# Memoria Técnica

## Introducción al proyecto:

Ha la hora de gestionar el proyecto se decidió utilizar *Laravel que es un Framework de PHP*, ya que es una herramienta que al haber utilizado durante el curso nos ha facilitado primeramente la elección de este, y segundo la capacidad de tener por la mano algo que ya hemos usado. La utilización de vistas de Laravel es intuitivo y te hace ser ordenado de manera inconsciente.
En cuanto a lenguajes usamos *Javascript* con su *libreria Jquery* este ha sido elegido ya que es un lenguaje muy ágil y para nuestro proyecto era necesario que nuestros lenguajes fueran lo mas rápido posible, sin que tuviéramos un *delay*. Además añadimos *Ajax* que nos permite variar contenido sin necesidad de recargar la pagina y hace mas dinámico nuestro diseño.

## Primeros pasos:

A la hora de empezar el proyecto estuvimos debatiendo el tipo de programa/ aplicación que queríamos hacer, esto hizo que la idea inicial fuera modificada por el tiempo de proyecto que teníamos. El proyecto aunque modificado no cambió de rumbo y la idea que se tenía en sí, siguió hacia adelante como un secuenciador de sonidos pero sin algunos aspectos que podrían convertirse en opciones futuras.

La base de datos en un inicio solo consistía en una tabla de audios con los parámetros necesarios, pero esto se modifico al crear un login y permitir la creación de usuarios.

La aplicación sobretodo consta de parte cliente.

Comenzamos generando la estructura de nuestra aplicación con *HTML* y estructurando este con *CSS*. La creación de la botonera fue lo primero a implementar con los botones( Play,Pause,Mp3…).Después de la botonera, la estructura principal se basaba en una sucesión de div's estructurados a través de columnas que nos proporcionaba *Bootstrap.*

A continuación al tener toda la estructura ya hecha, pasamos a la funcionalidad. En ese momento lo primero era hacer que todos los elementos cuadrados tengan un evento Click que cambiaría el color de estos, este evento luego se asignaría a una clase que seria utilizada para llamar a los sonidos. Los sonidos por defecto, eran guardados en una carpeta accesible al usuario que podía ser fácilmente linkada a través de

*Laravel* con una de sus propiedades "*Storage",* pulsando otra vez en el mismo recuadro se vuelve al estado por defecto.
Después de esto se implementó el primer botón( Clear ), este se utilizaría para dejar por defecto las casillas que tenían el evento anteriormente descrito.

A continuación después de un charla con el tutor del proyecto coincidimos en que un *Login* era necesario así que con las facilidades permitidas por Laravel estuvimos de acuerdo en la creación de este, lo que afecto a nuestra *BBDD* inicial siendo modificada para poder crear la gestión de los usuarios que hasta este momento no tiene el servicio SMTP creado.

En este punto del proyecto en una de las reuniones con nuestro tutor hablamos sobre la funcionalidad de nuestro botón mp3, al comentarlo fue automáticamente descartado, ya que literalmente sería como otro proyecto a parte, aunque esto no esta descartado del todo, lo tomamos como una funcionalidad para el futuro.

Para la correcta aplicación de los botones de guardado y descarga tuvimos que hacer una Refactorización del código para que funcionaran correctamente.

La manera que teníamos hasta este momento de recoger las casillas no era la adecuada así que tuvimos que cambiar la manera de recoger los datos. Creamos un objeto llamado *Tracks* en el cual se colocan todos los datos de cada una de las lineas horizontales o *Track*. Gracias a esto los botones de descarga y de subida fueron implementados.
Al implementar el objeto,los botones de descarga y subida recogían los datos de este y se pudo descargar un archivo en formato JSON y después subirlo y iluminara las casillas seleccionadas.

Como ya teníamos audios sonando creímos adecuado hacer un control de volumen.

Comenzamos a utilizar la función *SetInterval* que nos proporciona JavaScript para crear un bucle infinito, el cual podemos elegir parar. Además se creo un botón Stop, para detener el loop infinito generado por el *SetInterval*. Una vez hecho esto comprobamos que la funcionalidad no era la adecuada a lo que pretendíamos, debido a que hacia un barrido de todos los audios y los hacia sonar sin tener en cuenta ninguna posición.
La funcionalidad del intervalo que habíamos generado, no era la esperada. La función se Reestructuró hasta conseguir pasar los parámetros correctamente.

Utilizamos dos bucles *FOR* generando así dos *Array's* para poder pasar los datos de manera vertical con un tiempo determinado. Para poder generar un bucle infinito generamos un contador que al llegar a un numero determinado por nosotros hace que se reinicie.

Generamos una vista extra para que el usuario pueda subir sus audios personalizados con un maximo de 1 Mb de capacidad. La funcionalidad de esta vista esta basada en Ajax, para un tratamiento de datos inmediato de forma visual, esta vista contiene los botones de agregar audio, modificar el nombre del audio y eliminarlo.

Para finalizar de forma que el usuario no pierda su mezcla a la hora de recarga la página o cerrarla se implementó la funcionalidad "*LocalStorage*" que nos facilita *JavaScript* para que no se pierda la información. Esta función tiene que ir actualizándose cada vez que se haga un cambio en los audios del usuario. Si el usuario cerrase su sesión, perdería todos el trabajo hecho hasta ese momento( los audios subidos se guardan ).

A la hora de subirlo a producción decidimos utilizar *Heroku* el cual estaba limitado y no permitía al usuario subir sus audios, así que fue descartado y tomamos la alternativa de *Amazon Web Services*. Para el dominio comenzamos utilizando Freenom pero como nos parecía mas adecuado tener una extensión *.com* contratamos un servicio de dominio en la web *Hostalia*.