEVEN]
Llega un nuevo héroe

ARTICULOS

[U.R.P] Bienvenido al infierno
19-07-2020
Únete en este evento a los guerreros del inframundo para derrotar a la Reina Diane y obtén jugosas recompensas
#Evento #URB #UltimateRapBattle

[KoS] Evento de las luciérnagas
12-07-2020
Comienza el Festival de las Luciérnagas del Reino de Teivath. ¡Participa para obtener una skin exclusiva de Roaz!
#Evento #Skin #KoS #KingOfSeven

[BD] ¡Bienvenida Clarin!
05-07-2020
Clarin, la dulce maga de nieve, acaba de llegar al continente. Redúltala en el nuevo banner para que se una al Imperio
#Heroe #Skin #BD #BlueDestiny

<< < 1 2 3 ... 12 > >>

CATEGORIAS

- # Eventos
- # Personajes
- # Historia
- # Parches
- # Actualización
- # Canciones
- # URP
- # KoS
- # BD
- # Heroe
- # BlueDestiny
- # KingOfSeven
- # UltimateRapBattle
- # Skin
- # Lanzamiento
- # Novedad

ARCHIVO

- ▼ 2020
 - ◆ Julio
 - ◆ Junio
 - ◆ Mayo
 - ◆ Abril
 - ◆ Marzo
 - ◆ Febrero
 - ◆ Enero
- ▲ 2019
- ▲ 2018
- ▲ 2017
- ▲ 2016
- ▲ 2015
- ▲ 2014
- ▲ 2013
- ▲ 2012

Figura XIX: Mockup de la página Noticias

Figura XX: Mockup de un artículo

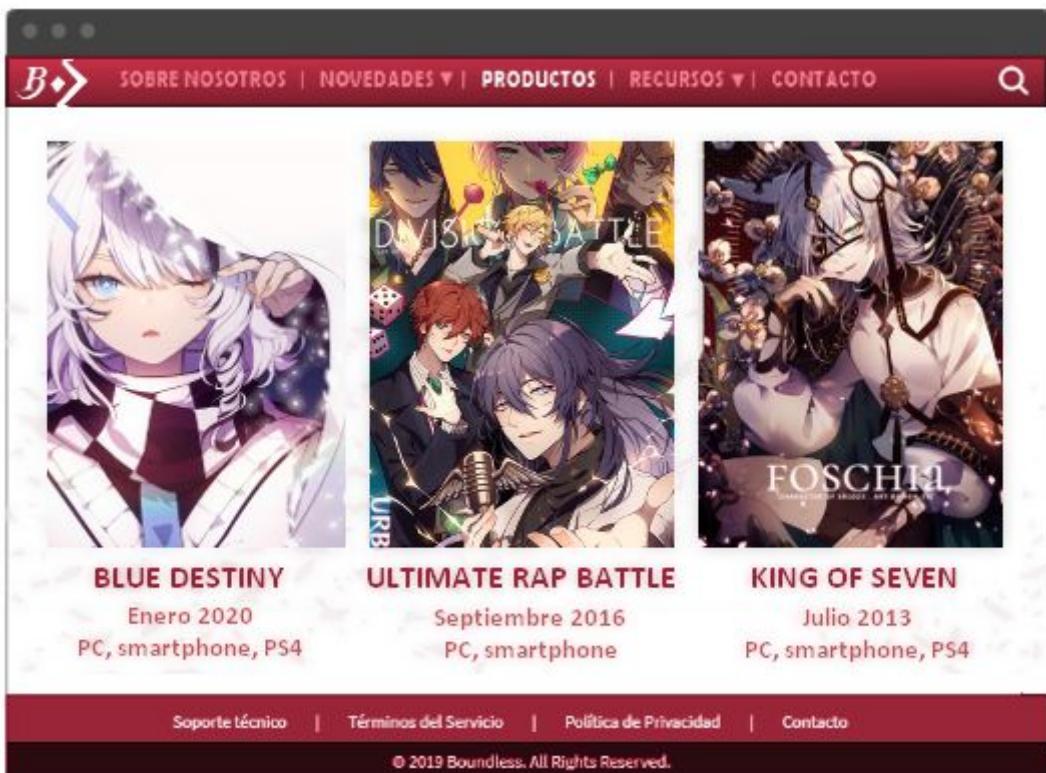


Figura XXI: Mockup de la página de Productos

Como se puede observar en las figuras de la XVII a la XXI todas las páginas presentan un diseño muy similar, el cual hace especial énfasis en las imágenes. En todas se ha mantenido el esquema propuesto en el apartado 5.2. *Wireframe*, variando únicamente el ancho de determinadas ilustraciones.

6. Casos de Uso

Las últimas etapas del diseño de la web consisten en optimizar la navegación entre las páginas y definir los perfiles de usuario que podrán acceder al sitio. Para ello, se realizará en primer lugar un **diagrama de casos de uso**^[18] con respecto a las actividades de la audiencia del sitio web. Se denomina diagrama de caso de uso a la descripción de las actividades que deberá realizar un actor para llevar a cabo algún proceso. Los casos de uso permiten entender qué acciones podrán realizar los diferentes tipos de usuarios en la página web. El diagrama de *Boundless* es el siguiente:

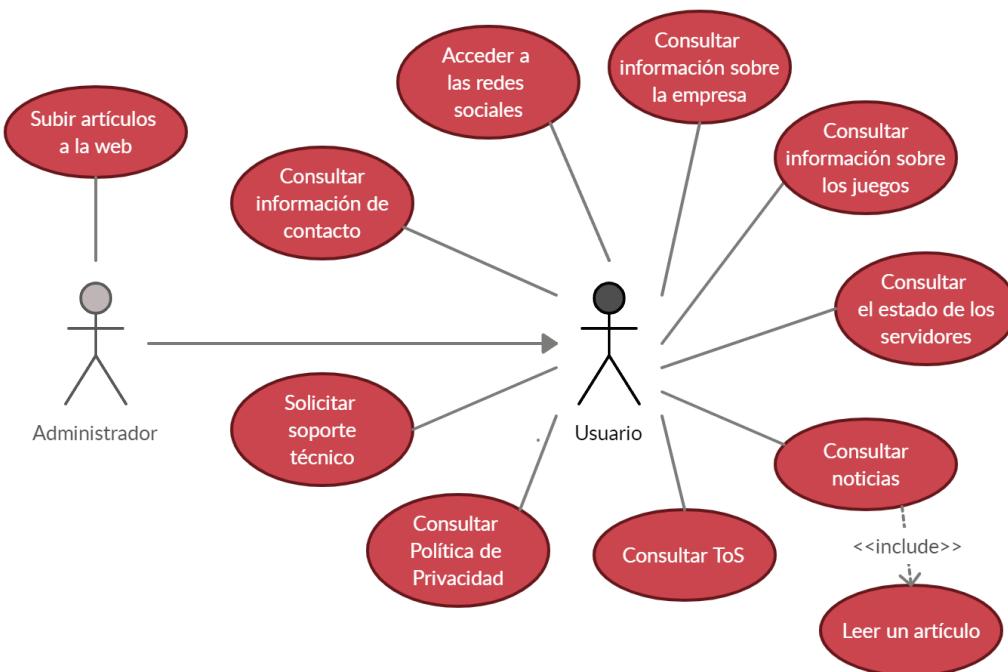


Figura XXII: Casos de uso del sitio web

La figura XXII muestra las diferentes acciones que se pueden hacer en el sitio web. En concreto:

- **Subir artículos a la web.** Un administrador puede colgar nuevos artículos en la sección de noticias. Esto debe hacerse desde la página *Noticias*.
- **Consultar noticias.** Un usuario puede solicitar ver un resumen que incluya las novedades de la empresa y sus productos. Esta información se obtiene en la página *Noticias*.
 - **Leer un artículo.** Un usuario puede solicitar leer un artículo completo. Esto se realiza desde la página *Noticias*.
- **Consultar información sobre la empresa.** Un usuario puede solicitar conocer más acerca de la empresa Boundless: perfil corporativo, objetivos, trayectoria, reconocimientos, localización y afiliados. Dicha información se encuentra disponible en la página *Sobre Nosotros*.
- **Ver información sobre los juegos.** Un usuario puede solicitar información acerca de los juegos de la empresa. Esto se hace en la página *Productos*.
- **Consultar el estado de los servidores.** Un usuario puede solicitar información acerca de los servidores de los juegos: capacidad, errores, actividad, etc. Dicha información se puede consultar en la página *Estado de los Servidores*.
- **Acceder a la información de contacto.** Un usuario puede solicitar la información de contacto de la empresa. Esto se hace desde la página *Contacto*.
- **Acceder a las redes sociales** de la empresa. Un usuario base puede solicitar las redes sociales de la empresa. Esto se hace desde la página *web principal*.

- **Consultar los Términos del Servicio.** Un usuario puede solicitar la lectura de los Términos del Servicio de la web y sus juegos. Dicha información se puede consultar en la página *Términos del Servicio*.
- **Consultar la Política de Privacidad.** Un usuario puede solicitar la lectura de la Política de Privacidad de la web y sus juegos. Dicha información se puede consultar en la página *Política de Privacidad*.
- **Solicitar soporte técnico.** Un usuario puede pedir soporte si ha experimentado problemas con algún producto de la marca. Dicha información se puede consultar en la página *Soporte*.

Como se puede observar, hay dos roles diferenciados: *administrador* y *usuario*. Por tanto, es posible que sea necesario implementar un sistema de *login* para controlar el acceso.

7. Mapa Web

La siguiente fase del diseño consiste en elaborar un **mapa del sitio web**^[18], es decir, una lista de las páginas de un sitio web accesibles por parte de los buscadores y los usuarios. Dicho mapa permite visualizar la relación entre las páginas. El mapa web de Boundless es el siguiente.

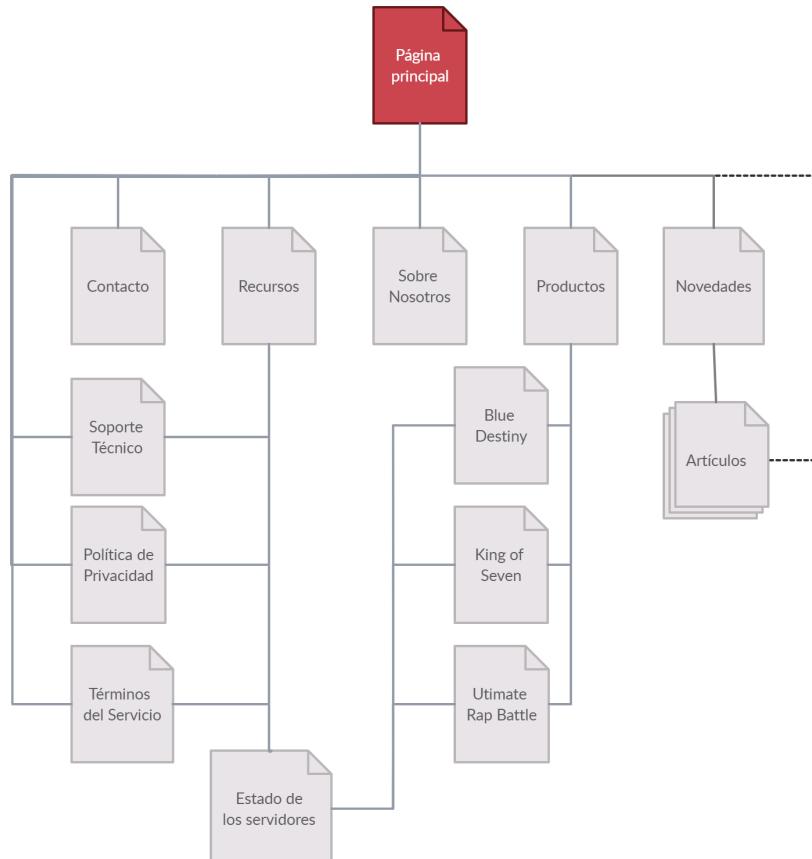


Figura XXIII: Mapa web de Boundless

La figura XXIII muestra el esquema propuesto como mapa web para *Boundless*. En él se puede apreciar como desde la *página principal* se pueden acceder a las páginas recogidas en el menú (*Sobre Nosotros, Novedades, Productos, Recursos y Contacto*) y a las situadas en el pie de página (*Soporte técnico, Términos del Servicio, Política de Privacidad y Contacto*). Por el contrario, solo se puede acceder a los artículos que se publican en el sitio web a través de la página de novedades o a través de enlaces en la página principal⁶. Lo mismo ocurre con las páginas dedicadas a los juegos (*Blue Destiny, King of Seven y Ultimate Rap Battle*), estas solo puede ser consultadas desde la página de *Productos*⁷. Así mismo, desde las páginas de los juegos y desde la página de recursos se puede acceder a la página de *Estado de los Servidores*.

8. Storyboard

Tras haber elaborado el diagrama de casos de uso y el mapas de navegación se pueden crear **storyboards**. Los *storyboards* permiten visualizar cómo se navega por el sitio web y permiten observar cómo interaccionan los elementos del mismo. Como la navegación por la web es muy sencilla, solo se podrá un ejemplo de *Storyboard*. Este muestra el proceso de acceso a un artículo destacado.

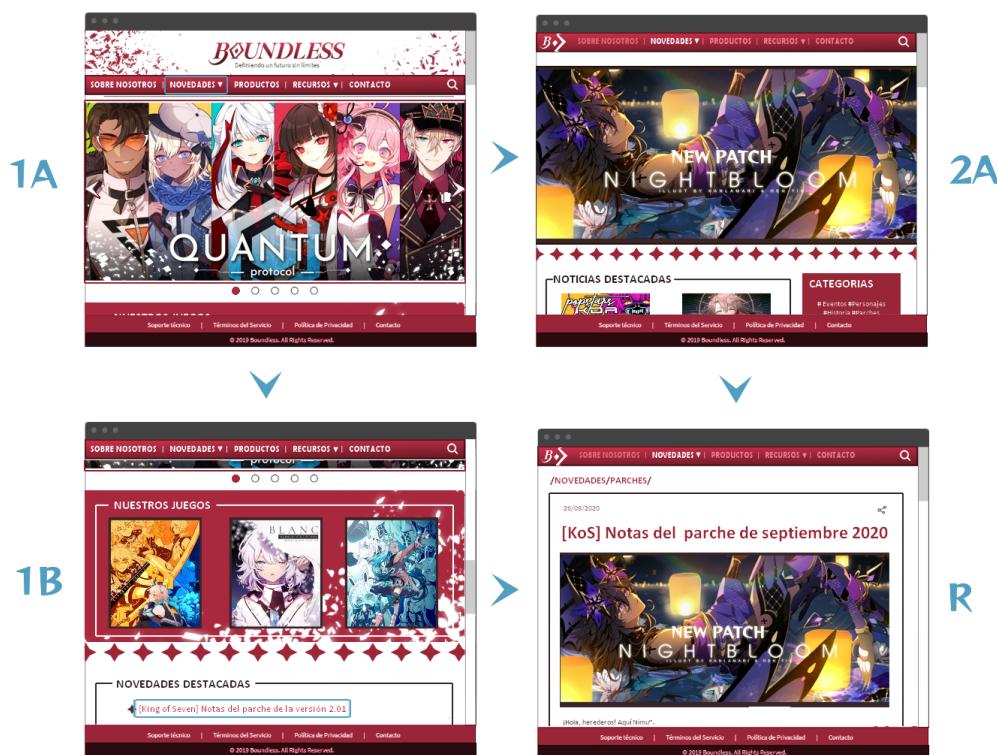


Figura XXIV: Storyboard de el acceso a un artículo destacado

⁶ Estos enlaces puntuales se han representado a través de una línea discontinua.

⁷ Es necesario destacar que aunque los juegos aparezcan en la página principal, desde ella se accedería a las páginas oficiales de los juegos y no a las páginas del sitio web de Boundless dedicadas a los mismos. La principal diferencia es que la página en Boundless ofrece información técnica sobre el título, mientras que la web oficial externa tiene un marcado carácter publicitario.

La figura XXIV muestra un storyboard muy escueto para el acceso a una noticia. Como se aprecia en la imagen, hay dos alternativas para ir al artículo. La primera (1A) de ellas consiste en hacer click en el apartado *Novedades* del menú de navegación para acceder a la página dedicada a las noticias del sitio. Como resultado, el usuario será redirigido a la página *Novedades*. Desde allí, sólo tiene que hacer click en el artículo que desee leer (2A). Así, se le redirigirá a la página del artículo concreto (R). Alternativamente, como la noticia que se desea ver está siendo destacada, se puede acceder a la misma haciendo scroll en la página principal (1A) hasta visualizar el apartado *Noticias Destacadas*. Al hacer click en la noticia listada (1B) el usuario será redirigido automáticamente al artículo (R).

El sitio web de Boundless se ha diseñado para que la navegación sea lo más sencilla posible y se requiera el menor número de clicks posibles. Esto es posible gracias a la inexistencia de operaciones complejas por parte del usuario.

9. Estructura de Ficheros

Finalmente, la última etapa del diseño consiste en proponer una **estructura de almacenamiento de los ficheros^[19]** para los elementos del sitio web. Para ello, se utilizará un sistema de archivos⁸ jerárquico que represente los contenidos de cada carpeta.

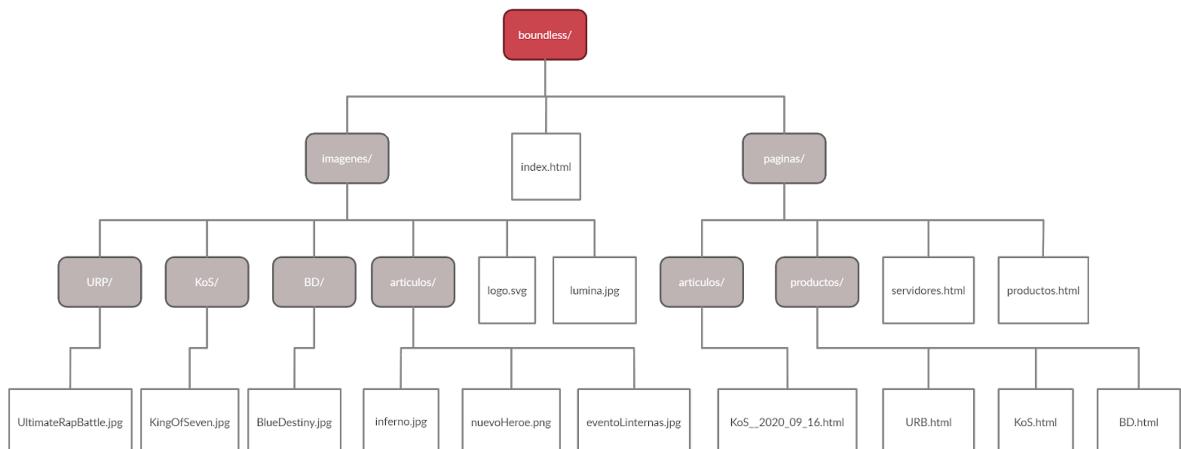


Figura XXV: Estructura de ficheros de Boundless

La estructura de ficheros propuesta en la figura XXV es sencilla. Para realizarla se ha tenido presente el *mapa web* visto en la sección 7 y la *arquitectura de información* de la sección 2 del presente documento. Así, se establece una carpeta raíz */boundless/* que contenga todos los archivos del sitio web. Dentro se podrán encontrar la página principal *index.html* y dos carpetas: *imágenes/* y *paginas/*.

En la primera se espera almacenar todas las imágenes de la web (ilustraciones, fotografías, logos, etc.). Este directorio cuenta con cuatro carpetas. La primera, */articulos/*, incluye las imágenes relacionadas con los diferentes artículos que se han publicado en la

⁸ No se debe confundir el path de los artículos con las rutas de este sistema.

web. Las otras tres restantes (*BD*, *KoS* y *URB*) almacenan ilustraciones promocionales de sus respectivos juegos.

En la segunda carpeta de *boundless/* se podrán encontrar el resto de páginas del sitio. Dentro de la misma hay un directorio dedicado a los artículos (*articulos/*). Este tipo de páginas se han separado del resto debido a que se irán publicando nuevos artículos periódicamente en la web y el número de estos irá creciendo considerablemente a medida que pasen los años. De esta forma es más sencillo acceder a ellos. Por otra parte, también existe otro directorio, */productos/*, donde se hallan páginas técnicas destinadas a los diferentes videojuegos de la empresa.

10. HTML y Mapas de Etiquetas

Ya finalizado el diseño, se puede comenzar con la elaboración de las páginas en html. En primer lugar, se elaborarán **mapas de etiquetas** de las diferentes páginas del sitio web. Esto permite estructurar de forma visual las marcas html y facilita la escritura del código html. A continuación se recogen los mapas de etiquetas realizados.

