Informe de Implementación sobre Landing Page

Este informe se estructuró siguiendo los pasos para el diseño y desarrollo de la Landing Page, asignada a llevar de una plantilla .psd a código web.

El primer paso consistió en descargar la versión 5.0.3 de Wordpress, ya que este fue el CMS indicado para la creación de la página web, la creación de una base de datos MySQL bajo el nombre de “wordpress”, esto debido a que es el requisito necesario, además de PHP >= 5.6, para la instalación de dicho CMS. Posteriormente, se instaló Wordpress localmente a través de su guía de instalación estableciendo la antes mencionada base de datos como la de uso del sistema, posteriormente la creación del usuario administrador y el ingreso a la página de administración.

Luego, se seleccionó el framework CSS, Bootstrap 4, como la herramienta a usar para el desarrollo bajo la filosofía mobile first de la Landing Page, gracias a que uno de los requisitos era la implementación de una estructura responsive para la misma, se procedió a descargar dicho framework para incluirlo localmente en la nueva estructura de carpetas enfocada para el nuevo tema. A pesar, de que Wordpress trae por defecto un enlace a un archivo JS de JQuery, se prefirió descargar la versión 3.3.1 minificada del mismo e incluirlo junto al Bootstrap 4 como una de sus dependencias. También, se descargó la versión minificada de Popper ya que es una de las dependencias que exige Bootstrap en su versión 4.

Se creó un repositorio git público en la cuenta angelToxicity con el nombre “wordpress_simple_page” y en la carpeta de trabajo del proyecto se inicializó git para poder manejar un control de las versiones y publicar en dicho repositorio el progreso de la página web. Luego se descargó la plantilla o diseño Simple_Landing.psd.zip proporcionado a través de correo electrónico, se descomprimió y en vista de que el sistema operativo que posee el equipo donde se trabajó es software libre, en este caso Ubuntu 16.04 LTS, la posibilidad de utilizar una herramienta de diseño o edición tales como Adobe Photoshop o Adobe Illustrator, se consideró nula debido al peso de dichos programas, la incompatibilidad con software libre y problemas con la conexión a Internet, es por esto que se optó por trabajar con la herramienta nativa del sistema operativo para la edición de imágenes y con esto nos referimos a GIMP 2.8, el cual a pesar de no poseer las bondades de los anteriores programas, nos permite trabajar de forma similar con archivos .psd a través de la edición de capas, aunque no es de facilidad, tampoco supone un reto su uso.

Ya teniendo las principales herramientas para el diseño de la página, creamos una carpeta correspondiente a un nuevo tema en la dirección /var/www/html/wordpress/wp-content/simplelanding/, a la cual le otorgamos los permisos de lectura y escritura para que no existiera conflicto de permisos con Worpress, internamente se crearon los siguientes archivos:

- index.php
- header.php
- footer.php
- functions.php
- style.css

También se crearon las siguientes carpetas:

- js → para almacenar los archivos .js de JQuery y Popper.
- font → con el fin de almacenar la fuente para estilizar nuestros títulos.
- images → esta carpeta almacena las imágenes utilizadas para nuestro sitio.
- boostrap → incluye el código del framework CSS.

Con toda la estructura de carpetas ya ordenada, procedimos a importar mediante las funciones internas de Wordpress los archivos .css y .js necesarios para el desarrollo del sitio, esto mediante la edición del archivo functions.php, donde colocamos el siguiente código:

```
function bootstrap_scripts() {
	wp_enqueue_style( 'mytheme-css', get_template_directory_uri() . '/style.css' );
	wp_enqueue_style('mytheme-bootstrap',
get_template_directory_uri() .'/bootstrap/dist/css/bootstrap.min.css' );
	wp_enqueue_script('mytheme-bootstrapjs',
get_template_directory_uri().'/bootstrap/dist/js/bootstrap.min.js',	array('jquery-3.3.1',
'popperjs') );
}
add_action( 'wp_enqueue_scripts', 'bootstrap_scripts' );

function js_scripts() {
	wp_enqueue_script('jquery-3.3.1',get_template_directory_uri().'/js/jquery-
3.3.1.min.js', array() );
	wp_enqueue_script('popperjs',get_template_directory_uri().'/js/popper.min.js',
array('jquery-3.3.1') );
	wp_enqueue_script('sweetalert',  get_template_directory_uri().'/js/sweetalert.min.js',
array('jquery-3.3.1') );
}
add_action( 'wp_enqueue_scripts', 'js_scripts' );
```

Estas funciones propias de wordpress nos permiten ahorrarnos el código HTML para importar a nuestro index, los css y js, al cargar en fila los archivos descritos en ellas. Además, es necesario en nuestro archivo principal abrir una etiqueta php para indicar que debe apuntar al functions.php con el fin de que cargue el código previo, esto mediante la función wp_head().

