Computación Orientada a Web

Entrega Tema1

UNIVERSITAT DE BARCELONA

Sebastian Andrade Zanotti - NIUB17692754

Barcelona, España, 14/03/2022

Apartado 1

Caracteristicas:

El documento HTML consiste de un Header, que contiene el logo y las opciones de registro de usuario; un Footer, que contiene la información de contacto; una columna izquierda, que tiene información extra sobre la página, y está pensada para funcionar como menú de navegación en el futuro, cambiando simplemente la lista por links; y por último un cuadro central con la opción de búsqueda.

Funcionalidades:

Las funcionalidades son las mismas que pone en el anuncio. En la columna izquierda tenemos un enlace que nos redirige a Booking si hacemos click en él. También tenemos un enlace en el Footer, que al clickar en el gmail de contacto abre el mail en nuestra pc para que el usuario pueda enviarnos un correo. En el cuadro central tenemos el formulario que al presionar el botón de “search” realiza una búsqueda por google de lo que sea que desee buscar el usuario.

Los botones de registro de usuario no están implementados pues en el futuro llevarán a una página con un formulario más extenso.

Comparación con el código ejemplo:

Comparando con el código html que está en las transparencias, mantenemos un layout prácticamente idéntico, utilizando la clase de main body que contiene el Header, un container con Left y Center (la columna de la izquierda y el cuadro central de la página), y un footer. No utilizamos la columna Right. Dejando por fuera los cambios al agregar los diferentes elementos html ya mencionados.

La mayoría de los cambios están en el archivo CSS, sobre todo en los parámetros de width y height, que ahora están en auto, a excepción del ancho en la columna Left, y la altura del footer, que están predeterminadas con píxeles, para que sea responsive dentro de lo posible.

Considero que lo más interesante en concepto está en Center, donde usamos width 100% para que use todo el ancho disponible. De fondo utilizamos una imagen con las propiedades: background-image: url("images/background.webp"); que nos da la imagen a usar como tal. background-size: cover; para que utilice todo el espacio disponible, y no se repita. background-position: center; para que utilice siempre el centro de la imagen, y se ajuste respectivamente.

Podrá ver que existen clases para casi cada elemento html. En mayoría se usan para jugar un poco con los márgenes para dar la posición deseada. En el caso de el Form que tenemos en Center, usamos especialmente text-shadow para que se distinga mejor el texto del form.

Apartado 2

Tiene las mismas características y funcionalidades que el apartado 1, ya que sigue la misma estructura que el html anterior, pero con una leve diferencia en la posición de algunos elementos. El cambio más drástico es el menú de la izquierda que ya no es la lista que teníamos antes sino un sidebar que muestra las posibles opciones de navegación, por los mementos, que interactúa según pasa el cursor por encima, y muestra la pestaña activa, en este caso Home.

Comparación con el código ejemplo:

La estructura es similar, si tomamos en cuenta que está dividida en Header, Main, y Footer. En nuestro HTML tenemos un container-fluid, en lugar de clase main, pero tiene el mismo propósito. Este contenedor utiliza la misma estructura de Fila y Columna. Tenemos un row, compuesto por 2 columnas, una de 2 para el sidebar y otra de 10 para el cuadro central. La clase sidebar utiliza un Style que está definido arriba del todo, como en la Demo que se nos proporciona, que contiene las propiedades para cada elemento que usamos dentro del sidebar, más que nada los del menú y sus links.

No se utiliza el álbum dado en la Demo, esto es algo que se implementara en el futuro para las pestañas de Top Hotels, etc.. para mostrar un álbum con las mejores opciones. Pero de momento prefiero las landing page sencillas y que no sobrecarguen al usuario.

Uno de los problemas que conseguí adaptando la web, es que el centro y el sidebar no se juntaban por completo al footer, y si lo hacían quedaban cortos de altura en la página. Una solución que conseguí fue usar el parámetro vh-100 en row para que utilice el 100% del espacio disponible, pero me da la impresión de que utiliza de más.

La mayoría de propiedades que definimos en cada clase son prácticamente iguales, solo se cambian propiedades relativas a la estética, o se eliminaron propiedades que daban la impresión de no hacer nada sobre nuestra web.

Un truco se se utilizó en el header por ejemplo es usar la clase nav-bar como en la demo, colocar los elementos html deseados, para que por defecto vayan a la izquierda, y luego usar un div, también de la clase nav-bar, pero con la propiedad de text-right para que caigan al otro extremo del header.

Si bien es cierto que bootstrap hace las cosas más bonitas, me parece que en el apartado 1 con css era más intuitivo, aunque el modelo por rejilla de bootstrap fue mucho más sencillo de utilizar para establecer el layout de la página. Pero por otro lado, me gusta tener el html limpio con todos los detalles en el css, en lugar de invocar el archivo css desde el html como hacemos utilizando bootstrap.