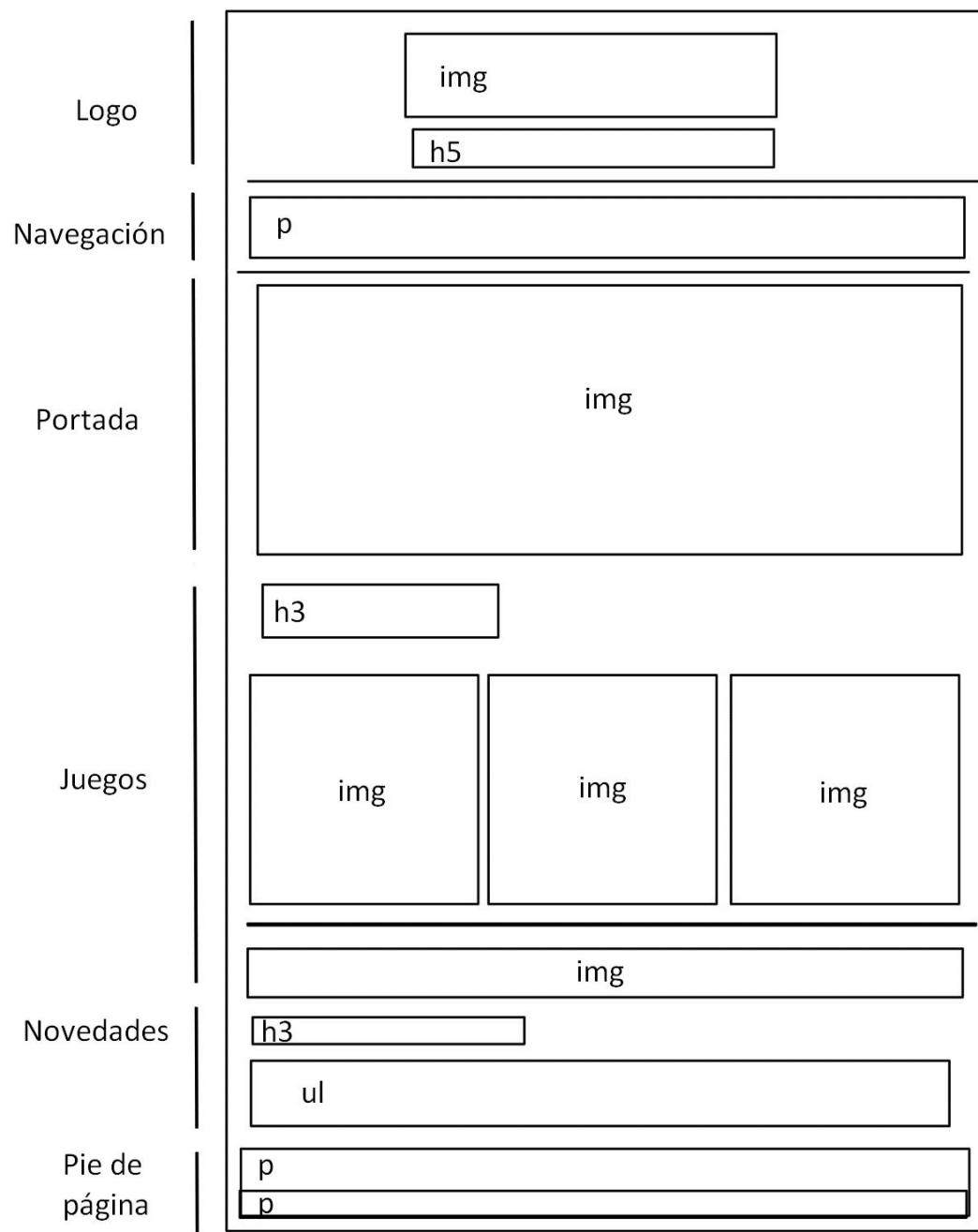


Figura XXVI: Mapa de etiquetas de la página de inicio

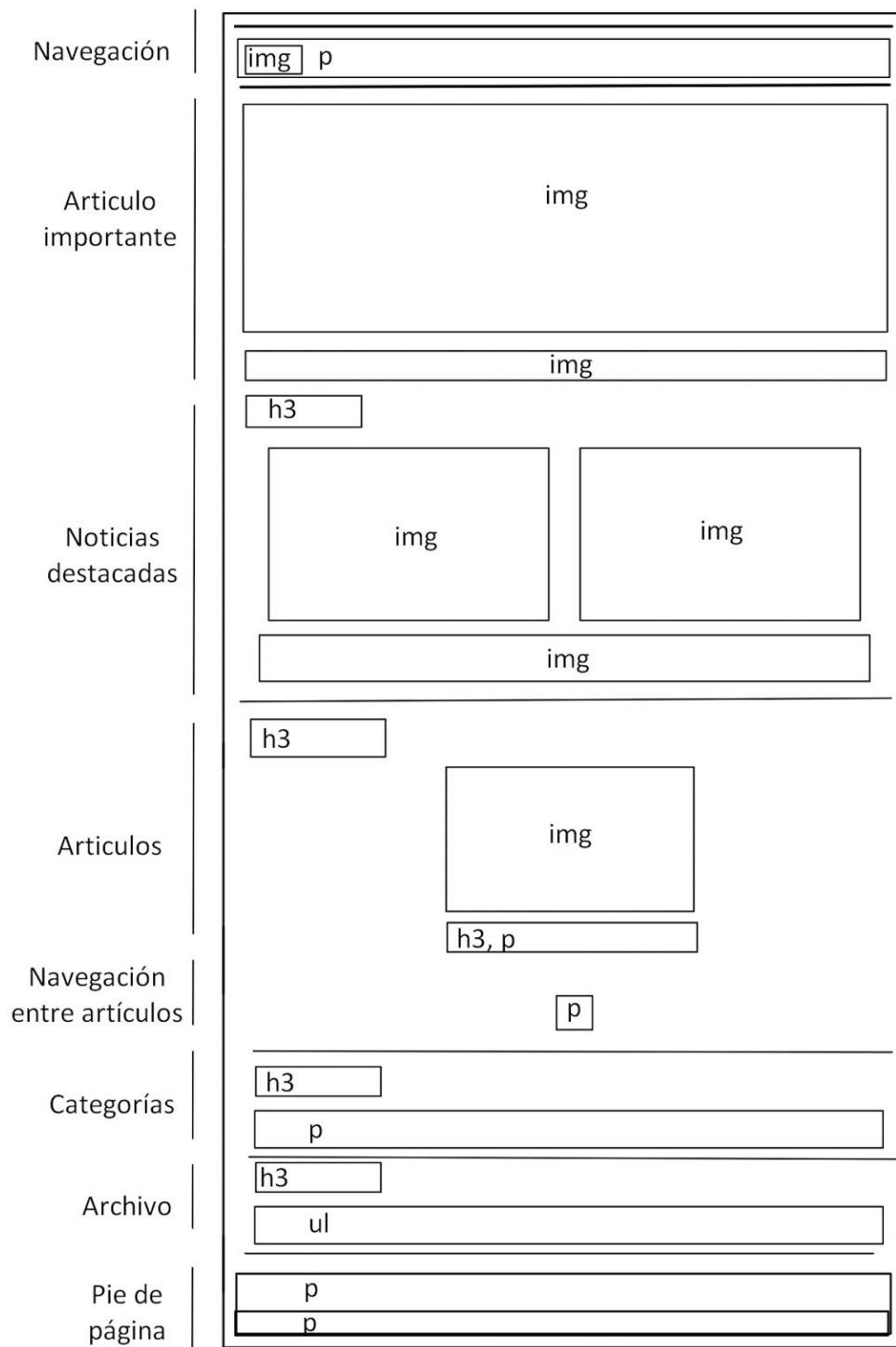


Figura XXVII: Mapa de etiquetas de la página de novedades

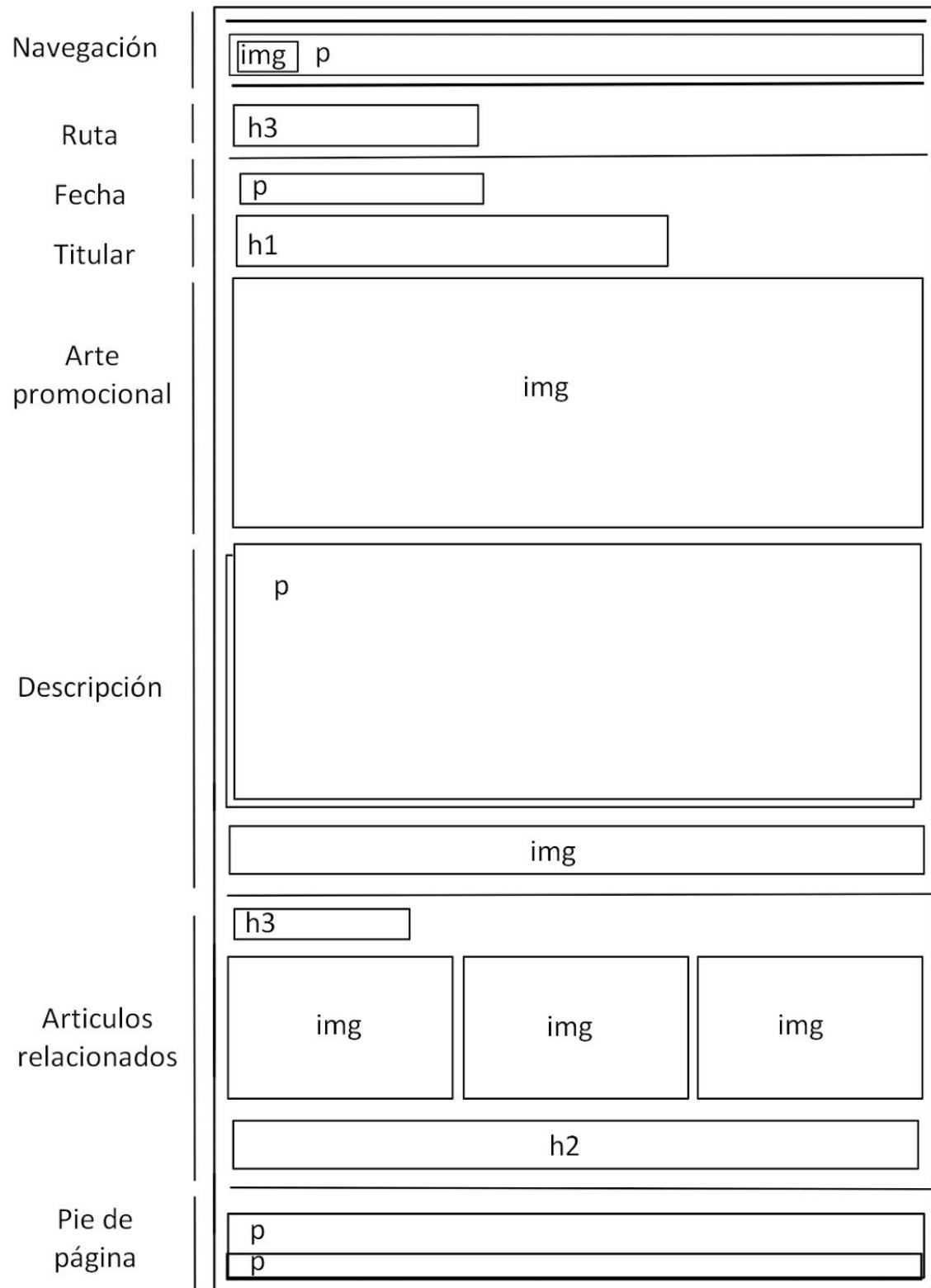


Figura XXVIII: Mapa de etiquetas de un artículo

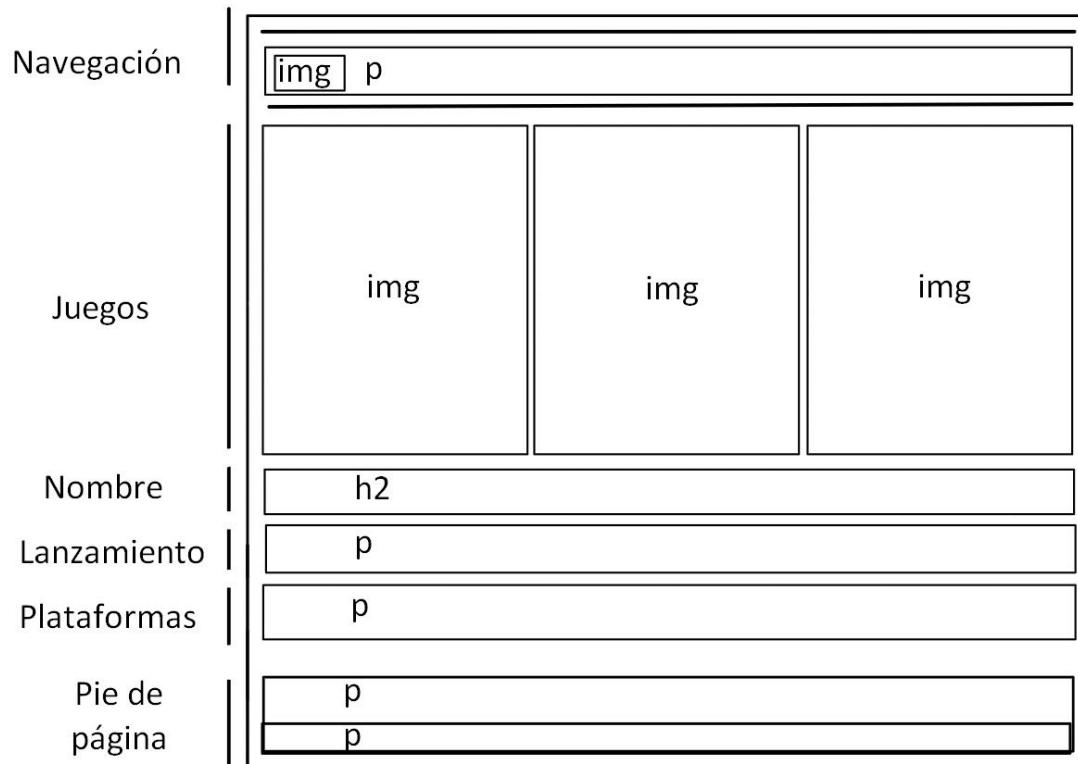


Figura XXIX: Mapa de etiquetas de la página de productos

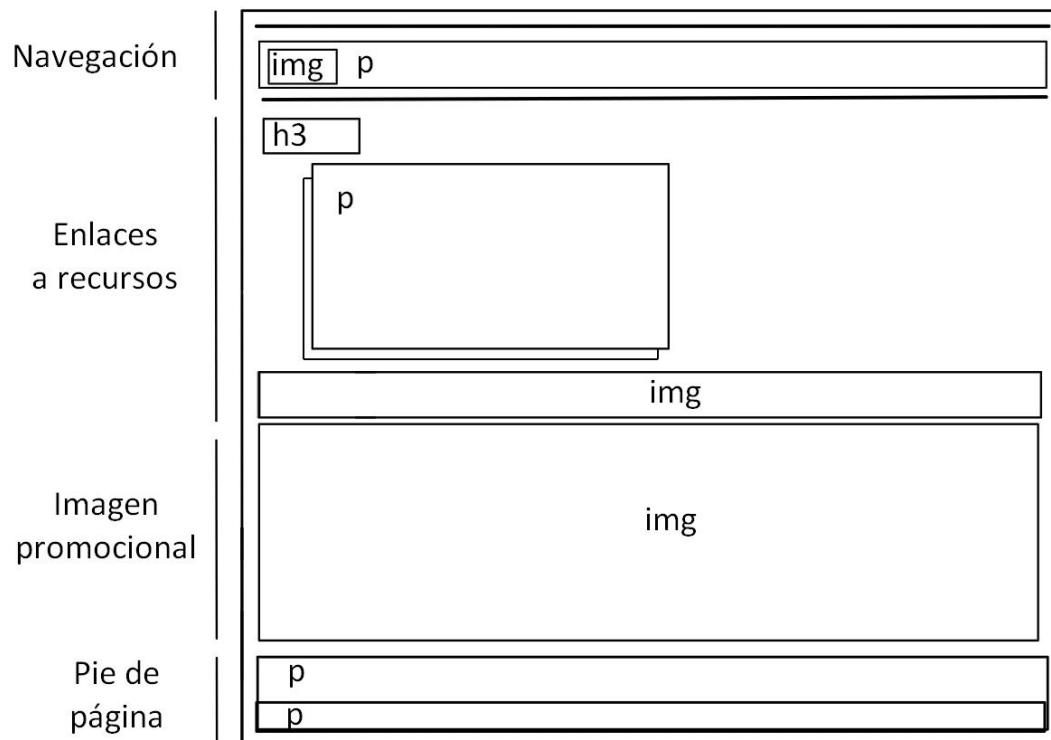


Figura XXX: Mapa de etiquetas de la página de recursos

Como se puede apreciar en las figuras XXVI a la XXX, la estructura de todas las páginas mostradas es muy similar. Todas cuentan con un menú sencillo de navegación, pie de página y contenido específico. En el caso de la página principal (figura XXVI), se puede observar que esta cuenta adicionalmente con el logo completo de la empresa en la parte superior.

Todos los *.html* se pueden encontrar en la carpeta *boundless*.

11. CSS y Mapas de clases

Tras la elaboración del esqueleto de la web en HTML5, se escribirá el código CSS que permita adecuar el diseño propuesto en el presente documento al sitio web de *Boundless*. Primero, se elaborarán los **mapas de clases** correspondientes a las páginas principales del sitio web. Estos son:

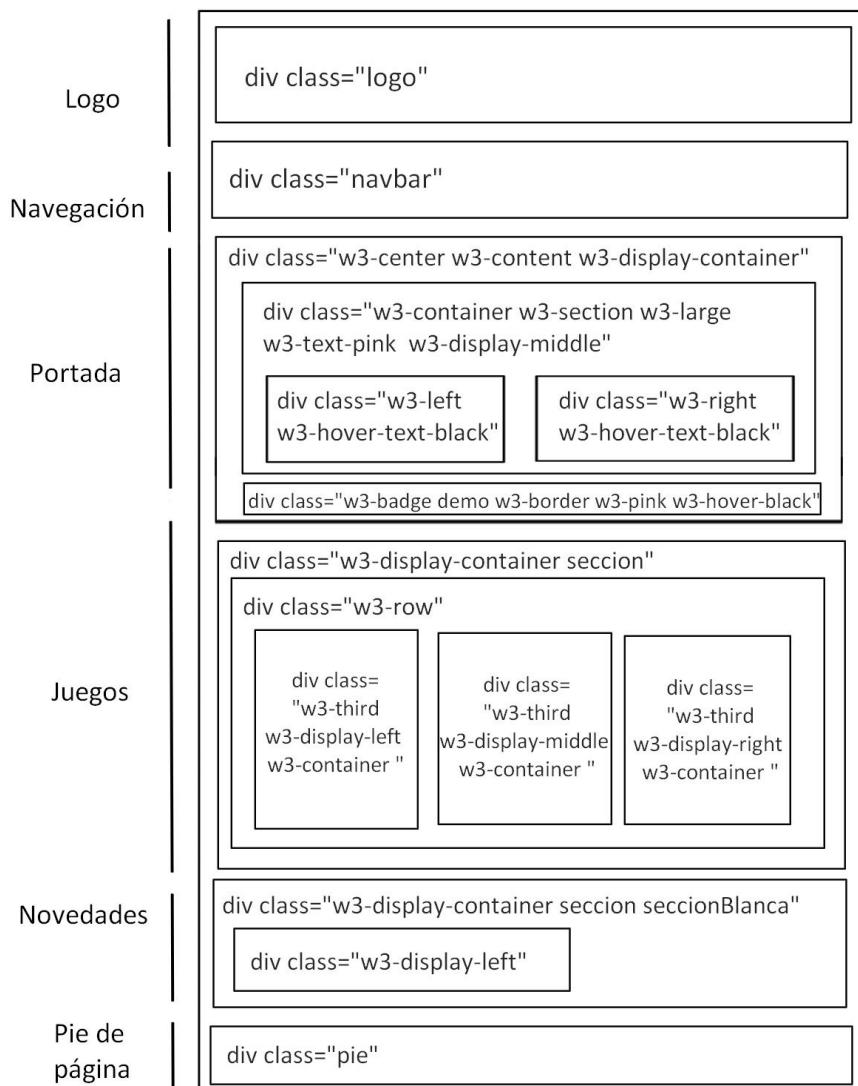


Figura XXXI: Mapa de clases de la página de inicio

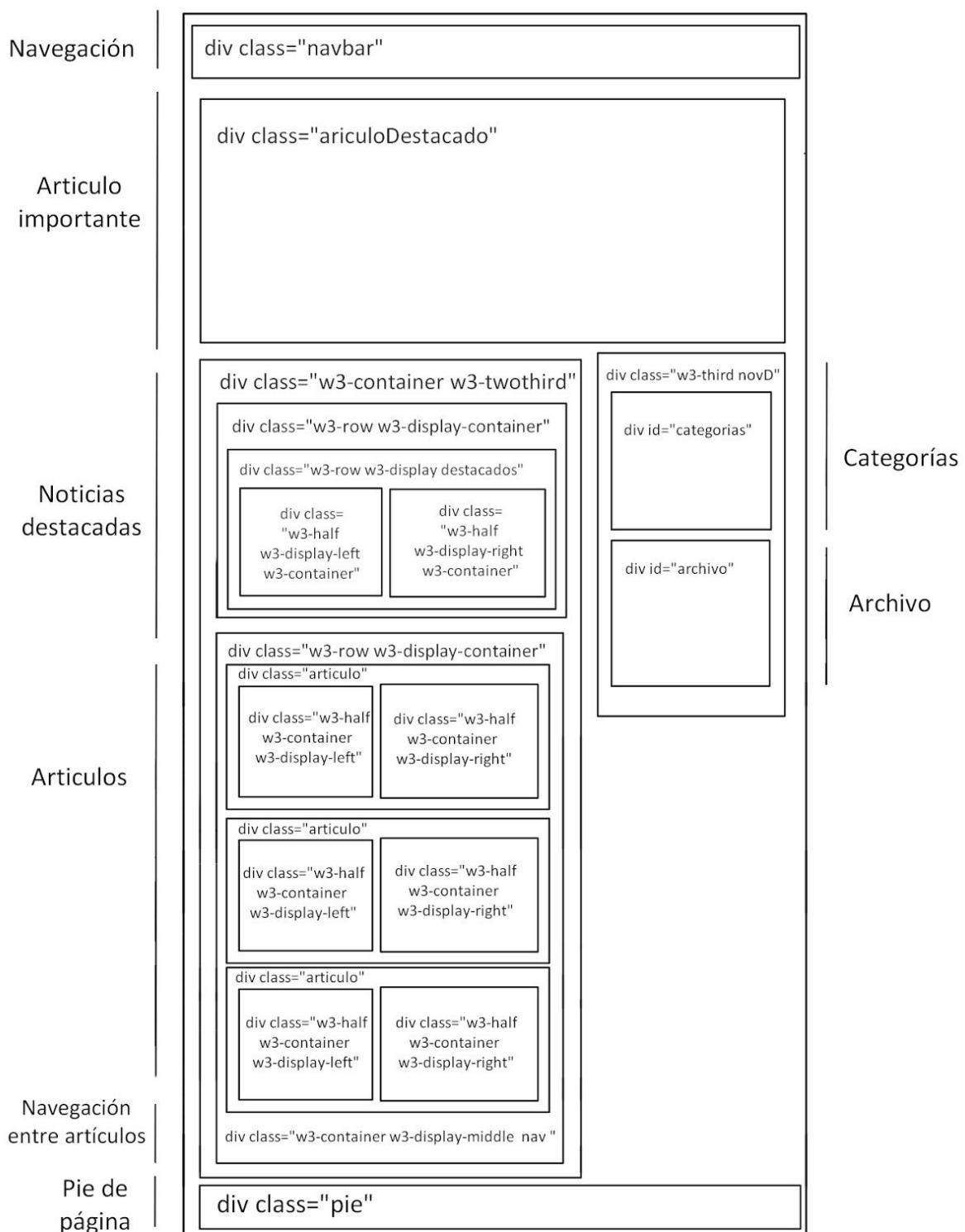


Figura XXXII: Mapa de clases de la página de novedades

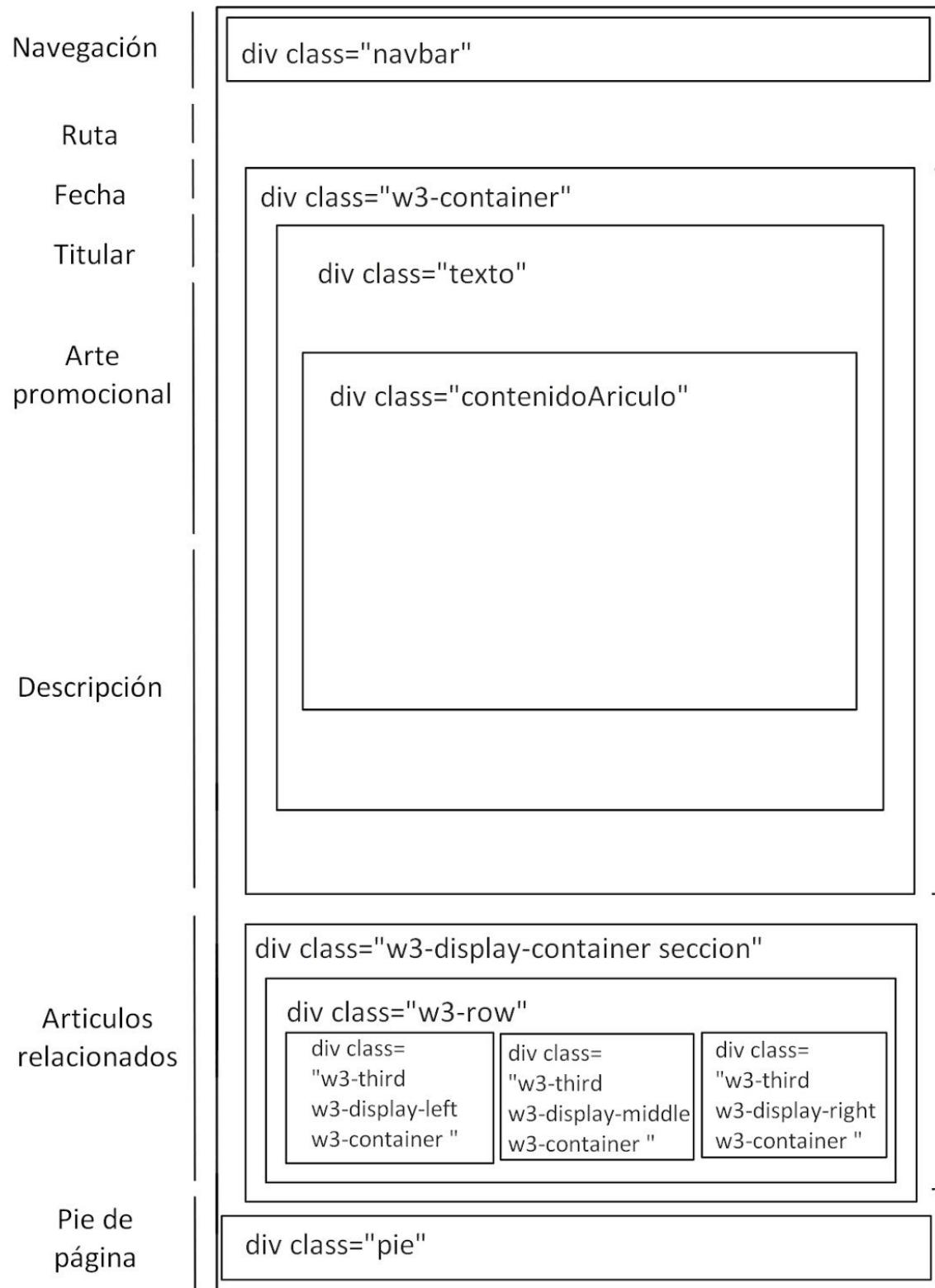


Figura XXXIII: Mapa de clases de un artículo

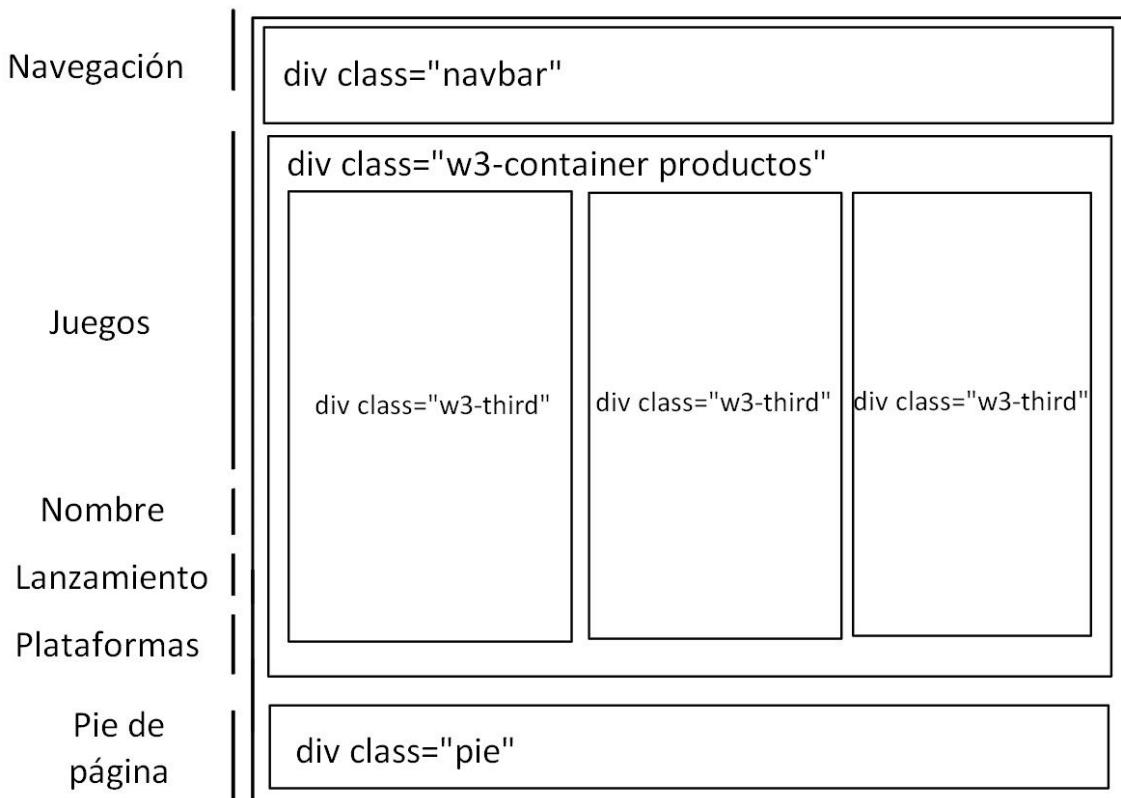


Figura XXXIV: Mapa de clases de la página de productos

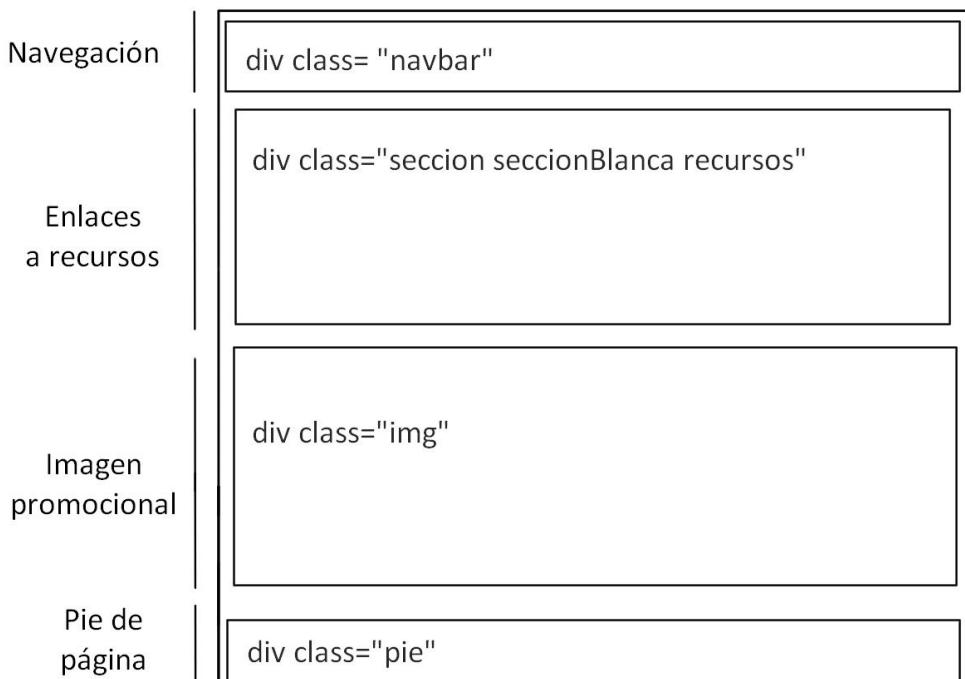


Figura XXXV: Mapa de clases de la página de recursos

Como se puede apreciar en las figuras XXXI a la XXXV, la estructura de todas las páginas mostradas es muy similar. No obstante, se han realizado una serie de cambios con respecto al diseño original para mejorar el aspecto del sitio web.

12. Javascript

Al incluir hojas de estilo en el sitio web de Boundless se ha logrado implementar el diseño mostrado en el Mockup. No obstante, se ha considerado necesario añadir dinamismo a ciertas páginas. Con esto se busca que el usuario sienta que la web reacciona a sus acciones de una forma más visual. Para ello, se ha incluido código JavaScript. El código añadido incorpora 3 nuevos efectos visibles que reaccionan a eventos diferentes y que están caracterizados por el uso de métodos de acceso al DOM.

12.1. Slideshow promocional

El primer script permite la implementación de un slideshow de novedades a la página de inicio. Este contiene tres imágenes que van rotando automáticamente cada 5 segundos. Adicionalmente, el slideshow cuenta con dos flechas que permiten al usuario navegar a su gusto por las diferentes imágenes.

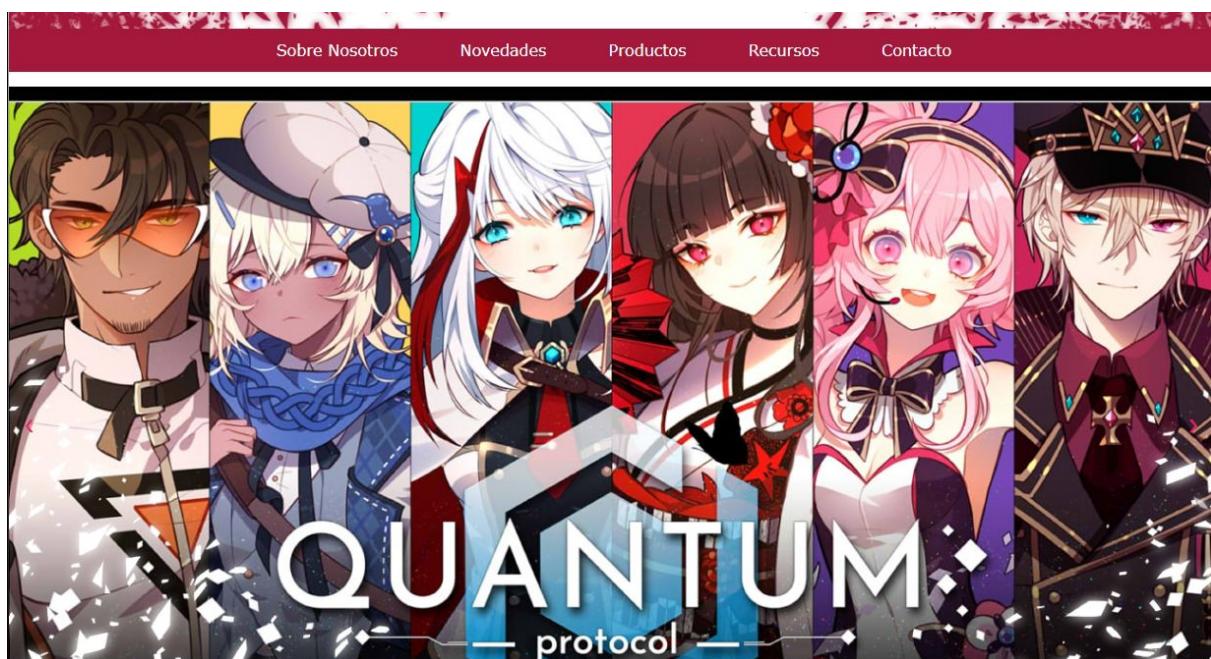


Figura XXXVI: Primera imagen del slideshow

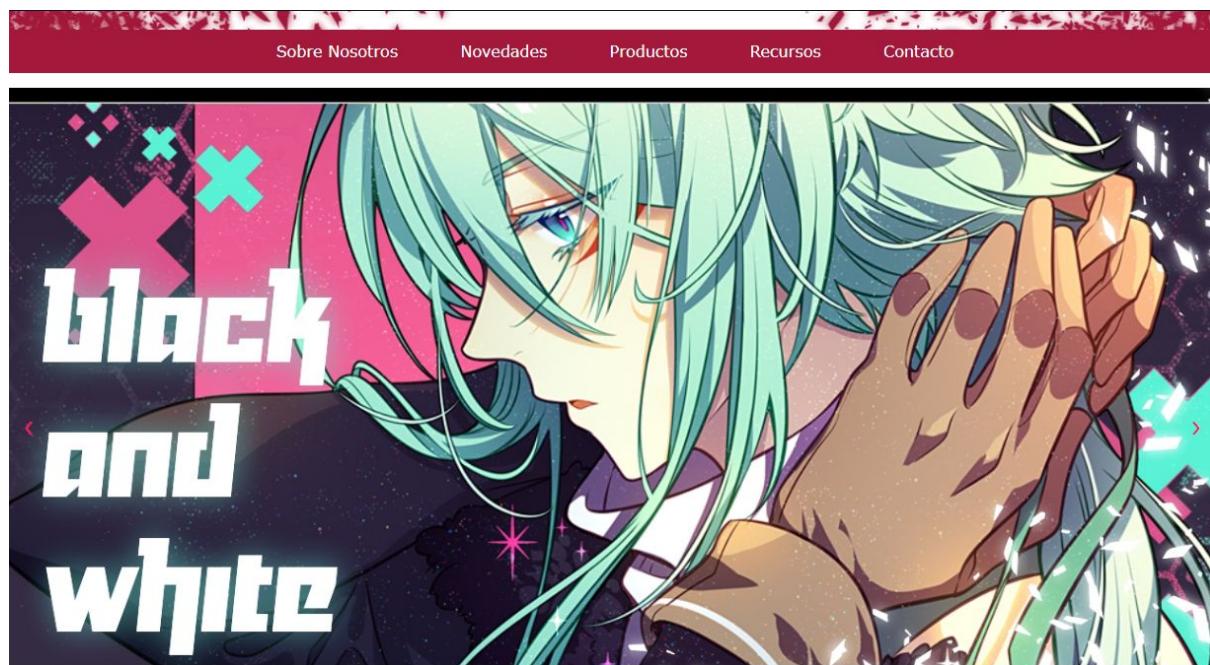


Figura XXXVII: Segunda imagen del slideshow

Las figuras XXXVI y XXXVII representan la transición automática entre dos imágenes del slideshow. Para lograrlo:

```
<!--Script para que funcione el slideshow-->
<script>

    //Variables
    var slideIndex = 1; //Índice de la imagen mostrada
    carousel(); //Función carrusel
    showDivs(slideIndex); //Función para mostrar la primera de las
    imágenes

    //Función para transitar a otra imagen
    function plusDivs(n) {
        showDivs(slideIndex += n);
    }

    //Función que se mantiene en la misma imagen
    function currentDiv(n) {
        showDivs(slideIndex = n);
    }

    //Función que muestra la imagen n del slideshow
    function showDivs(n) {
        var i;
        //Recuperaremos el slideshow con métodos de acceso al DOM
    }
</script>
```

```

var x = document.getElementsByClassName("mySlides ");
//Recuperamos los botones circulares que permite seleccionar
una imagen concreta
var dots = document.getElementsByClassName("demo ");

//Si hemos avanzado desde la última imagen del slideshow
if (n > x.length) {
    //La imagen que mostraremos será la primera
    slideIndex = 1
}
//Por el contrario, si hemos retrocedido desde la primera
if (n < 1) {
    //Entonces la imagen que mostraremos será la última
    slideIndex = x.length
}

//Ocultamos todas las imágenes del slideshow
for (i = 0; i < x.length; i++) {
    x[i].style.display = "none ";
}
//Desmarcamos todos los botones circulares
for (i = 0; i < dots.length; i++) {
    dots[i].className      =      dots[i].className.replace("w3-black ", " ");
}
//Finalmente mostramos la imagen
x[slideIndex - 1].style.display = "block ";
//Y marcamos el botón circular que la representa
dots[slideIndex - 1].className += " w3-black ";
}

//Función que hace que las imágenes que componen el slideshow
transiten automáticamente
function carousel() {
    //Recuperamos el slideshow y los botones circulares
    var i;
    var x = document.getElementsByClassName("mySlides ");
    var dots = document.getElementsByClassName("demo ");
    //Ocultamos todas las imágenes y desmarcamos todos los
botones
    for (i = 0; i < x.length; i++) {
        x[i].style.display = "none ";
    }
}

```

```

        dots[i].className = dots[i].className.replace(" w3-black
", " ");
    }
    //Incrementamos el indice del slideshow
    slideIndex++;
    //evitando que este supere el número de imágenes que lo
componen
    if (slideIndex > x.length) {
        slideIndex = 1
    }
    //Mostramos la imagen que toca y marcamos el botón circular
que la representa
    x[slideIndex - 1].style.display = "block ";
    dots[slideIndex - 1].className += " w3-black ";
    //Establecemos un timer para que esta función se llame a sí
misma cuando pasen 5 segundos. De este modo hacemos que aunque el
usuario no interactúe con el slideshow este siga cambiando
periódicamente las imágenes que muestra
    setTimeout(carousel, 5000);
}
</script>

```

Código I: Funcionamiento del Slideshow

Como se ha visto explicado en el código, el Slideshow tiene una implementación muy sencilla. *showDiv()* es llamada por el usuario cuando el usuario interactúa con las flechas o los puntos de navegación mientras que *carousel()* se llama a sí misma cada 5 segundos para transitar las imágenes. En concreto:

```
<div class="w3-left w3-hover-text-black" onclick="plusDivs(-1)
">&#10094;</div> <!--Flecha de navegación de retroceso-->
```

Código II: Aplicación de la función *plusDivs()*

Como se puede apreciar en el código superior, la flecha de navegación llama a *plusDivs(-1)* cuando se hace click en ella, lo que permite que se muestre la imagen que la antecede en el slideshow. De un modo similar:

```
<span class="w3-badge demo w3-border w3-transparent w3-hover-pink "
onclick="currentDiv(1)" style="margin-bottom: 1em; margin-top: 1em;
"></span> <!--Botones circulares de navegación-->
```

Código III: Aplicación de la función *currentDiv()*

Los puntos de navegación llaman a `currentDiv(n)` para solicitar que se muestre la n -ésima imagen que contiene el Slideshow. Así, en el código de ejemplo se muestra el botón que permite mostrar la primera de las imágenes del Slideshow.

12.2. Highlight de juegos

El segundo evento implementado consiste en resaltar el juego sobre el que el usuario coloca el puntero del ratón.