El siguiente paso ya consistió en el desarrollo mobile first de nuestra página. Principalmente del header comprendido por un navbar hecho fácilmente gracias a Bootstrap, el cual se diseñó para que se adaptara a nivel de dispositivos móbiles con un botón toggler que desplegara una lista de cinco link pertenecientes a la página. Para la versión de tablets, permanece el diseño anterior aunque con una mayor separación entre el título y el botón toggler. Prosiguiendo en orden descendente para el header, se decidió incluir la imagen de fondo junto a los títulos principales en el header, para lograr los diversos estilos que poseen tanto Íla imagen como el texto, se editó el archivo style.css añadiendo clases específicas para obtener el resultado deseado. De forma análoga se utilizó la función @font-face de css para importar las fuentes de texto a partir de nuestra carpeta font, sin necesidad de consumir el contenido de un API que genere una conexión obligatoria a Internet de parte de nuestro sitio. Todo lo descrito fue redactado en el archivo header.php, el cual debe ser llamado en nuestro index.php mediante una etiqueta php y con la función get_header().

Continuamos, en este caso ya en el archivo index.php donde creamos dos secciones, la primera corresponde a las tres imágenes y sus respectivos textos los cuales fueron adaptados para las vistas móviles y de escritorio gracias nuevamente a clases personalizadas en el archivo style.css y la forma circular de dos de ellas fueron obtenidas con el uso de clases nativas de bootstrap, es importante acotar que fueron definidos como fijos los valores de altura y anchura para las tres imágenes. Esta sección es interesante a nivel de diseño ya que para las tres versiones del responsive, la disposición de elementos cambia:

- Para dispositivos móviles, la disposición de las imágenes como del texto es uno sobre el otro, obteniendo una sección dividida en tres con un espaciado para separa cada imagen con su texto del siguiente bloque.
- La vista de dispositivos tablets cambia la configuración visual de los elementos ya que tenemos en una fila dos bloques de elementos (imagen y texto) que se distribuyen hacia el inicio y el final de la fila de tal forma que existe una simetría en el espacio que comparten. La tercera imagen se posiciona en la fila posterior con un margen superior y se encuentra centrada y su texto sigue estando debajo de la imagen.
- Por último, la vista de dispositivos desktop, se asemeja a la mostrada en la plantilla .psd, donde las tres imágenes cohabitan en la misma fila con una separación lateral pero manteniendo un margen de los laterales de la pantalla.

La siguiente sección donde la identificamos por un color de fondo azul claro y el inicio de la misma con una barra de transición multicolor, hechos ambos con clases definidas nuevamente en el style.css específicamente el fondo con el estilo background y la barra con los estilos border-top y border-image con el valor linear-gradient que nos permite especificar el color y el porcentaje de espacio de la barra que ocupará. El siguiente bloque se conforma por texto el cual ocupa alrededor de la mitad de la fila destinada a su posición, aunque este espacio se modifica a medida que la resolución de la pantalla del dispositivo disminuye, ya que no podría ocupar el mismo espacio o simplemente se distorsionaría la ubicación de los elementos del bloque. El bloque que le sigue es muy parecido al anterior ya que el texto se comporta visualmente igual, aunque para esta sección tenemos un botón que adoptará una posición para las distintas resoluciones: el caso de dispositivos móviles y tablets, dicho botón se posicionará debajo del texto y centrado; mientras que para la vista desktop obtenemos que el botón se posicionará hacia el lado derecho del texto con una separació y centrado horizontal y verticalmente respecto al texto. Para concluir con esta sección nos encontramos con el bloque de una imagen y su descripción, dicho apartado maneja un comportamiento distinto para las tres visualizaciones, para lograrlo se utilizó los media queries de Bootstrap que nos permite definir una clase CSS y luego modificar sus estilos dependiendo del tipo de resolución para el cual nos encontremos; con estas funciones podremos manejar menos código HTML y si las clases nativas de bootstrap no nos permiten definir el posicionamiento visual de un elemento, podremos recurrir a los media queries que nos facilitan el código CSS.

Para finalizar, tenemos en el footer un navbar, localizado en el archivo footer.php y que instanciamos en el index.php de forma similar que al header, con una etiqueta php y la función get_footer(). Para el navbar localizado en el footer, debemos recalcar que posee la misma estructura y comportamiento que el navbar del header. Queremos decir con esto que el titulo del navbar se mantiene para las tres vistas, pero la lista de links solo se muestra desplegada para dispositivos desktop, mientras que las versiones móviles solo visualizan el título y un botón toggler que sí despliega la lista de links.

Con esto dado por finalizado el informe de implementación de la Landing Page en wordpress.