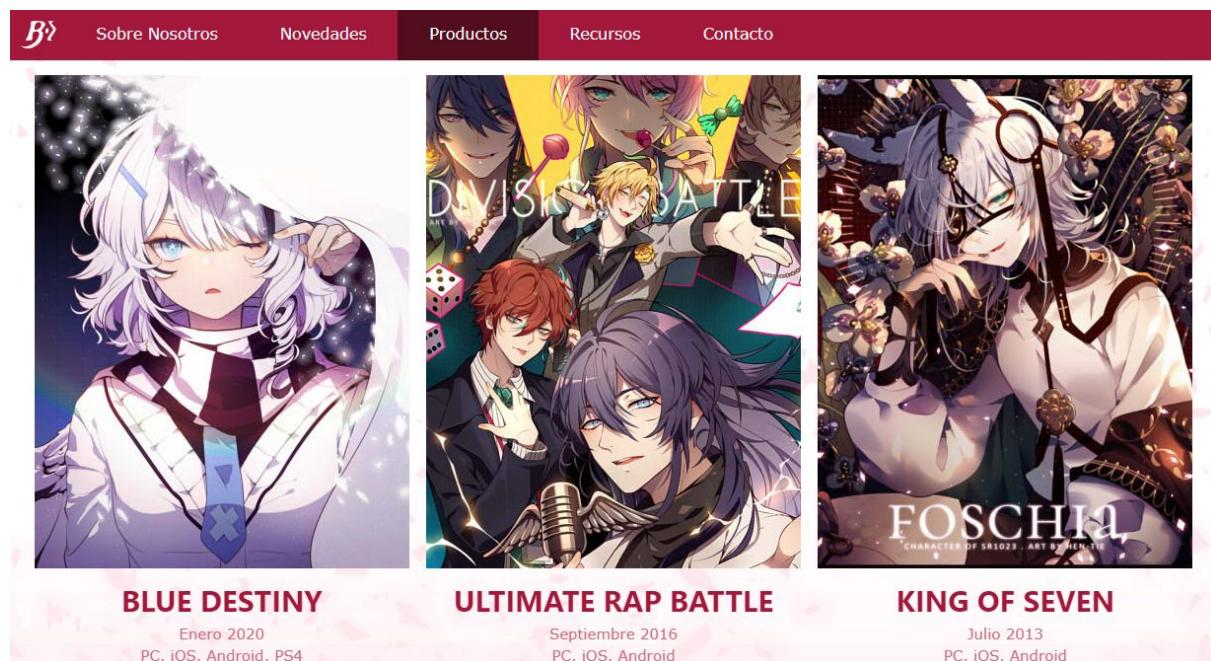


Figura XXXVIII: Página de productos

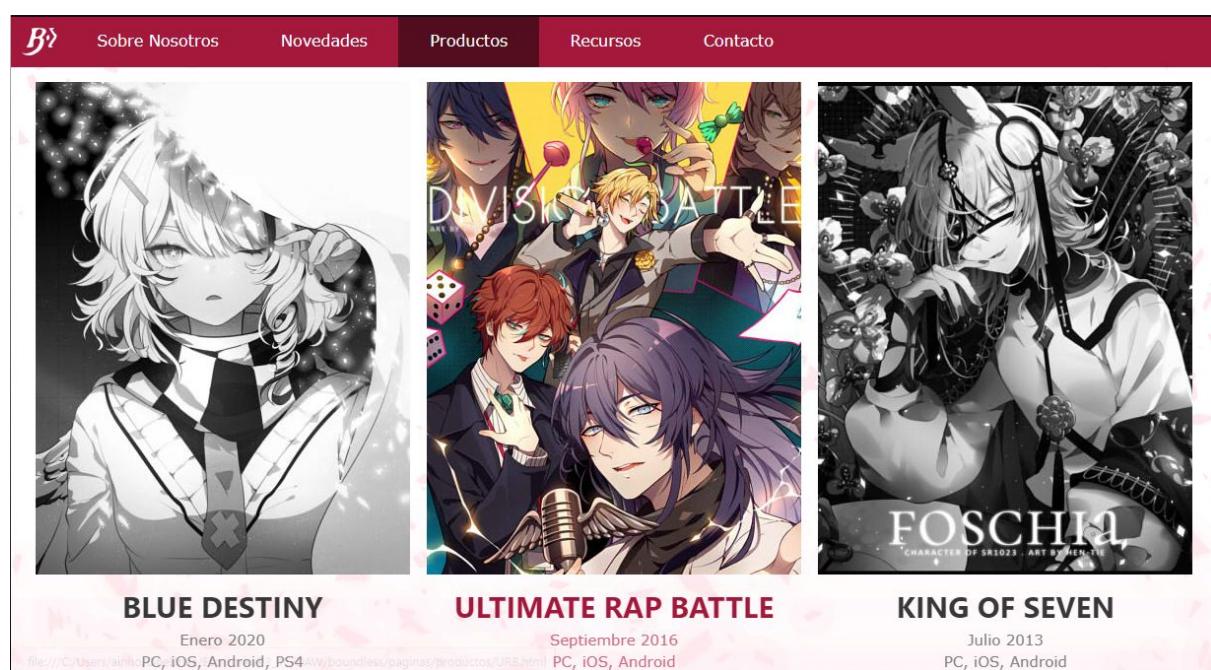


Figura XXXIX: Página de productos con URB destacado

La figura XXXVIII muestra como se vería la página de *Productos* cuando el usuario no ha acercado el ratón a ninguno de los juegos mostrados. En contraposición, la figura XXXIX muestra el resultado de que el usuario coloque el puntero del ratón sobre el juego *Ultimate Rap Battle* de la empresa. El resto de productos de la página pasan a segundo plano a través de un filtro de gris que les quita presencia y logra resaltar a *URB*. Este efecto se ha introducido en el sitio web para llamar la atención del usuario e invitarle a entrar en la página resaltada. Esto se ha logrado implementando dos funciones:

```
<script>
    var juego; //Variable que identifica el juego sobre el cual descansa
    el puntero del ratón del usuario

    //Función que resalta uno de los juegos
    function highlight() {
        //Para ello se aplica un filtro de gris en todos los juegos
        document.getElementById("J1").style.filter = "grayscale(100%)";
        document.getElementById("J2").style.filter = "grayscale(100%)";
        document.getElementById("J3").style.filter = "grayscale(100%)";
        //Y se retira el filtro del juego que le interesa al usuario
        document.getElementById(juego).style.filter = "grayscale(0%)";
    }
    //Función que deshace el highlight
    function fullColor() {
        //Para ello retira el filtro gris de todas las secciones
        document.getElementById("J1").style.filter = "grayscale(0%)";
        document.getElementById("J2").style.filter = "grayscale(0%)";
        document.getElementById("J3").style.filter = "grayscale(0%)";
    }
</script>
```

Código IV: Highlight en la página de productos

Como se ha visto, la función *highlight()* permite destacar la sección de la página dedicada al juego sobre el que el usuario ha colocado su ratón. Para ello se aplica el filtro *grayscale* a todas las secciones y a continuación se retira de la sección escogida por el usuario. Finalmente, cuando el usuario retira el puntero del ratón entonces se quita el filtro de gris de todas las secciones. Para que esto funcione:

```
<div id="juegos" class="w3-container productos w3-animate-opacity"
    ALIGN=center style="margin-top: 1em;" onmouseover="highlight()"
    onmouseout="fullColor()">
```

Código V: Aplicación de las funciones *highlight()* y *fullColor()*

Se debe definir una sección de juegos en la página que contenga todas las secciones de los juegos. A esta sección *juegos* se le aplica la función *highlight()* cuando se pone el puntero del ratón sobre ella y *fullColor()* cuando se retira. Además:

```
<!--Primer juego-->
<div id="J1" class="w3-third promo" onmouseover="juego = this.id">
```

Código VI: Asignación del *id* del elemento a la variable *juego*

En el código superior se muestra que cuando el usuario coloca el ratón sobre esta sección, la variable *juego* toma como valor el *id* del elemento: *J1*.

12.3. Alerta de redirección

El tercer evento gestionado es la redirección a páginas webs externas. Este script solo se ha implementado en la página *Contacto*. El resto de páginas del dominio que redirigen a sitios externos no alertan al usuario. Esto se ha realizado con la finalidad de derivar al usuario al mayor número de dominios diferentes. Para asegurarse de que no abandona la web de Boundless todas las páginas externas se abren en una nueva ventana.

The screenshot shows a website layout. At the top is a dark header bar with a logo on the left and five menu items: "Sobre Nosotros", "Novedades", "Productos", "Recursos", and "Contacto". The "Contacto" item is highlighted with a darker background. Below the header is a section titled "Información de contacto" containing text about contacting for game, service, or event inquiries. It lists four game titles with their official websites and contact forms. At the bottom is a footer bar with links for "Soporte técnico", "Términos del Servicio", "Política de Privacidad", and "Contacto". The footer also includes a copyright notice: "© 2012-2020 Boundless, Inc. Todos los derechos reservados."

Figura XL: Página de contactos

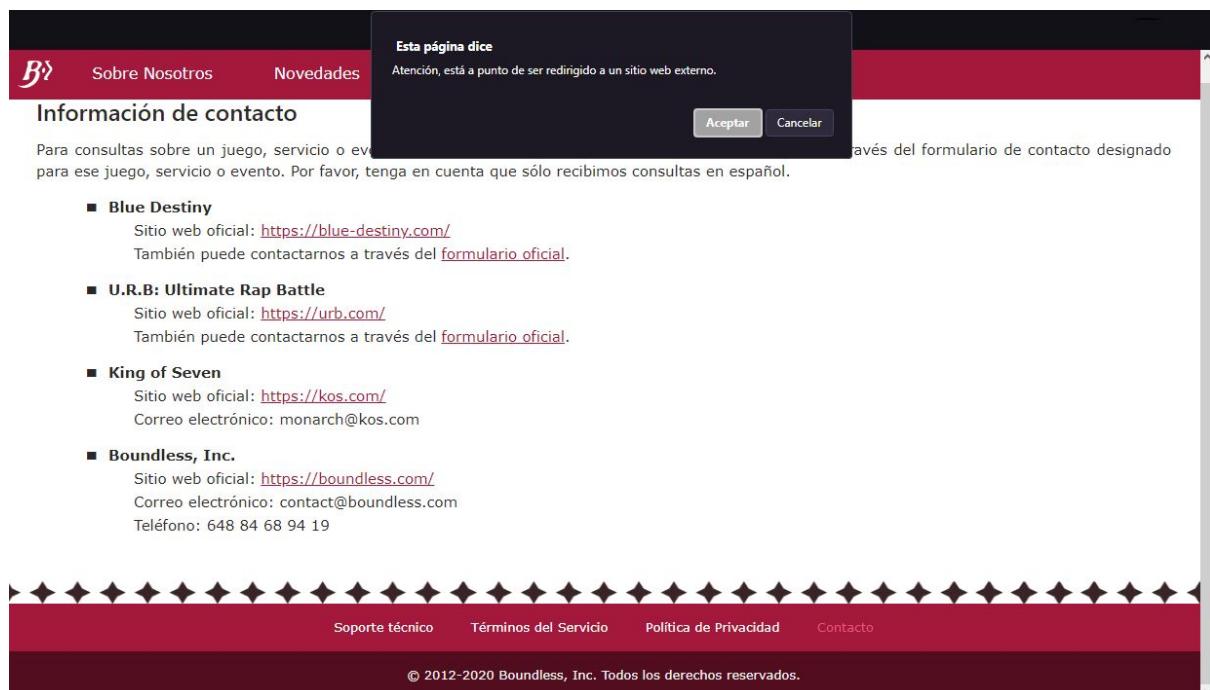


Figura XLI: Página de contactos tras hacer click en un enlace

Las figuras XL y XLI muestran qué ocurre al hacer click en un enlace de la página de contactos. Como se puede observar, al hacerlo salta una ventana emergente que notifica al usuario y le solicita que confirme o niegue la apertura de la página. Si confirma, la página web en la que hizo click con el ratón se abrirá en una ventana nueva. Si lo descarta, la página no se abrirá. Para que esto funcione se ha implementado la función *redirect(link)*:

```
<script>
    function redirect(link) {
        if (confirm("Atención, está a punto de ser redirigido a un sitio web externo.") == true) {
            window.open(link)
        }
    }
</script>
```

Código VII: Implementación de la función de redirección

El código superior muestra el funcionamiento de *redirect(link)*. Esta recibe como argumento un enlace. Crea una ventana emergente de confirmación que explica al usuario que será redirigido. Si este acepta, entonces se abre *link* en una nueva pestaña del navegador. Si rechaza la redirección entonces la ventana emergente se cierra y el usuario permanece en la página de contacto. Esta función debe llamarse:

```
<span class="enlace"
      onclick="redirect('https://global.destiny-child.com')">https://blue-des
tiny.com/</span></p>
```

Código VII: Uso de la función de redirección

Así, cuando se hace click en un elemento de la clase *enlace* (una clase que tiene como objetivo imitar el aspecto de los enlaces del sitio web) se llama a *redirect(link)*, donde *link* es un enlace válido a un sitio web externo de videojuegos.

13. AJAX

El siguiente paso en el desarrollo del sitio web de Boundless consiste en unificar la información en archivos *xml* y *json*. Para ello se utilizará AJAX^[21], *Asynchronous JavaScript And XML*, una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

Se han generado dos ficheros nuevos: *novedades.json* y *productos.xml*. Así, toda la información disponible de los productos de la empresa y los artículos se encuentra recogida en un único lugar. Desde el sitio web se accederá a dichos archivos para recuperar la información que requiera la página. Por tanto, se ha considerado necesario modificar el esquema del sitio web de la siguiente forma:

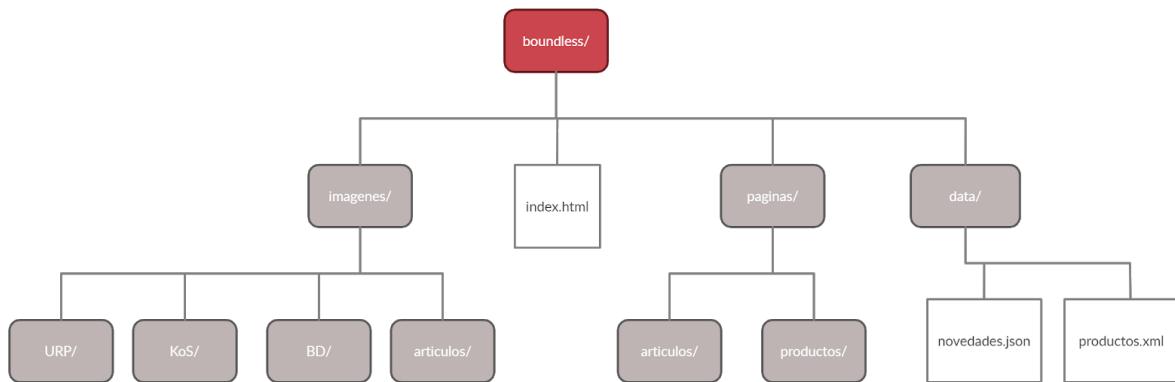


Figura XLII: Estructura de ficheros actualizada de Boundless

Como se puede observar en la figura XLII, se ha incorporado un nuevo directorio denominado *data* en el sitio web. Este contiene los archivos *.xml* y *.json* usados en la web de Boundless. Las páginas afectadas por la incorporación de estos ficheros son:

- **Página principal.** La ruta y nombre de las imágenes usadas para promocionar los juegos en la sección *Nuestros Productos* se obtienen del archivo *productos.xml*

- **Novedades.** El título, imagen, fecha, etiquetas y resumen de las noticias de la sección *Artículos* se obtiene del archivo *novedades.json*
- **Productos.** El título, imagen, fecha de lanzamiento y plataformas de cada juego se obtienen del archivo *productos.xml*
- **Páginas específicas de los juegos (BD, URB y KoS).** Todos los datos recogidos en la sección *Información técnica* de cada páginas, los banner y los títulos de cada juego se obtienen del archivo *productos.xml*
- **Estado de los servidores.** El título, imagen, estado del servidor y reporte de funcionamiento del servidor de cada juego se obtienen del archivo *productos.xml*
- **Contacto.** El título, información de contacto y enlaces para cada juego se obtienen del archivo *productos.xml*

De esta forma se logra evitar repetir los mismos datos en diversas páginas, previniendo así las incoherencias en el contenido.

Para que la lectura de los ficheros de extensión *xml* y *json* funcione se ha utilizado el *servidor HTTP Apache*. Para ello, se ha introducido el sitio web completo en el servidor por lo que la dirección de la página de inicio ahora es <http://localhost/Boundless/index.html>.

13.1. XML

Como se ha especificado, el sitio web accede a la información del fichero *productos.xml* desde diversas páginas. Para ello, debe implementar dos funciones. Por ejemplo, en la página de *Productos* se accede a ella del siguiente modo:

```
<!--AJAX-->
<script>

    function loadDoc() {
        var xhttp = new XMLHttpRequest();

        xhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                myFunction(this);
            }
        };
        xhttp.open("GET", "/Boundless/data/productos.xml", true);
        xhttp.send();
    }

    //Función para establecer la información de los productos
    function myFunction(xml) {
        var xmlDoc = xml.responseXML;
        //Primer juego
```

```

        var J1 = xmlDoc.getElementsByTagName("BD");
                var j1 = "<li>" +
J1[0].getElementsByTagName("TITULO")[0].childNodes[0].nodeValue +
"</li>" + '<p>Sitio web oficial: <span class="enlace" onclick=' +
"redirect('" +
J1[0].getElementsByTagName("WEB_OFICIAL_E")[0].childNodes[0].nodeValue +
"')'" +'">' +
J1[0].getElementsByTagName("WEB_OFICIAL_N")[0].childNodes[0].nodeValue +
'+ '</span></p><p>También puede contactarnos a través del <span
class="enlace" onclick=' +'"redirect('" +
J1[0].getElementsByTagName("FORM")[0].childNodes[0].nodeValue +
"')'" + '>formulario oficial</span>.</p>';
                document.getElementById("J1").innerHTML = j1;

        //Segundo juego
        var J2 = xmlDoc.getElementsByTagName("URB");
                var j2 = "<li>" +
J2[0].getElementsByTagName("TITULO")[0].childNodes[0].nodeValue +
"</li>" + '<p>Sitio web oficial: <span class="enlace" onclick=' +
"redirect('" +
J2[0].getElementsByTagName("WEB_OFICIAL_E")[0].childNodes[0].nodeValue +
"')'" +'">' +
J2[0].getElementsByTagName("WEB_OFICIAL_N")[0].childNodes[0].nodeValue +
'+ '</span></p><p>También puede contactarnos a través del <span
class="enlace" onclick=' +'"redirect('" +
J2[0].getElementsByTagName("FORM")[0].childNodes[0].nodeValue +
"')'" + '>formulario oficial</span>.</p>';
                document.getElementById("J2").innerHTML = j2;

        //Tercer juego
        var J3 = xmlDoc.getElementsByTagName("KoS");
                var j3 = "<li>" +
J3[0].getElementsByTagName("TITULO")[0].childNodes[0].nodeValue +
"</li>" + '<p>Sitio web oficial: <span class="enlace" onclick=' +
"redirect('" +
J2[0].getElementsByTagName("WEB_OFICIAL_E")[0].childNodes[0].nodeValue +
"')'" +'">' +
J3[0].getElementsByTagName("WEB_OFICIAL_N")[0].childNodes[0].nodeValue +
'+ '</span></p><p>Correo electrónico: ' +
J3[0].getElementsByTagName("FORM")[0].childNodes[0].nodeValue + '</p>';
                document.getElementById("J3").innerHTML = j3;
        }
    </script>

```

Código VIII: Implementación de lectura y recuperación de datos de un fichero .xml

El código superior muestra cómo se ha recuperado la información de contacto del archivo *productos.xml* (disponible en el directorio */Boundless/data/*). Primero se ha implementado la función *loadDoc()* que se llama cuando el cuerpo de la página se carga:

```
<body onload="loadDoc();">
```

Código IX: Uso de la función de carga del fichero en el cuerpo del documento

Esta función permite abrir el archivo .xml y preparar al servidor para que recupere la información que necesita de él a través de la función *myFunction(xml)*. Este fichero contiene la información sobre los juegos estructurada de la siguiente forma:

```
<!--SEGUNDO JUEGO (URB)-->
<URB>
    <TITULO>U.R.B: Ultimate Rap Battle</TITULO>
    <FECHA>14 de Septiembre de 2016</FECHA>
    <LANZAMIENTO>Septiembre 2016</LANZAMIENTO>
    <GENERO>Musical, Gacha, Narrativo, CCG</GENERO>
    <COSTE>Gratis (incluye compras in-game)</COSTE>
    <REQUISITOS>4GB RAM, 8GB almacenamiento disponible</REQUISITOS>
    <PLATAFORMA>PC, iOS, Android</PLATAFORMA>
    <IMAGEN>imagenes/URB/UltimateRapBattleSmall.jpg</IMAGEN>
    <BANNER>imagenes/URB/URBBanner.jpg</BANNER>
    <WEB>productos/vistaURB.jsp</WEB>
    <WEB_OFICIAL_N>https://urb.com/</WEB_OFICIAL_N>
    <WEB_OFICIAL_E>https://hypnosismic.com</WEB_OFICIAL_E>
    <FORM>https://hypnosismic.com/contact/</FORM>
    <SERVER>ONLINE</SERVER>
    <S_MENSAJE>Sin incidencias</S_MENSAJE>
</URB>
```

Código X: Ejemplo del contenido del fichero .xml

Por tanto, para acceder al atributo TITULO de este juego se deberá utilizar:

```
var xmlDoc = xml.responseXML;
var J2 = xmlDoc.getElementsByTagName("URB"); //Recuperación del juego
J2[0].getElementsByTagName("TITULO")[0].childNodes[0].nodeValue
//Recuperación del contenido de un tag
```

Código XI: Ejemplo de recuperación de información del fichero .xml

Como se puede observar, para acceder a la información de los ficheros .xml se utilizan los tags.

13.2. JSON

Aunque la mayoría de las páginas que acceden a ficheros obtienen la información que necesitan de *productos.xml*, la página *Novedades* la recupera del fichero *novedades.json*:

```
<!--AJAX-->
<script>

    function loadDoc() {
        var xmlhttp = new XMLHttpRequest();
        xmlhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                myFunction(this);
            }
        };
        xmlhttp.open("GET", "/Boundless/data/novedades.json",
true);
        xmlhttp.send();
    }

    //Función para establecer la información de los artículos
    function myFunction(json) {
        var myObj = JSON.parse(json.responseText);

        //Primer artículo
        var a1 = '<div class = "w3-half w3-container
w3-display-topleft" style="margin:3em; margin-top: 8%;">' +
'<a href = "' + myObj.articulos[0].enlace + '"><img src = "../' +
myObj.articulos[0].imagen + '"width = "80%"> </a> </div >' +
'<div class = "w3-half w3-container w3-display-topright"
style="margin:3em; margin-top: 8%;">' + '<a href = "' +
myObj.articulos[0].enlace + '"style = "text-decoration: none;"><h3
style = "color: #a30d33;font-weight: 600;">' +
myObj.articulos[0].titulo + '</h3></a><p style = "color: #c4617a;">' +
myObj.articulos[0].fecha + '</p> <p style = "color: #c4617a;">' +
myObj.articulos[0].contenido + '</p> </p> </div>';
        document.getElementById("A1").innerHTML = a1;
    }
}
```

```
//Segundo artículo

                var a2 = '<div class = "w3-half w3-container
w3-display-left" style="margin:3em; margin-top: -5%;">' +
'<a href = "' + myObj.articulos[1].enlace + '" ><img src = "../' +
myObj.articulos[1].imagen + '"width = "80%" > </a> </div >' +
'<div class = "w3-half w3-container w3-display-right" style =
"margin:3em; margin-top: -5%;">' + '<a href = "' +
myObj.articulos[1].enlace + '"style = "text-decoration: none;"><h3
style = "color: #a30d33;font-weight: 600;" >' +
myObj.articulos[1].titulo + '</h3></a><p style = "color: #c4617a;">' +
myObj.articulos[1].fecha + '</p><p style = "color: #c4617a;">' +
myObj.articulos[1].contenido + '</p> <p style = "color: #c4617a;">' +
myObj.articulos[1].tags; + '</p> </div>';

                document.getElementById("A2").innerHTML = a2;

//Tercer artículo

                var a3 = '<div class = "w3-half w3-container
w3-display-left" style="margin:3em; margin-top: 25%;">' +
'<a href = "' + myObj.articulos[2].enlace + '" ><img src = "../' +
myObj.articulos[2].imagen + '"width = "80%" > </a> </div >' +
'<div class = "w3-half w3-container w3-display-right"
style="margin:3em; margin-top: 25%;">' + '<a href = "' +
myObj.articulos[2].enlace + '"style = "text-decoration: none;"><h3
style = "color: #a30d33;font-weight: 600;" >' +
myObj.articulos[2].titulo + '</h3></a><p style = "color: #c4617a;">' +
myObj.articulos[2].fecha + '</p><p style = "color: #c4617a;">' +
myObj.articulos[2].contenido + '</p> <p style = "color: #c4617a;">' +
myObj.articulos[2].tags; + '</p> </div>';

                document.getElementById("A3").innerHTML = a3;
            }
</script>
```

Código XII: Implementación de lectura y recuperación de datos de un fichero .json

Como se puede observar, la función *loadDoc()* no ha sufrido variaciones con respecto a su homónima en la apertura de archivos .xml. La única diferencia entre ambas es el fichero que solicitan. No obstante, la información de dicho archivo se recupera de una forma completamente distinta. Por ejemplo, si el contenido del mismo es:

```

        "contenido": "Únete en este evento a los guerreros del
inframundo para derrortar a la Reina Diane y obtén jugosas
recompensas.",
        "juego": "Ultimate Rap Battle",
        "fecha": "19.07.2020",
        "tags": "#Evento #URB #Canciones",
        "enlace": "vistaProximamente.jsp",
        "imagen": "imagenes/articulos/inferno.jpg"

    }
]

}

```

Código XIII: Ejemplo de contenido del fichero .json

Entonces para acceder al enlace del único artículo que contiene se hará:

```

var myObj = JSON.parse(json.responseText);
var enlace = myObj.articulos[0].enlace //archivo.elemento[n].atributo

```

Código XIV: Ejemplo de recuperación de información del fichero .json

Como se puede observar, la información se recupera utilizando el esquema *archivo.elemento[n].atributo*.

14. JQuery

Tras introducir dinamismo en el sitio web mediante javascript, se ha decidido incrementar la interacción con el usuario. Para ello, en esta ocasión se implementarán una serie de efectos programados utilizando JQuery^[21], una biblioteca multiplataforma de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web. Dos de las numerosas funciones implementadas⁹ con JQuery son:

14.1. Highlight de juegos

El primer de los efectos implementado con JQuery se encuentra en la página principal, en concreto, en la sección de *Nuestros Juegos*. Este consiste es consiste en resaltar la portada del juego sobre el que el usuario coloca el puntero del ratón¹⁰.

⁹ Muchas de las funcionalidades implementadas con JQuery no se comentarán en esta sección al utilizarse para gestionar errores en el servidor.

¹⁰ El mismo efecto se había implementado en la página de *Productos* utilizando JavaScript y métodos de acceso al DOM.

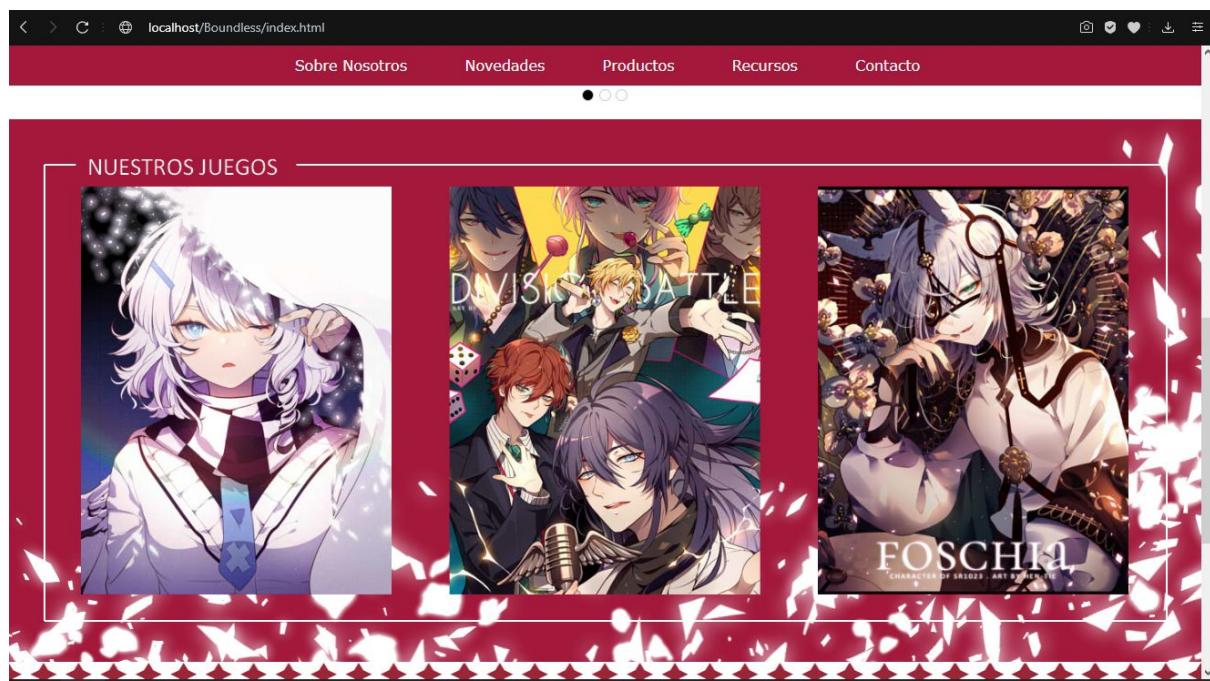


Figura XLIII: Sección de Nuestros Productos en la página de inicio

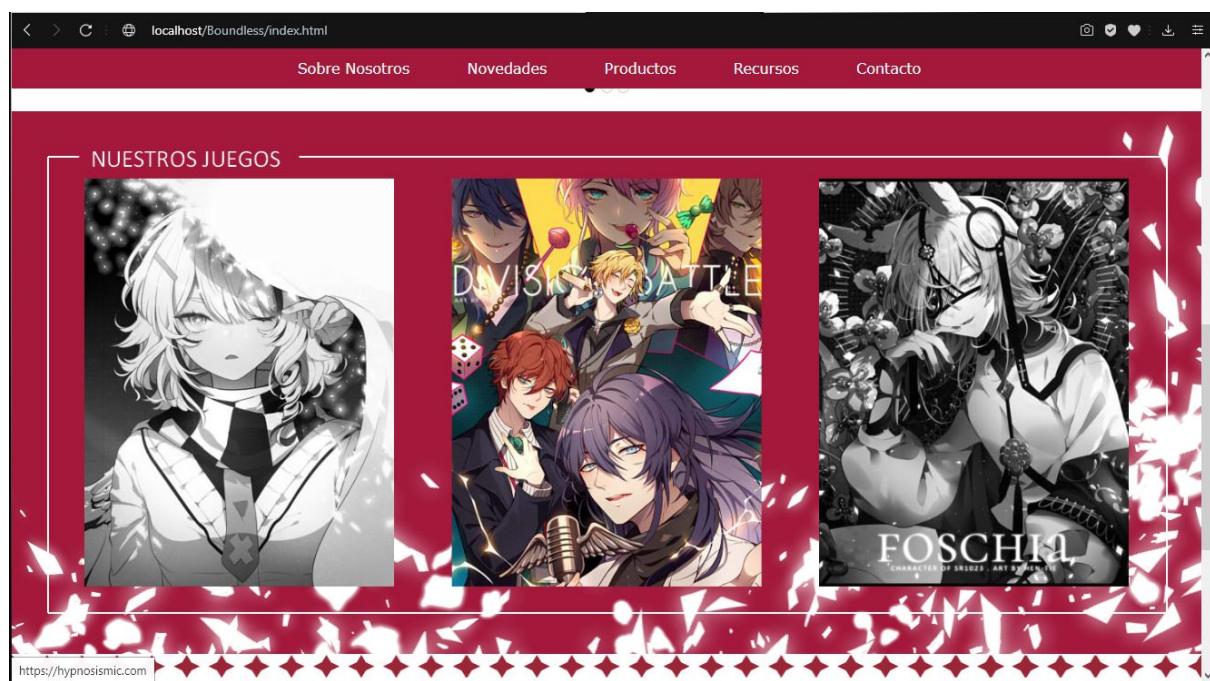


Figura XLIV: Sección de Nuestros Productos en la página de inicio con URB destacado

La figura XLIII muestra cómo se vería la sección de *Nuestros Juegos* cuando el usuario no ha acercado el ratón a ninguna de las imágenes mostradas. En contraposición, la figura XLIV muestra el resultado de que el usuario coloque el puntero del ratón sobre el juego *Ultimate Rap Battle*. El resto de productos de la página pasan a segundo plano a través de un filtro de gris que les quita presencia y logra resaltar a URB. Este efecto se ha introducido en el sitio web para llamar la atención del usuario e invitarle a entrar en la página resaltada.

El código utilizado para implementar este efecto es el siguiente:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
    //Cuando la página está preparada
    $(document).ready(function() {
        //Aplicamos un filtro de gris a los elementos de la
        //clase "promo" cuando se coloca el ratón sobre ella
        $(".promo").mouseover(function() {
            $(".promo").css("filter", "grayscale(100%)");
        });
        //Quitamos el filtro de gris cuando se coloca el ratón
        //sobre el elemento "BD"
        $("#BD").mouseover(function() {
            $("#BD").css("filter", "grayscale(0%)");
        });
        $("#URB").mouseover(function() {
            $("#URB").css("filter", "grayscale(0%)");
        });
        $("#KoS").mouseover(function() {
            $("#KoS").css("filter", "grayscale(0%)");
        });
        $(".promo").mouseout(function() {
            $(".promo").css("filter", "grayscale(0%)");
        });
    });
</script>
```

Código XV: Highlight en la página de Inicio

Como se puede observar en el código I, el script programado aplica un filtro de escala de grises a los elementos de la clase *promo* cuando el usuario pone el puntero del ratón sobre ella. A continuación, se invierte el filtro aplicado para el elemento concreto sobre el que se encuentre el ratón. Finalmente, al desplazar el ratón fuera de los elementos de *promo*, se invierte el filtro de grises aplicado para todos los elementos.

14.2. Menú desplegable

El segundo evento permite la implementación de un sencillo menú desplegable en la sección *Archivo* de la página de *Novedades* que muestra u oculta los meses del 2020 en los que se han publicado artículos.

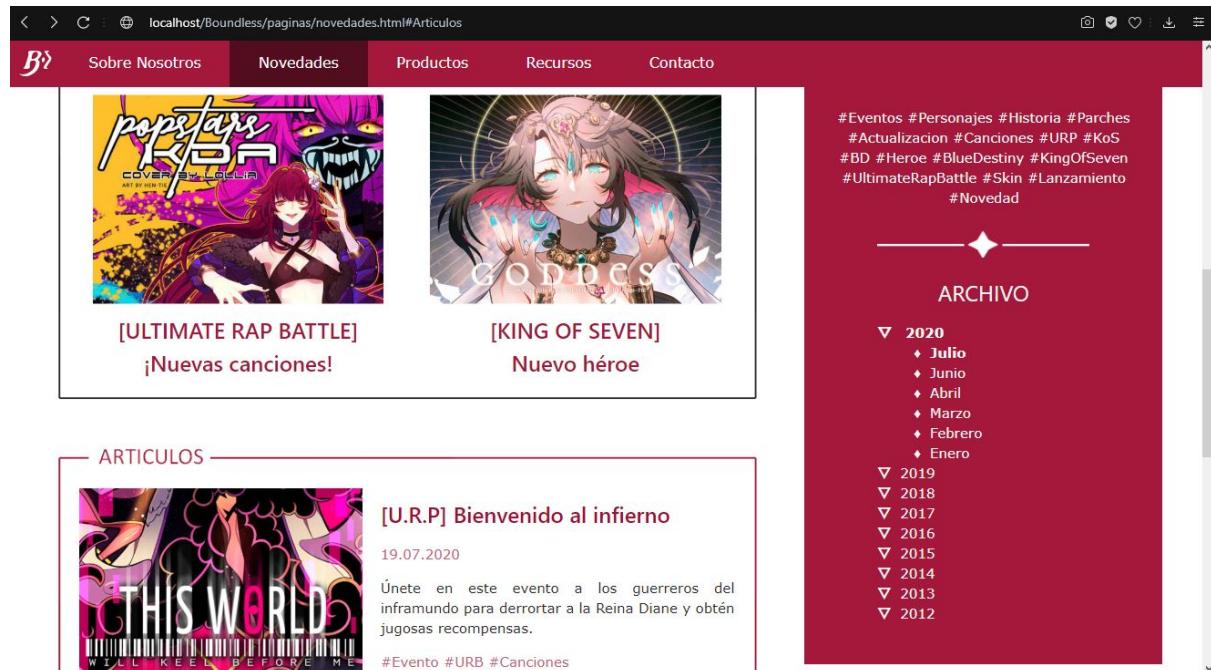


Figura XLV: Menú desplegado

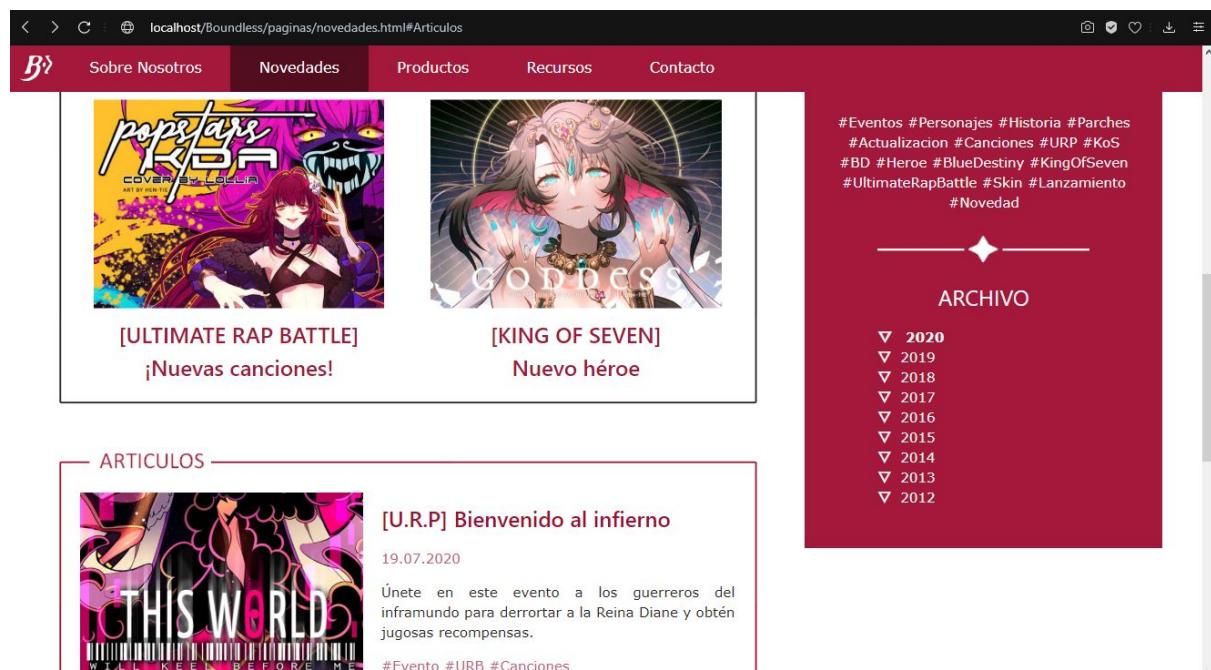


Figura XLVI: Menú oculto

Las figura XLV muestra el aspecto de la web cuando el listado de meses está visible y la figura XLVI el aspecto cuando está oculto. Por defecto, el menú está desplegado al cargar la página. Adicionalmente, se ha modificado el aspecto del puntero para indicar al usuario que 2020 es un elemento sobre el que se puede hacer click. El código programado es el siguiente:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js">
</script>
<script>
    $(document).ready(function() {
        //Modificamos el aspecto del cursor para que el usuario sepa
        que es un elemento clickable
        $("#2020").mouseover(function() {
            $("#2020").css("cursor", "pointer")
        });
        $("#2020").mouseout(function() {
            $("#2020").css("cursor", "default")
        });
        //Cuando se hace click en el elemento de id "2020" se
        despliegan todos los elementos que sean de la clase "mes"
        $("#2020").click(function() {
            $(".mes").slideToggle("slow",
                function() {
                    // Animation complete.
                });
        });
    });
</script>
```

Código XVI: Highlight en la página de Inicio

En primer lugar, se puede ver en el código II cómo el script programado gestiona el aspecto del puntero del ratón dependiendo de si este está sobre el elemento 2020 o no (*mouseover* y *mouseout*, respectivamente) . Además, se hace click en dicho elemento, se muestra u oculta el menú dependiendo de su visibilidad actual (*slideToggle*).

15. JSP, Java Beans, EL, JSTL y JDBC

El último paso en el desarrollo del sitio web de Boundless consiste en implementar un sistema de login y registro, además de incorporar una tienda online con

retroalimentación a una base de datos externa. Para ello se utilizará el patrón MVC para estructurar la aplicación:

- **Vistas:** HTML, JSP, EL y JSTL.
- **Controlador:** Servlets
- **Modelo:** Javabeans y clases Java.

La aplicación será desplegada en Apache TomCat. Se han generado numerosos ficheros nuevos *.java* y *.jsp* además de un fichero *web.xml*. Desde el sitio web se accederá a dichos archivos para recuperar de la base de datos la información que requiera cada página. Por tanto, se ha considerado necesario modificar el esquema del sitio web de la siguiente forma:

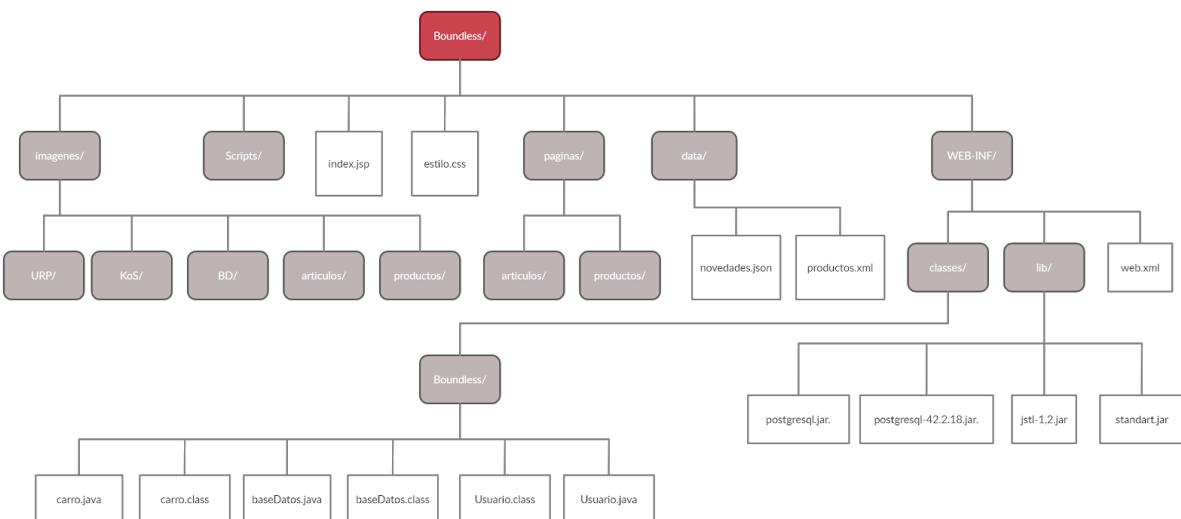


Figura XLVII: Estructura final de ficheros simplificada de Boundless

15.1. Implementación de la base de datos

En primer lugar, se creó la base de datos de *Boundless*, la cual se encargará de almacenar toda la información necesaria para gestionar los usuarios del sitio web, los productos en venta y los pedidos que se realizan:

```
-- Database: Boundless
-- DROP DATABASE "Boundless";

CREATE DATABASE "Boundless"
WITH
OWNER = postgres
ENCODING = 'UTF8'
LC_COLLATE = 'Spanish_Spain.1252'
LC_CTYPE = 'Spanish_Spain.1252'
TABLESPACE = pg_default
CONNECTION LIMIT = -1;
```

Código XVII: Script de creación de la base de datos

A continuación, se generaron las diferentes tablas que se necesitan para llevar a cabo todas las funcionalidades requeridas. Estas son *usuario*, *pedido* y *mercancia*.

```
-- Table: public.usuario

-- DROP TABLE public.usuario;

CREATE TABLE public.usuario
(
    apellido1 character varying(45) COLLATE pg_catalog."default",
    correo character varying(150) COLLATE pg_catalog."default" NOT NULL,
    direccion character varying(60) COLLATE pg_catalog."default",
    nombre character varying(45) COLLATE pg_catalog."default" NOT NULL,
    password character varying(45) COLLATE pg_catalog."default" NOT NULL,
    sexo character(1) COLLATE pg_catalog."default" NOT NULL,
    telf character varying(12) COLLATE pg_catalog."default",
    apellido2 character varying(45) COLLATE pg_catalog."default",
    "fechaNacimiento" date NOT NULL,
    pais character varying(3) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT usuario_pkey PRIMARY KEY (correo)
)

TABLESPACE pg_default;

ALTER TABLE public.usuario
OWNER to postgres;

-- SEQUENCE: public.pedido_id

-- DROP SEQUENCE public.pedido_id;

CREATE SEQUENCE public.pedido_id
CYCLE
INCREMENT 1
START 1
MINVALUE 0
MAXVALUE 100
CACHE 1;
```

```

ALTER SEQUENCE public.pedido_id
    OWNER TO postgres;

-- Table: public.pedido

-- DROP TABLE public.pedido;

CREATE TABLE public.pedido
(
    usuario character varying(200) COLLATE pg_catalog."default" NOT
NULL,
    fecha date NOT NULL,
    contenido character varying(1000) COLLATE pg_catalog."default" NOT
NULL,
    id numeric NOT NULL DEFAULT nextval('pedido_id'),
    CONSTRAINT pedido_pkey PRIMARY KEY (id),
    CONSTRAINT usuario FOREIGN KEY (usuario)
        REFERENCES public.usuario (correo) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)
TABLESPACE pg_default;

ALTER TABLE public.pedido
    OWNER to postgres;

-- Table: public.mercancia

-- DROP TABLE public.mercancia;

CREATE TABLE public.mercancia
(
    nombre character varying(200) COLLATE pg_catalog."default" NOT
NULL,
    img character varying(200) COLLATE pg_catalog."default",
    cantidad integer NOT NULL,
    precio numeric NOT NULL,
    CONSTRAINT mercancia_pkey PRIMARY KEY (nombre)
)

```

```

TABLESPACE pg_default;

ALTER TABLE public.mercancia
OWNER to postgres;

```

Código XVIII: Script de creación de tablas

Como se puede observar en el Código XVIII la tabla de usuario contiene información sobre el nombre, primer apellido, segundo apellido, dirección, país, fecha de nacimiento, sexo, teléfono, correo electrónico y contraseña de un usuario. Por otra parte, la tabla de mercancía incluye información sobre la descripción, cantidad en stock, imagen y precio de cada producto. La tabla de pedidos incluye un ID, fecha y contenido del pedido además del correo electrónico del usuario que lo ha realizado. No todos los atributos son obligatorios en las tablas. Aquellos que necesitan introducirse en la base de datos son los marcados con NOT NULL. Finalmente, se introdujeron datos en la base:

```

INSERT INTO public.mercancia(nombre, img, cantidad, precio)
VALUES ('U.R.B: Llavero de K-ING', 'imagenes/mercancia/llavero.jpg',
'0', '19.99');

INSERT INTO public.mercancia(nombre, img, cantidad, precio)
VALUES ('BD: Taza Silveth', 'imagenes/mercancia/peluche.jpg',
'2', '24.99');

INSERT INTO public.mercancia(nombre, img, cantidad, precio)
VALUES ('KoS: Camiseta blanca STAIN',
'imagenes/mercancia/camiseta.jpg', '27', '44.99');

INSERT INTO public.usuario(apellido1, correo, nombre, password, sexo,
apellido2, "fechaNacimiento", pais)
VALUES ('Vivel', 'ainhoa.vivel@rai.usc.es', 'Ainhoa', 'password',
'f', 'Cousu', '2000-03-07', 'ES');

```

Código XIX: Script de inserción

De este modo, se ha creado e inicializado con éxito la base de datos del sitio web.

15.2. Login y registro

La primera de las funcionalidades implementada fue el sistema de login y registro. Este permite crear una cuenta de usuario en el sitio web que se almacena en la base datos o iniciar sesión en el sitio web si la cuenta ya está creada. Todo este sistema se gestiona desde

vistaAcceso.jsp, quien redirige las peticiones del usuario a los servlets *login* y *registro*. A esta se accede haciendo click en el botón *Acceso* desde cualquier de las páginas de la web.



Figura XLVIII: Aspecto de la página de inicio (*index.jsp*)



Figura XLIX: Aspecto de la página de Acceso (*vistaAcceso.jsp*) con el formulario de logueo

En la Figura XLVIII se puede ver el aspecto de la página de acceso. En ella se encuentra visible un formulario de inicio de sesión y otro oculto de registro. Para regresar a la página de inicio se hará click en el logo del sitio web, mientras que para visualizar el formularios de registro habrá que hacer click en *¿Aún no tienes cuenta?*. Los formularios se han implementado:

```
<!--Formulario de logueo-->
<div id="formularioLoggeo" class="w3-display-right w3-half w3-animate-opacity w3-center">
    <div class="w3-round" style="background-color: white; margin-right: 5em; padding: 1em; box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19); text-align: center;">
        <h1 style="margin-bottom: 1px; color:#a6001a; font-weight: bold;">Inicia sesión</h1>
        <form id="form2" name="log" method="POST" action="/Boundless/login">
```

```

<div class="w3-container w3-col m12 w3-animate-opacity w3-center">
    <input id="cL" class="w3-input w3-padding-16" type="email"
maxlength="100" pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}".$
placeholder="Correo electrónico*" required name="correo">
</div>
<div class="w3-container w3-col m12 w3-animate-opacity w3-center"
style="margin-bottom: 1em;">
    <input id="pL" class="w3-input w3-padding-16" type="password"
maxlength="20" placeholder="Contraseña*" name="password" required>
</div>
<p id="merrorLogin" style="display: none">El correo y/o la contraseña
introducidos no son válidos</p>
    <button class="w3-button w3-round w3-section w3-text-white
w3-highway-red w3-hover-black w3-block" style="font-size: large;" type="submit"> Entrar </button>
</form>
<a id="registro" style=" text-decoration: underline;">¿Aún no tienes
cuenta? Regístrate</a>
</div>
</div>

<!--Formulario de registro-->
<div id="formularioResgistro" class="w3-display-right w3-half
w3-animate-opacity w3-center" style="display: none;">
    <div class="w3-round" style="background-color: white; margin-right: 5em;
padding: 1em; box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0
rgba(0, 0, 0, 0.19); text-align: center;">
        <h1 style="margin-bottom: 1px; color:#a6001a; font-weight:
bold;">Registro</h1>
        <p style="margin-top: 0;">Es muy rápido y sencilo</p>
        <form name="reg" method="POST" action="/Boundless/registro">
            <div class="w3-row">
                <div class="w3-container w3-col m4 12 w3-animate-opacity
w3-center" style="padding: 0;">
                    <input class="w3-input w3-padding-16" type="text"
maxlength="45" pattern="[a-zA-Zá-úÁ-Ú]{2,}" placeholder="Nombre*" required
name="nombre">
                </div>
                <div class="w3-container w3-col m4 12 w3-animate-opacity
w3-center">
                    <input class="w3-input w3-padding-16" type="text"
maxlength="45" pattern="[a-zA-Zá-úÁ-Ú]{2,}" placeholder="Primer apellido"
name="apellido1">
                </div>
                <div class="w3-container w3-col m4 12 w3-animate-opacity
w3-center" style="padding-left: 0;">

```

```

                <input class="w3-input w3-padding-16" type="text"
maxlength="45" pattern="[a-zA-Zá-úÁ-Ú]{2,}" placeholder="Segundo apellido"
name="apellido2">
            </div>
        </div>

        <div class="w3-row">

            <div class="w3-container w3-col m6 12 w3-animate-opacity
w3-center" style="padding: 0;">
                <input id="cR" class="w3-input w3-padding-16"
type="email"                                     maxlength="100"
pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}+$"    placeholder="Correo
electrónico*" required name="correo">
            </div>
            <div class="w3-container w3-col m6 12 w3-animate-opacity
w3-center">
                <input id="pR" class="w3-input w3-padding-16"
type="password"      maxlength="20"      placeholder="Contraseña*"      required
name="password">
            </div>
        </div>

        <div class="w3-row">
            <div class="w3-container w3-col m6 12" style="padding-left:
0;">
                <input class="w3-input w3-padding-16" type="date"
placeholder="Fecha" required name="fecha" value="2020-12-22">
            </div>
            <div class="w3-container w3-col m6 12" style="padding-left:
0; margin-top: 1.5em;">
                <input class=" w3-padding-16" type="radio"
id="hombre" name="gender" value="h" checked>
                <label for="hombre"> Hombre</label>
                <input class=" w3-padding-16" type="radio" id="mujer"
name="gender" value="m">
                <label for="mujer"> Mujer</label>
                <input class="w3-padding-16" type="radio" id="otro"
name="gender" value="o">
                <label for="otro"> Otro</label>
            </div>
        </div>
        <div class="w3-row">
            <div class="w3-container w3-col m6 12" style="padding-left: 0;">
                <input class="w3-input w3-padding-16" type="tel"
maxlength="12"      pattern="[0-9]{3} [0-9]{2} [0-9]{2} [0-9]{2}"
placeholder="Teléfono - 648 84 68 19" name="telf">

```

```

                </div>
            <div class="w3-container w3-col m6 12" style="padding: 0; padding-right: 1em;" style="margin-bottom: 1em;">
                <select class="w3-input w3-round w3-padding-16" required id="Pais" name="pais">
                    <option value="AF">Afganistán</option>
                    ...
                    <option value="ES" selected>España</option>
                    ...
                    <option value="ZW">Zimbabue</option>
                </select>
            </div>
        </div>

        <div class="w3-container w3-col m12 12 w3-animate-opacity w3-center" style="padding: 0; padding-right: 1em;">
            <input class="w3-input w3-padding-16" type="text" maxlength="150" placeholder="Dirección" name="direccion">
        </div>

        <label class="w3-input w3-padding-16 w3-col m12"><input type="checkbox" required id="checkbox" value=""> He leído y acepto los <a href="/Boundless/paginas/vistaToS.jsp" target="_blank">ToS</a> y la <a href="/Boundless/paginas/vistaPrivacidad.jsp" target="_blank">Política de privacidad</a></label>
        <c:if test="${!empty param.errorRegistro}">
            <p style="margin-top: 1em">El correo introducido ya está asociado a un cuenta existente.</p>
        </c:if>
        <button class="w3-button w3-round w3-section w3-text-white w3-highway-red w3-hover-black w3-block" type="submit">Registrarse</button>
    </form>
    <a id="login" style=" text-decoration: underline;">¿Ya tienes cuenta? Inicia sesión</a>
</div>
</div>
<!--Control de visibilidad de los formularios y mensajes de error--&gt;
&lt;c:choose&gt;
    &lt;c:when test="${!empty param.errorLogin}"&gt;
        &lt;script&gt;
            $("#merrorLogin").css("display", "inline");
            $("#merrorRegistro").css("display", "none");
        &lt;/script&gt;
    &lt;/c:when&gt;
&lt;/c:choose&gt;
</pre>

```

```

<c:choose>
    <c:when test="#{!empty param.errorRegistro}">
        <script>
            $("#formularioLoggeo").css("display", "none");
            $("#formularioResgistro").css("display", "inline");
        </script>
    </c:when>
</c:choose>

```

Código XX: Formularios de login y registro

Como se puede ver en el código, los formularios limitan el contenido de algunos de sus campos. Este es el caso del teléfono, nombre, apellidos y correo.

```

<!--Fragmento del input teléfono-->
pattern="[0-9]{3} [0-9]{2} [0-9]{2} [0-9]{2}" placeholder="Teléfono - 648 84 68 19"
<!--Fragmento del input nombre-->
pattern="[a-zA-Zá-úÁ-Ú]{2,}" placeholder="Nombre*" required placeholder = "Nombre*"
<!--Fragmento del input correo electrónico-->
pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-z]{2,}$" placeholder = "Correo electrónico*"

```

Código XXI: Ejemplo de control de inputs en los campos de los formularios

De este modo, si se intenta introducir una dirección de correo no válida:

The screenshot shows a login form with the title "Inicia sesión". There are two input fields: one for email and one for password. The email field contains "correo@prueba". Above the password field, there is an error message: "Utiliza un formato que coincida con el solicitado". Below the password field is a red button labeled "Entrar". At the bottom of the form, there is a link "¿Aún no tienes cuenta? Regístrate".

Figura L: Aspecto de la página de Acceso al introducir un correo no válido en el campo

Además, como se ha visto en el Código XX, los formularios también especifican que determinados campos son obligatorios utilizando *required*. Los campos obligatorios son consistentes con la base de datos. Uno de ellos es, por ejemplo, la contraseña. Así, si intentamos enviar el formulario de ingreso de sesión sin introducir la contraseña:

The screenshot shows a light blue header with the text "Inicia sesión". Below it is a white input field containing the email "correo@prueba.es". Underneath is another input field labeled "Contraseña*" which is empty. A red button below these fields has a yellow exclamation mark icon and the text "Completa este campo" pointing to the empty password field. At the bottom of the form is a link "¿Aún no tienes cuenta? Regístrate".

Figura LI: Aspecto de la página de Acceso al intentar enviar el formulario de logueo sin cubrir todos los campos obligatorios

Por el contrario, si se introduce un correo electrónico que no está registrado en la base de datos o una contraseña que no coincide con la almacenada para esa dirección, entonces salta un mensaje de error:

The screenshot shows a light blue header with the text "Inicia sesión". Below it are two input fields: one for "Correo electrónico*" and one for "Contraseña*". Between them is a message "El correo y/o la contraseña introducidos no son válidos". Below these is a large red "Entrar" button. At the bottom of the form is a link "¿Aún no tienes cuenta? Regístrate".

Figura LII: Aspecto de la página de Acceso al introducir valores no almacenados en la base de datos o que no se corresponden con los existentes

Esto se ha logrado al implementar la clase *login.java*:

```
// Obtenemos el correo de usuario a partir de la petición
String correo = request.getParameter("correo");
// Obtenemos la contraseña a partir de la petición
String password = request.getParameter("password");

// Creamos una nueva sesión
HttpSession session = request.getSession(true);

// Comprobamos que existe
baseDatos BD = new baseDatos();
String nombreUsuario = BD.validarAcceso(correo, password);
// Si se encuentra al usuario
if(nombreUsuario != null){
    //Añadimos el usuario a la sesión
    session.setAttribute("usuario", nombreUsuario);
    session.setAttribute("correoUsuario", correo);
    // Presentamos los datos
    gotoPage("/Boundless/index.jsp", response);
}
else{
    //Marcamos el error
    gotoPage("/Boundless/paginas/vistaAcceso.jsp?errorLogin=1",
response);
}
```

Código XXII: Funcionamiento del servlet de logueo

Donde:

```
private void gotoPage(String address, HttpServletResponse response)
throws ServletException, IOException {
    response.sendRedirect(address);
}
```

Código XXIII: Función de redirección a páginas del servidor

Finalmente, si se introducen unos datos válidos que se correspondan con algunos de los almacenados en la base de datos el usuario será redirigido automáticamente al *index.jsp* y arriba a la derecha podrá ver que el botón de *Acceso* ha sido sustituido por uno de *Cerrar Sesión*, precedido por el nombre del usuario:



Figura LIII: Aspecto de la página de inicio

Si se hace click en *Cerrar sesión*, el usuario será redirigido a la página de inicio pero ya no estará logueado en el sistema. Para conseguirlo se ha implementado la clase *cerrar.java*:

```
// Creamos una nueva sesión
HttpSession session = request.getSession(true);

// Borramos los datos del usuario
session.setAttribute("usuario", null);
session.setAttribute("correoUsuario", null);

// Redirigimos
gotoPage("/Boundless/index.jsp", response);
}
```

Código XXIV: Funcionamiento del cierre de sesión

La otra funcionalidad recogida dentro de la página de Acceso es el registro:

The screenshot shows the 'Registro' (Registration) form on the Boundless website. The form is titled 'Registro' and describes itself as 'Es muy rápido y sencillo'. It contains fields for 'Nombre*', 'Primer apellido', and 'Segundo apellido'. There are also fields for 'Correo electrónico*' and 'Contraseña*'. Below these are date and gender selection fields ('22/12/2020' and radio buttons for 'Hombre', 'Mujer', 'Otro'). Further down are fields for 'Teléfono - 648 84 68 19' and 'España' (with a dropdown arrow). A 'Dirección' field is present, though empty. At the bottom, there is a checkbox for accepting terms and conditions and a link to the privacy policy. A large red 'Registrarse' button is at the bottom right, and a link to existing accounts is at the bottom left.

Figura LIV: Aspecto de la página de Acceso (vistaAcceso.jsp) con el formulario de registro

El funcionamiento del formulario de registro es muy similar al de logueo. El usuario deberá cubrir los datos obligatorios que están indicados por un asterisco y aceptar los ToS y la Política de privacidad de la empresa marcando la casilla de verificación.

Registro
Es muy rápido y sencillo

Nombre*	Primer apellido	Segundo apellido
<input type="text"/> ! Completa este campo		Contraseña*
22/12/2020	<input type="button" value=""/>	<input checked="" type="radio"/> Hombre <input type="radio"/> Mujer <input type="radio"/> Otro
Teléfono - 648 84 68 19	España	<input type="button" value=""/>
Dirección		
<input type="checkbox"/> He leído y acepto los ToS y la Política de privacidad		
Registrarse		
¿Ya tienes cuenta? Inicia sesión		

Figura LV: Formulario de registro cuando no se completan los campos obligatorios

Registro
Es muy rápido y sencillo

Ainhoa	Primer apellido	Segundo apellido
ainhoa.vivel@rai.usc.es	*****	
22/12/2020	<input type="button" value=""/>	<input checked="" type="radio"/> Hombre <input type="radio"/> Mujer <input type="radio"/> Otro
Teléfono - 648 84 68 19	España	<input type="button" value=""/>
Dirección		
<input type="checkbox"/> He leído y acepto los ToS y la Política de privacidad		
<input type="checkbox"/> ! Selecciona esta casilla de verificación si quieres continuar		
Registrarse		
¿Ya tienes cuenta? Inicia sesión		

Figura LVI: Formulario de registro cuando no se selecciona la casilla obligatoria

Por otra parte, si un usuario intenta registrarse en la página web con un correo electrónico que ya está almacenada en la base de datos entonces salta un error y no se realiza el registro:

The screenshot shows a registration form titled "Registro" (Registration) with the subtitle "Es muy rápido y sencillo" (It's very quick and simple). The form fields include:

- Nombre* (Name*)
- Primer apellido (First name)
- Segundo apellido (Last name)
- Correo electrónico* (Email address*)
- Contraseña* (Password*)
- Fecha de nacimiento (Date of birth): 22/12/2020
- Género (Gender): Hombre Mujer Otro
- Teléfono - 648 84 68 19
- Lugar (Place): España
- Dirección (Address)
- A checkbox labeled "He leído y acepto los [Tos](#) y la [Política de privacidad](#)" (I have read and accept the [Tos](#) and the [Privacy Policy](#))
- A message below the checkbox: "El correo introducido ya está asociado a una cuenta existente." (The email entered is already associated with an existing account.)
- A red button labeled "Registrarse" (Register).
- A link at the bottom: "¿Ya tienes cuenta? Inicia sesión" (Do you have an account? Log in).

Figura LVII: Aspecto de la página cuando un usuario intenta registrarse con una cuenta de correo electrónico que ya está en la base de datos

Esto se ha conseguido:

```
// Obtenemos el correo de usuario a partir de la petición
String correo = request.getParameter("correo");
// Obtenemos la contraseña a partir de la petición
String password = request.getParameter("password");

// Creamos una nueva sesión
HttpSession session = request.getSession(true);

// Comprobamos que existe
baseDatos BD = new baseDatos();
```

```

String nombreUsuario = BD.validarRegistro(correo);

if(nombreUsuario == null){

    // Obtenemos el resto de datos
    Usuario usuario = new Usuario();

    usuario.setNombre(request.getParameter("nombre"));
    usuario.setApellido1(request.getParameter("apellido1"));
    usuario.setApellido2(request.getParameter("apellido2"));
    usuario.setCorreo(correo);
    usuario.setPassword(password);

    usuario.setFechaNacimiento(Date.valueOf(request.getParameter("fecha")));
;

    usuario.setSexo(request.getParameter("gender"));
    usuario.setTelf(request.getParameter("telf"));
    usuario.setPais(request.getParameter("pais"));
    usuario.setDireccion(request.getParameter("direccion"));

    BD.registrarUsuario(usuario);

    //Creamos el usuario
    session.setAttribute("usuario", usuario.getNombre());
    session.setAttribute("correoUsuario", correo);
    // Presentamos los datos
    gotoPage("/Boundless/index.jsp", response);
}

else{
    //Marcamos el error
    gotoPage("/Boundless/paginas/vistaAcceso.jsp?errorRegistro=error",
response);
}

```

Código XXV: Funcionamiento del registro

Finalmente, si el registro se realiza con éxito el usuario inicia sesión automáticamente en el sitio web y es redirigido a la página de inicio.

15.3. Tienda

La siguiente funcionalidad a implementar es la tienda de merchandising de Boundless. Esta incluye información sobre los productos que el usuario puede comprar.

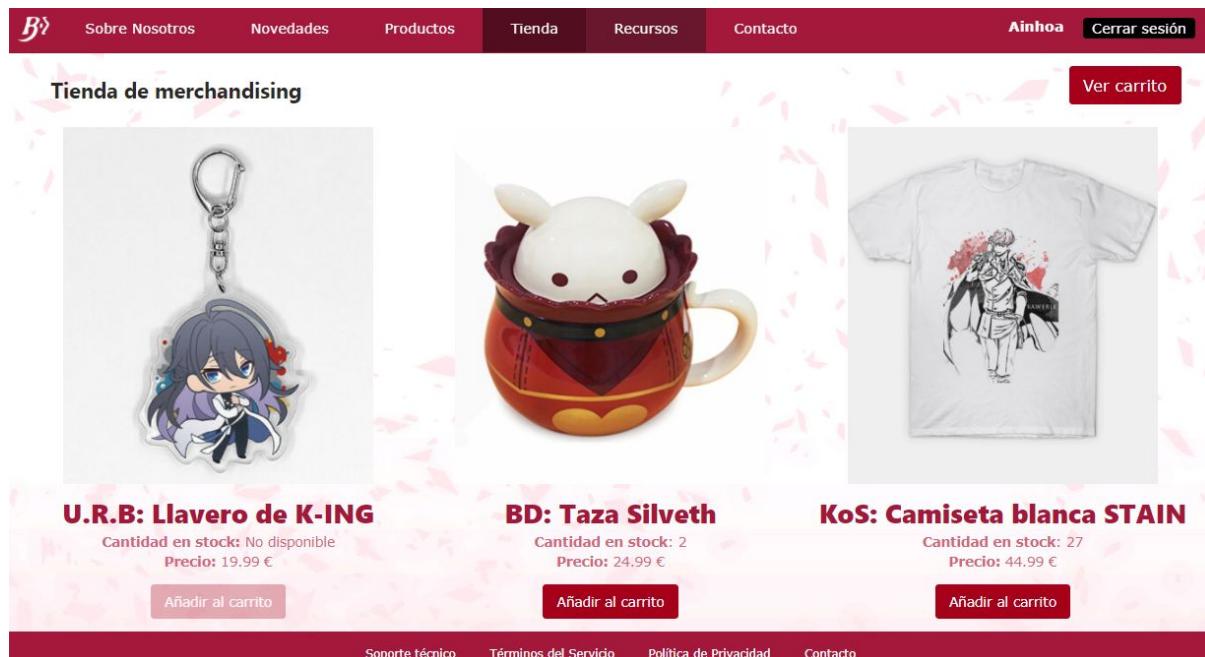


Figura LVIII: Aspecto de la página de la Tienda

Como se puede observar en la Figura LVIII, la página contiene una imagen, descripción, cantidad en stock y precio para cada producto a la venta. Además, debajo de cada producto se incluye un botón que permite añadirlo automáticamente al carro. Tras añadirlo, será redirigido a ver su carro. El usuario puede también acceder al carro a través de *Ver carrito* sin necesidad de añadir ningún producto a su cesta. La página se actualiza con la información de la base de datos cada vez que se accede a ella. Esto se ha conseguido:

```
<!--Productos de la empresa-->
<div class="w3-animate-opacity">
    <div class="w3-row">
        <div class="w3-half">
            <h3 style="margin: 1em; margin-left: 2em; font-weight:bold;">Tienda de merchandising</h3>
        </div>

        <div class="w3-half" style="margin-top: 1em">
            <a href="/Boundless/paginas/vistaCarro.jsp" class="w3-button w3-round w3-text-white w3-highway-red w3-hover-black w3-right" style="font-size: large; margin: 0; margin-right: 2em">Ver carrito</a>
        </div>
    </div>
    <div class="w3-row w3-container productos">
        <!--Para cada producto en venta (recuperado de la base de datos)-->
    </div>
</div>
```

```

<c:forEach var="Mercancia" items="${vectorMercancia}">
    <div class="w3-third">
        <center>
            <br>
            <h2><b>${Mercancia.descripcion}</b></h2>

            <c:choose>
                <c:when test="${Mercancia.cantidad == 0}">
                    <p><b>Cantidad en stock:</b> No disponible </p>
                </c:when>
                <c:otherwise>
                    <p><b>Cantidad en stock</b>: ${Mercancia.cantidad}</p>
                </c:otherwise>
            </c:choose>
            <p><b>Precio:</b> ${Mercancia.precio} €</p>
            <form method="post" action="/Boundless/carro" style="margin-top: 1em; margin-bottom: 1em;">
                <input type="hidden" name="cantidad" value="1">
                <input type="hidden" name="stock" value="${Mercancia.cantidad}">
                <input type="hidden" name="descripcionMercancia" value="${Mercancia.descripcion}">
                <c:choose>
                    <!--Si no hay stock impedimos su compra-->
                    <c:when test="${Mercancia.cantidad == 0}">
                        <input class="w3-button w3-round w3-text-white w3-highway-red w3-hover-black" type="submit" value="Añadir al carrito" disabled>
                    </c:when>
                    <!--Si hay stock permitimos su compra-->
                    <c:otherwise>
                        <input class="w3-button w3-round w3-text-white w3-highway-red w3-hover-black" type="submit" value="Añadir al carrito" >
                    </c:otherwise>
                </c:choose>
            </form>
        </center>
    </div>
</c:forEach>
</div>
</div>

```

Código XXVI: Funcionamiento de la vista de la tienda

Finalmente, como se puede observar en el código, si un producto no está en stock en estos momentos su cantidad se muestra como *No disponible* y se deshabilita el botón de *Añadir al carrito*.

15.4. Carro

Tras haber implementado la tienda es necesario implementar también un carrito al que el usuario pueda acceder para ver los productos que ha seleccionado. El carrito está disponible en *vistaCarro.jsp*:

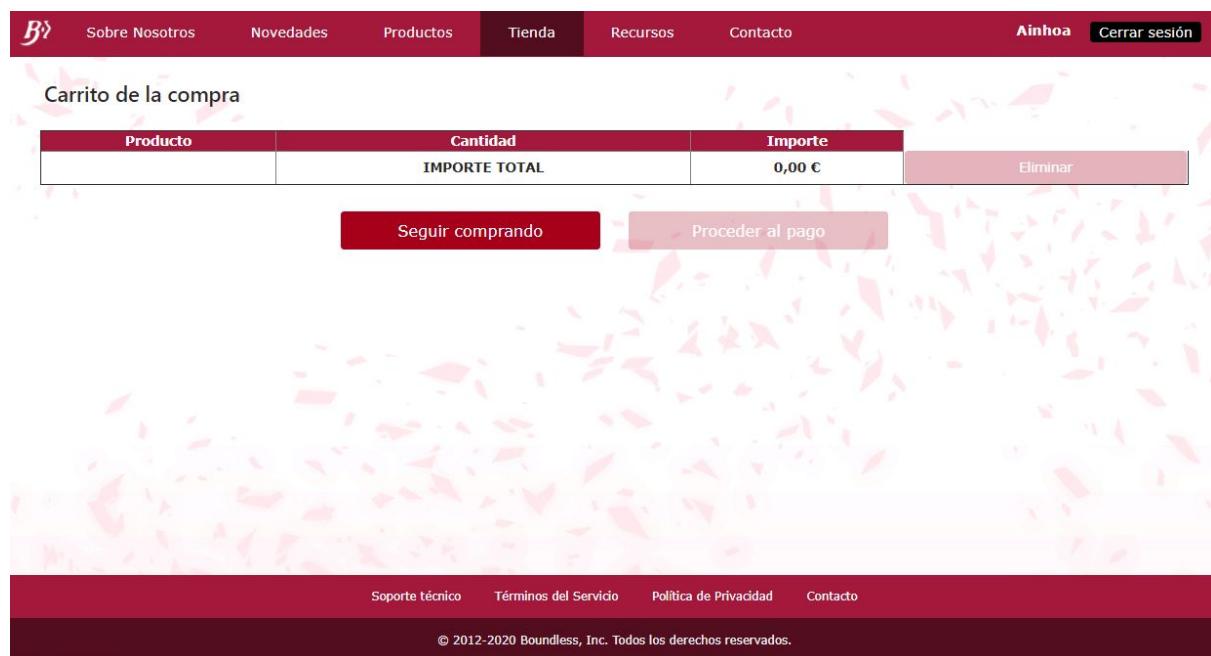


Figura LIX: Aspecto de la página del carrito cuando está vacío

La Figura LIX muestra el aspecto del carrito cuando está vacío. Como se puede observar, los botones de eliminar y proceder al pago están inhabilitados ya que son acciones que el usuario no debe poder hacer. Un usuario no puede realizar un pedido sin haber incluido productos a su cesta ni tampoco puede eliminar productos de una cesta vacía. No obstante, si el usuario añade productos de la tienda a su carrito:

The screenshot shows a shopping cart page with a header containing links for 'Sobre Nosotros', 'Novedades', 'Productos', 'Tienda', 'Recursos', 'Contacto', 'Ainhoa' (logged-in user), and 'Cerrar sesión' (Logout). The main content area is titled 'Carrito de la compra' (Shopping Cart) and displays a table of items. The table has columns for 'Producto' (Product), 'Cantidad' (Quantity), and 'Importe' (Amount). It lists two items: 'KoS: Camiseta blanca STAIN' with quantity 2 and total amount 89,98 €, and 'BD: Taza Silveth' with quantity 1 and total amount 24,99 €. A summary row shows 'IMPORTE TOTAL' (Total Amount) as 114,97 €. On the right side of the table is a red 'Eliminar' (Delete) button. Below the table are two buttons: 'Seguir comprando' (Continue shopping) and 'Proceder al pago' (Proceed to payment).

Producto	Cantidad	Importe
KoS: Camiseta blanca STAIN	2	89,98 €
BD: Taza Silveth	1	24,99 €
	IMPORTE TOTAL	114,97 €
		Eliminar

Figura LX: Aspecto de la página del carrito cuando tiene elementos y el usuario está logueado

Cuando el usuario añade productos al carrito, estos se muestran listados en la página del carrito. Para cada producto se muestra su descripción, cantidad a comprar e importe total. Además, a la derecha se le da la opción de eliminar una instancia de producto. Cuando no quedan instancias de dicho producto, este se elimina automáticamente de la lista. Con el carrito como se muestra en la Figura LX se puede proceder al pago. No obstante, si no estamos logueados:

The screenshot shows a shopping cart page similar to the previous one, but with a different header. The header includes 'Sobre Nosotros', 'Novedades', 'Productos', 'Tienda', 'Recursos', 'Contacto', and 'Acceso' (Login). The main content area is titled 'Carrito de la compra' and displays a table of items. The table has columns for 'Producto', 'Cantidad', and 'Importe'. It lists two items: 'BD: Taza Silveth' with quantity 1 (maximum available) and total amount 24,99 €, and 'KoS: Camiseta blanca STAIN' with quantity 1 and total amount 44,99 €. A summary row shows 'IMPORTE TOTAL' as 69,98 €. On the right side of the table is a red 'Eliminar' button. Below the table are two buttons: 'Seguir comprando' and 'Proceder al pago'. A message 'Debe iniciar sesión para continuar' (You must log in to continue) is displayed above the payment button.

Producto	Cantidad	Importe
BD: Taza Silveth	1 (máximo disponible)	24,99 €
KoS: Camiseta blanca STAIN	1	44,99 €
	IMPORTE TOTAL	69,98 €
		Eliminar

Figura LXI: Aspecto de la página del carrito cuando tiene elementos pero el usuario no está logueado

Un usuario que no ha iniciado sesión puede introducir productos en su carrito pero no podrá proceder al pago. La Figura LXI muestra como la página imprime un mensaje de error y deshabilita la opción de continuar para pagar. Por tanto, si quiere tramitar su pedido deberá iniciar sesión. Además, como se puede observar en el primer producto listado, el número de elementos que se pueden añadir al carrito está limitado por el stock disponible. Todo esto se ha implementado:

```
<tr>
<c:set var="importeTotal" value="0"/>
<c:set var="carro" value="${carrito}" />
<c:forEach var="mercancia" items="${carro}">
    <tr>
        <td text-align: center; >
```

```

        <bd>${mercancia.descripcion}</b>
    </td>
    <td style="text-align: center">
        <c:choose>
            <c:when test="${!empty param.stockMax &&
mercancia.descripcion == param.mercanciaString}">
                <bd>${mercancia.cantidad} (máximo
disponible)</b>
            </c:when>
            <c:otherwise>
                <bd>${mercancia.cantidad}</b>
            </c:otherwise>
        </c:choose>
    </td>
    <td style="text-align: center">
        <bd><fmt:formatNumber value="${mercancia.precio * mercancia.cantidad}" type="currency"/></b>
    </td>
    <td>
        <center>
            <input type="radio" name="descripcionMercancia"
value="${mercancia.descripcion}">
        </center>
    </td>
</tr>
        <c:set var="importeTotal" value="${importeTotal +
mercancia.precio * mercancia.cantidad}">
    </c:forEach>
<tr>
    <td></td>
    <td style="text-align: center"><b>IMPORTE TOTAL</b></td>
        <td style="text-align: center"><b><fmt:formatNumber
value="${importeTotal}" type="currency"/></b></td>
    <td style="text-align: center; border: none;">
        <c:choose>
            <c:when test="${importeTotal == 0}">
                <input class="w3-button w3-round w3-text-white
w3-highway-red w3-hover-black w3-block"
type="submit" value="Eliminar" disabled>
            </c:when>
            <c:otherwise>
                <input class="w3-button w3-round w3-text-white
w3-highway-red w3-hover-black w3-block"
type="button" value="Actualizar" onclick="location.href='
${pageContext.request.contextPath}/actualizarMercancia?mercanciaId=${mercancia.id}'">
            </c:otherwise>
        </c:choose>
    </td>
</tr>

```

```

w3-highway-red w3-hover-black w3-block"
type="submit" value="Eliminar">
    </c:otherwise>
</c:choose>
</td>
</tr>
</tr>

```

Código XXVII: Funcionamiento de la vista del carrito

Además, la introducción en el carrito se gestiona de la siguiente forma:

```

//////////////////DATOS DEL FORMULARIO
// Almacenamos los parametros de entrada en variables temporales
String cantidadString = request.getParameter("cantidad");
Integer cantidad = new Integer(0);

String stockString = request.getParameter("stock");
Integer stock = null;
if(stockString != null){
    stock = Integer.parseInt(stockString);
}

String mercanciaString =
request.getParameter("descripcionMercancia");

//Si no se ha introducido un número
if (cantidadString == null){
    cantidad=-1;
}
else{
    // Convertimos "numero" a entero
    cantidad = Integer.parseInt(cantidadString);

    //Si se ha introducido un número negativo se interpreta como un
0
    if(cantidad<0)
        cantidad=0;
}

// Recuperamos la descripción del producto
if(mercanciaString != null) {

```

```

baseDatos BD = new baseDatos();
Mercancia mer = new Mercancia();

// Creamos una nueva sesión
HttpSession session = request.getSession(true);

// Recuperamos lista de la sesión
// Si es la primera vez, creamos la lista
carro = (ArrayList<Mercancia>) session.getAttribute("carrito");

boolean e = false;

if ( carro == null )
{
    // Creamos la lista
    carro = new ArrayList<>();

    mer = BD.recuperarMercancia(mercanciaString);
    mer.setCantidad(cantidad);

    // COMPLETAR: Introducimos el producto en la lista
    carro.add(mer);
}

// De existir una lista previa
else{
    mer.setCantidad(cantidad);
    for (Mercancia aux : carro ){
        if(aux.getDescripcion().equals(mercanciaString)){
            e = true;
            if(aux.getCantidad()+mer.getCantidad() < 1){
                //Borramos el producto de la lista
                carro.remove(carro.indexOf(aux));
                break;
            }
            else{
                if(stock != null){
                    if(aux.getCantidad()+mer.getCantidad() > stock){
                        session.setAttribute("carrito", carro);
                        // Presentamos los datos
                    }
                }
            }
        }
    }
}

```

```

gotoPage("/Boundless/paginas/vistaCarro.jsp?stockMax=true&mercanciaString=" + mercanciaString, response);
        return;
    }
    else {
        aux.setCantidad(aux.getCantidad() + mer.getCantidad());
    }
}
else{
    if(mer.getCantidad() < 0){
        aux.setCantidad(aux.getCantidad() + mer.getCantidad());
    }
}
}
}
}
}
}
}
if(e==false){
    mer = BD.recuperarMercancia(mercanciaString);
    mer.setCantidad(cantidad);

    //Introducimos el producto en la lista
    carro.add(mer);
}
}
session.setAttribute("carrito", carro);
}
// Presentamos los datos
gotoPage("/Boundless/paginas/vistaCarro.jsp", response);

```

Código XXVIII: Funcionamiento de inserción y eliminado de elementos del carrito

De este modo, el usuario podrá añadir y eliminar libremente los productos de su carrito siempre limitado por la cantidad que hay en stock.

15.5. Pago

La última de las funcionalidades que hubo que implementar fue el pago del pedido. Esto implicaba la introducción de los datos del pedido en la base de datos y la reducción de la cantidad de productos en stock. El aspecto de la página es muy similar al del carrito:

Figura LXII: Aspecto de la página de confirmación de pago

Como se puede observar, se muestra la tabla de productos pero ya no se da opción a eliminar elementos del pedido. Los botones disponibles ahora son *Cancelar* que redirecciona al carrito y *Proceder al pago* que confirma la compra de los productos listados. En esta página se proporciona además la información de contacto obtenida del *web.xml*:

```
<h3 style="font-weight: bold; color: #a3183b">Información de  
contacto:</h3>  
<p><b>Correo:</b> ${initParam['mailContacto']}</p>  
<p><b>Teléfono:</b> ${initParam['telefonoContacto']}</p>
```

Código XXIX: Representación de la información de contacto

Si el usuario confirma el pago entonces será redirigido a una nueva página que le agradece la compra realizada:



Figura LXIII: Aspecto de la página de agradecimiento de compra

Esta página le da la opción de regresar a la página principal (*index.jsp*) o regresar a la tienda para realizar otro pedido (*/Boundless/stock*). Si se accede de nuevo a la tienda:

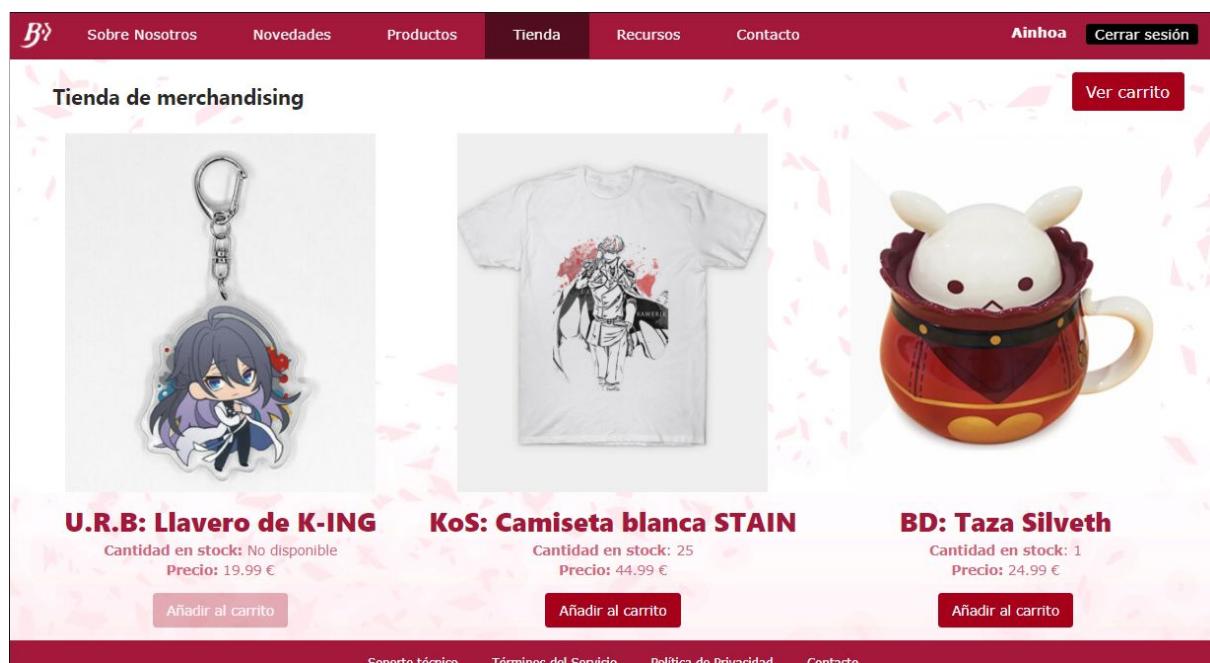


Figura LXIV: Aspecto de la tienda actualizada tras la compra

En la Figura LXIV se puede observar como la disposición de la tienda no es la misma que antes. Esto se debe a que se han realizado cambios en determinados productos de la base de datos, así que el orden de los productos se ve alterado. En la imagen se puede ver

que ha disminuido el stock disponible para la camiseta y la taza, los productos que se han comprado. Esto se ha logrado implementado:

```
// Obtenemos el resumen del pedido
String resumenPedido = request.getParameter("resumenPedido");

// Creamos una nueva sesion
HttpSession session = request.getSession(true);

//Accedemos a la BD
baseDatos BD = new baseDatos();

//Lo desglosamos
for(String s: resumenPedido.split("\\";")){
    String[] res = s.split("\\&");
    String descripcionString = res[0]; // Descripcion

    String cantidadString = res[1]; // Cantidad comprada

    //Convertimos la cantidad
    Integer cantidad = Integer.parseInt(cantidadString);

    //Modificamos la base de datos
    BD.actualizarStock(descripcionString, cantidad);
    BD.realizarPedido((String)session.getAttribute("correoUsuario"),
resumenPedido);

}

//BORRAMOS EL CARRITO
session.setAttribute("carrito", null);

// Presentamos los datos
gotoPage("/Boundless/paginas/vistaCompra.jsp", response);
```

Código XXX: Funcionamiento del pago de un producto

Donde la descripción del pedido se construye como:

```
<c:set var="pedido" value =
"${pedido}${mercancia.descripcion}&${mercancia.cantidad}&${mercancia.precio * mercancia.cantidad};"/>
```

Código XXXI: Estructura de la descripción del pedido

El código XXXI describe como en la descripción de un pedido los productos diferentes se separan por punto y coma (;) mientras que la descripción, cantidad y precio total de un producto se divide con ampersands (&).

De este modo, se han completado todas las funcionalidades requeridas.

15.6. Listado de consultas de la base de datos

Para que el sitio web funcione correctamente se han programado una serie de funciones que realizan consultas SQL a la base de datos. Estas son:

```
stmUsuario = con.prepareStatement("INSERT INTO public.usuario"
+ "apellidol, correo, direccion, nombre, password, sexo, telf,
apellido2, \"fechaNacimiento\", pais)"
+ "VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?);");
//Sustituimos
stmUsuario.setString(1, usuario.getApellido1());
stmUsuario.setString(2, usuario.getCorreo());
stmUsuario.setString(3, usuario.getDireccion());
stmUsuario.setString(4, usuario.getNombre());
stmUsuario.setString(5, usuario.getPassword());
stmUsuario.setString(6, usuario.getSexo());
stmUsuario.setString(7, usuario.getTelf());
stmUsuario.setString(8, usuario.getApellido2());
stmUsuario.setDate(9, usuario.getFechaNacimiento());
stmUsuario.setString(10, usuario.getPais());
```

Consulta I: Registrar usuario

En la Consulta I, se introduce en la base de datos toda la información recibida del formulario de registro.

```
String cst = "select * from usuario where correo = ?";
//Preparamos la sentencia donde se seleccionan todos los campos de
la tabla de que tengan como usuario y contraseña los proporcionados por
argumentos
stmUsuario = con.prepareStatement(cst);
//Sustituimos
stmUsuario.setString(1, correo); //Id del usuario (correo)
```

Consulta II: Validad acceso del usuario

En la Consulta II, se recupera un usuario de la base de datos a partir de su correo electrónico ya que es la clave primaria de la tabla. Esto permite saber si está registrado y

servirá para recuperar la contraseña. Así se podrá comprobar si el usuario que intenta ingresar en el sistema utiliza las credenciales almacenadas en la base de datos.

```
String cst = "select * from usuario where correo = ?";
//Preparamos la sentencia donde se seleccionan todos los campos de
la tabla de usuarios que tengan como usuario y contraseña los
proporcionados por argumentos
stmUsuario = con.prepareStatement(cst);
//Sustituimos
stmUsuario.setString(1, correo); //Id del usuario (correo)
```

Consulta III: Validar registro

En la Consulta III, se intenta recuperar de la base de datos el usuario que tiene el correo especificado. Si no existe dicho usuario, se podrá proceder a realizar el registro.

```
String cst = "select * from mercancia";
//Preparamos la sentencia donde se seleccionan todos los campos
stmMercancia = con.prepareStatement(cst);
//Ejecutamos
rsMercancia = stmMercancia.executeQuery();
```

Consulta IV: Recuperar mercancía

En la Consulta IV, se recupera de la base de datos toda la información de los productos que están a la venta. Esta servirá para construir la tienda.

```
String cst = "select * from mercancia where nombre = ?";
//Preparamos la sentencia donde se seleccionan todos los campos
stmMercancia = con.prepareStatement(cst);
stmMercancia.setString(1, descripcion);
```

Consulta V: Recuperar un producto

En la Consulta V, se recupera de la base de datos la información de un producto. De este modo se podrá utilizar dicha información para generar el carrito.

```
String cst = "INSERT INTO public.pedido("+
    "usuario, fecha, contenido) " +
    "VALUES (?, ?, ?);";
//Preparamos la sentencia donde se seleccionan todos los campos
stmMercancia = con.prepareStatement(cst);
stmMercancia.setString(1, correo);
stmMercancia.setDate(2, new
Date(Calendar.getInstance().getTimeInMillis()));
```

```
stmMercancia.setString(3, pedido);
```

Consulta VI: Realizar pedido

En la Consulta VI, se introduce un nuevo pedido en la base de datos. Este no necesita introducir el ID a pesar de ser su clave primaria ya que se ha definido una secuencia que lo genera automáticamente:

```
CREATE SEQUENCE public.pedido_id
CYCLE
INCREMENT 1
START 1
MINVALUE 0
MAXVALUE 100
CACHE 1;
```

Código XXXII: Definición de la secuencia pedido_id

```
id numeric NOT NULL DEFAULT nextval('pedido_id'),
```

Código XXXIII: Descripción del ID del pedido

De este modo, el sitio web no tendrá que generar un ID. Esto evita que se generen IDs repetidos o no válidos.

```
String cst = "UPDATE public.mercancia " +
    "SET cantidad = cantidad-? " +
    "WHERE nombre = ?";
//Preparamos la sentencia donde se seleccionan todos los campos
stmMercancia = con.prepareStatement(cst);
stmMercancia.setInt(1, nuevoStock);
stmMercancia.setString(2, producto);
```

Consulta VII: Actualizar el stock de un producto

En la Consulta VII, se actualiza la tabla de mercancía para que disminuya la cantidad disponible en stock en el número de unidades adquiridas al realizar el pedido.

15.6. Información de contacto

Finalmente, también se actualizó la página de contacto para que recupere los datos de contacto de Boundless del archivo web.xml. Estos son teléfono y correo electrónico:



Información de contacto

Para consultas sobre un juego, servicio o evento patrocinado por Boundless, por favor envíe su consulta a través del formulario de contacto designado para ese juego, servicio o evento. Por favor, tenga en cuenta que sólo recibimos consultas en español.

■ Blue Destiny

Sitio web oficial: <https://blue-destiny.com/>
También puede contactarnos a través del [formulario oficial](#).

■ U.R.B: Ultimate Rap Battle

Sitio web oficial: <https://urb.com/>
También puede contactarnos a través del [formulario oficial](#).

■ King of Seven

Sitio web oficial: <https://kos.com/>
Correo electrónico: monarch@kos.com

■ Boundless, Inc.

Sitio web oficial: <https://boundless.com/>
Correo electrónico: contact@boundless.com
Teléfono: +34 648 84 68 19

Soporte técnico Términos del Servicio Política de Privacidad Contacto

© 2012-2020 Boundless, Inc. Todos los derechos reservados.

Figura LXV: Aspecto de la página de contacto

Para que el contenido del fichero se represente en la página como se muestra en la Figura LXV, se deberán recuperar de la siguiente forma:

```
<p>Correo electrónico: ${initParam['mailContacto']}</p>
<p>Teléfono: ${initParam['telefonoContacto']}</p>
```

Código XXXIV: Estructura de la descripción del pedido

16. Control de cambios

A continuación se muestran detalladamente las diversas modificaciones que se han realizado sobre los elementos que componen el sitio web de Boundless con respecto al diseño original propuesto el 27 de Setiembre de 2020.

16.1. HTML

Cambios realizados a 04/10/2020

- Implementación de todas la páginas adaptándolas a las restricciones de html.
 - Secuencialización del contenido.
 - Limitación en los colores.
 - Uso de separadores de contenido (`<hr>`) para diferenciar secciones.
 - Uso de espaciado (` `) y saltos de línea (`
`) para posicionar elementos.
 - Sustitución de las imágenes de fondo en las secciones por texto.
 - Redimensión del tamaño de las imágenes.
 - Sustitución del slideshow por una imagen estática en la página principal.

- Creación de la página *Próximamente* usada en las redirecciones de artículos no implementados y de *Servicio Técnico*.
- Rediseño del logo de la empresa.

15.2. CSS

Cambios realizados a 18/10/2020

- **Generales**
 - Implementación de un menú de navegación móvil.
 - Supresión de las barras blancas verticales en el menú de navegación.
 - Supresión de los triángulos en el menú de navegación.
 - Cambios menores en la paleta de color del sitio web: tonalidad, saturación, luminosidad y harmonía cromática.
 - Aumento del número de estrellas utilizadas en las guirnaldas, lo que resulta en una disminución de su altura.
 - Sustitución de un pie de página móvil por uno estático.
 - Eliminación de las barras blancas verticales utilizadas como separadores en el pie de página.
 - Eliminación de la opción de búsqueda al no considerarse una herramienta útil en la página.
 - Implementación de un hover en la barra de navegación y oscurecimiento del color de fondo utilizado para marcar la página seleccionada.
 - Implementación de transiciones que simulan la carga de los elementos de la página para aportar dinamismo al sitio web.
 - Diseño e inserción de un favicon.
- **Página de inicio**
 - Modificaciones en el logo de la empresa.
 - Modificaciones menores en la densidad de elementos en el fondo de la sección del logo.
 - Modificaciones en el Slideshow.
 - Eliminación del borde rojo carmesí que lo rodeaba.
 - Rediseño de las flechas de navegación.
 - Eliminación de los otros *dots* de navegación debido a su inutilidad.
 - Sustitución del ícono de estrella por el de diamante en la sección de *Novedades Destacadas*.
- **Sobre Nosotros**
 - Eliminación del menú de navegación intrínseco de la página al poder considerarse confuso para el usuario.
 - Eliminación del apartado *Afiliados* debido a que provocaba que la última sección de la página fuese marrón y no permitía ver la imagen de fondo. Se suprimió *Afiliados* al considerarse la sección de menor importancia de la página.

- **Novedades**
 - Sustitución de los iconos de la sección del Archivo debido a las limitaciones de Unicode.
 - Modificaciones menores en la imagen de fondo de la página.
- **Artículos**
 - Modificación en el formato del PATH. La ruta de los artículos está ahora compuesta por: *JUEGO/TIPO/AÑO/MES*
 - Modificación en la opacidad de la sección *Artículos Relacionados* para ajustarlo al diseño original.
 - Eliminación de la posibilidad de compartir el artículo utilizando el ícono situado en la esquina superior derecha de la noticia.
- **Productos**
 - Adición de una guirnalda gris al final de la página.

16.3. JavaScript

Cambios realizados a 24/10/2020

- **General**
 - Se han añadido imágenes a la carpeta /imagenes.
- **Página de inicio**
 - Inclusión de dos imágenes adicionales en el slideshow.
 - Eliminación de dos de los círculos de navegación con respecto al diseño original debido a su desuso.
 - **Sobre nosotros**
 - Inserción de un iframe en la sección Localización que muestre la ubicación del edificio Lumina en un mapa interactivo.
- **Productos**
 - Implementación del highlight utilizando JavaScript y métodos de acceso al DOM en los productos.
- **Contacto**
 - Programación de código en JavaScript con métodos de acceso al DOM que permitan la generación de una ventana confirmadora al hacer click en un enlace. Esta ventana avisa al usuario de que será redirigido y le permite decidir si quiere o abrir la nueva pestaña o cancelar.

16.4. AJAX

Cambios realizados a 29/10/2020

- **Página de inicio**
 - Cambio del color de los círculos de navegación del slideshow de la página de inicio. El color del hover se cambió a rosa.
 - Sustitución de rutas estáticas a las imágenes de los productos de la sección *Nuestros Juegos* por rutas obtenidas de *productos.xml*

- Redimensión del fondo del logo usando en la cabecera.
- **Novedades**
 - Sustitución del texto incrustado en el body por texto construido a partir de la información sobre los artículos recuperada de *novedades.json*
- **Productos**
 - Sustitución del texto incrustado en el body por texto construido a partir de la información sobre los artículos recuperada de *productos.xml*
 - Sustitución de las rutas estáticas a las imágenes promocionales de los juegos por rutas obtenidas de *productos.xml*
- **Páginas específicas de los juegos (BD, URB y KoS)**
 - Sustitución del texto incrustado en el body por texto construido a partir de la información sobre los artículos recuperada de *productos.xml*
 - Sustitución de las rutas estáticas a las imágenes y el banner del juego promocionado por rutas obtenidas de *productos.xml*
- **Estado de los servidores**
 - Sustitución del texto incrustado en el body por texto construido a partir de la información sobre los artículos recuperada de *productos.xml*
 - Sustitución de las rutas estáticas a las imágenes y el banner del juego promocionado por rutas obtenidas de *productos.xml*
- **Contacto**
 - Sustitución del texto incrustado en el body por texto construido a partir de la información sobre los artículos recuperada de *productos.xml*

16.5. JQuery

Cambios realizados a 02/11/2020

- **General**
 - Se ha modificado el color del hover en el menú de navegación. Este es ahora un poco más claro que el color utilizado para marcar la página en la que se encuentra el usuario.
 - Se ha implementado un fino y sutil borde inferior en el menú de navegación para delimitarlo.
- **Página de inicio**
 - Implementación del highlight en los juegos de la sección *Nuestros Juegos*.
- **Novedades**
 - Programación de código JQuery que permite que al hacer click en el año 2020 de la sección *Archivo* se muestren u oculten los meses en los cuales ha habido publicaciones.

16.6. JSP, Java Beans, EL, JSTL y JDBC

Cambios realizados a 21/12/2020

- **General:**

- Se ha incorporado un botón de *Acceso* al menú de navegación. Si el usuario ya está logueado en el sitio web este se sustituye por un botón de *Cerrar Sesión* precedido por el nombre del usuario.
 - Se ha incorporado un botón de *Tienda* al menú de navegación.
 - Se han sustituido todas las páginas .html por .json equivalentes.
 - Se han creado las páginas *vistaAcceso.json*, *vistaCarro.json*, *vistaCompra.json*, *vistaPagar.json* y *vistaTienda.json*.
 - Se han creado las clases *baseDatos.java*, *carro.java*, *cerrar.java*, *login.java*, *Mercancia.java*, *pagar.java*, *registro.java*, *stock.java* y *Usuario.java*.
 - Se ha creado el fichero *web.xml*.
 - Se ha implementado una base de datos *Boundless*.
 - Se han generado scripts para la creación de la base de datos y las tablas, para la inserción de elementos de prueba y para la eliminación de las tablas.
 - Se han agregado nuevos estilos a *estilo.css*.
 - Se han agregado nuevas imágenes a la carpeta *imagenes*.
 - Se ha modificado la estructura de ficheros para que la aplicación cumpla el patrón MVC.
 - Se ha actualizado el diagrama que muestra la estructura de ficheros.
- **Contacto:**
 - Se ha sustituido la información de contacto estática por información obtenida del fichero *web.xml*

17. Referencias

A continuación se recogen todas las páginas web utilizadas para la obtención de información adicional e imágenes para el trabajo, así como aquellas empleadas para elaborar el diseño de interfaces.

17.1. Bibliografía

- [1] **Práctica 1.** Pablo García Tahoces. *Desarrollo de Aplicaciones Web*, Campus Virtual:
https://cv.usc.es/pluginfile.php/895411/course/section/111086/Practica_1_c.pdf
- [2] **Desarrollo de Aplicaciones Web. Diseño Web.** Pablo García Tahoces. *Desarrollo de Aplicaciones Web*, Campus Virtual:
<https://cv.usc.es/pluginfile.php/895411/course/section/111086/DisWeb.pdf?time=1567708787950>
- [3] **Qué es un inventario de contenidos y cómo hacerlo con éxito.** Emanuel Olivier Peralta. Semrush: <https://es.semrush.com/blog/inventario-contenidos-que-es/>
- [4] **¿Sabes lo que es un inventario de contenidos?** Seigoo. El blog de Seigoo:
<https://blog.seigoo.com/inventario-de-contenidos>
- [5] **Electronic Arts.** <https://www.ea.com/es-es>
- [6] **Cygames.** <https://www.cygames.co.jp/en/>
- [7] **Nintendo.** <https://www.nintendo.es>

- [8] **Sony Computer Entertainment.** <https://www.sie.com/en/index.html>
- [9] **Ubisoft.** <https://www.ubisoft.com/es-es/>
- [10] **Página de Inicio.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/P%C3%A1gina_de_inicio
- [11] **Política de privacidad.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/Pol%C3%ADtica_de_privacidad
- [12] **Arquitectura de la información.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/Arquitectura_de_la_informaci%C3%B3n
- [13] **Card Sorting: Técnica de categorización de contenidos.** *Yusef Hassan Montero, Francisco J. Martín Fernández.* No solo usabilidad (nsu): <http://www.nosolousabilidad.com/articulos/cardsorting.htm>
- [14] **¿Qué es el sketching y qué aporta en el diseño de un producto?** *Trabajadores de ESDESIGN.* ESDESIGN (Escuela Superior de Diseño de Barcelona): <https://www.esdesignbarcelona.com/es/expertos-diseno/que-es-el-sketching-y-que-aporta-en-el-diseno-de-un-producto>
- [15] **Website wireframe.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/Website_wireframe
- [16] **Wireframes: Qué son y cómo crearlos.** *Colaboradores de Web desde Cero.* Web desde Cero: <https://webdesdecero.com/wireframes-que-son-y-como-crearlos/>
- [17] **Mockup.** *Colaboradores de Wikipedia.* Wikipedia, The Free Encyclopedia: <https://en.wikipedia.org/wiki/Mockup>
- [18] **Mapa de sitio web.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/Mapa_de_sitio_web
- [19] **Sistema de archivos.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: https://es.wikipedia.org/wiki/Sistema_de_archivos
- [20] **AJAX.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: <https://es.wikipedia.org/wiki/AJAX>
- [21] **JQuery.** *Colaboradores de Wikipedia.* Wikipedia, la Enciclopedia Libre: <https://es.wikipedia.org/wiki/JQuery>

17.2. Recursos complementarios

- El logo de la empresa (Figura XV) fue diseñado utilizando la fuente [Tornado](#) de [177Studio](#). Esta está disponible de forma gratuita para uso personal. Se ha usado en este proyecto debido a que el susodicho no tiene intención comercial.
- Se ha utilizado la web [awwwards](#), como fuente de inspiración para el diseño del sitio web.
- Los esquemas y diagramas (figuras I, II, XXII, XXIII, XXV, XLII y XLVII) fueron realizados en [Creately](#).
- Los sketches (figuras III a la VII) fueron realizados a mano
- El wireframe (figuras VIII a la XVI) fue realizado en [Mockflow](#)

- El *mockup* (figuras XVII a la XXI), el *storyboard* (figura XXIV), el mapa de etiquetas (figuras XXVI a la XXX) y el mapa de clases (figuras XXXI a la XXXV) fueron realizados en [MediBang Paint](#)
- El edificio usado en la página *Sobre Nosotros* es el [Edificio Lumina](#).
- Las ilustraciones de la web fueron realizadas por Hen-tie y son propiedad intelectual de su autor. Se han utilizado de forma gratuita en este trabajo puesto que no tiene carácter lucrativo. Sus redes sociales son:
 - [DeviantArt](#)
 - [Instagram](#)
 - [Portfolio](#)
 - [Twitter](#)
- Para la elaboración del CSS de Boundless se ha utilizado el Wireframe [W3.CSS](#).
- Se ha utilizado el [VS Code](#) como editor de código fuente.
- Se ha utilizado [postgresql](#) y [pgAdmin 4](#) para la gestión de la base de datos.
- Se ha utilizado [Apache Tomcat](#) como servidor para el sitio web.