­­­­

PROYECTO DE FIN DE CURSO

Javier Rodríguez Montero



* 1. DESCRIPCIÓN DEL PROYECTO

Hoy en día, siempre llevamos encima un aparato electrónico ya sea un smartphone, una tablet o un ordenador portátil y llevar mucha documentación en nuestros equipos puede ser muy tedioso a la hora de organizarla y poder administrar mejor el espacio físico de nuestro dispositivo, ya sea por la cantidad de fotos que hacemos, de videos o simplemente documentos que almacenamos en nuestro ordenador o tablet.

MySky da solución a todos estos problemas. MySky es un servidor remoto y de uso personal donde puedes guardar tus datos personales, tales como fotos, vídeos, música y cualquier tipo de documento de forma segura en cualquier momento desde tu smartphone, Tablet u ordenador.

MySky te permite ahorrar espacio en la memoria física del dispositivo. Podrás subir cualquier tipo de archivo al servidor y de esta manera, poder borrar de forma segura el archivo de tu dispositivo. ¡Estés donde estés! De esta manera, podrás consultar tus archivos guardados en el servidor.

* 1. OBJETIVOS
* Crear un espacio de uso personal o familiar, donde los usuarios registrados podrán compartir entre sí los distintos archivos cargados al servidor.
* Cubrir la necesidad del usuario de ahorrar espacio físico en sus dispositivos.
* Crear una copia de seguridad de sus datos para evitar perderlos en caso de estar almacenados en una memoria física del dispositivo.
* Gracias a MySky podrá acceder a sus datos desde cualquier dispositivo, sea móvil, ordenador o Tablet.
* Es totalmente seguro. Se implementará en el servidor una capa de seguridad para que tus datos puedan mantenerse de forma segura.
* Portabilidad y multiplataforma gracias a la tecnología Docker. El servidor será ejecutable desde cualquier dispositivo con cualquier Sistema operativo que sea capaz de soportar la tecnología Docker.
  1. ANÁLISIS

El análisis de la aplicación se compone principalmente de dos partes:

#### 3.1.- Requisitos funcionales.

En esta fase describiremos las interacciones que tendrá el usuario con la aplicación.

##### 3.1.1.- Pantalla de bienvenida.

* La vista de bienvenida contiene un total de tres botones. El botón *regístrate* le redirecciona a la vista de /signup. El botón *Iniciar sesión* le redirecciona a la vista de /login. Además también dispone de un botón *Acerca de MySky* donde le redireccionará a otra visa la cual contiene una breve descripción de lo que es MySky.

##### 3.1.2.- registro.

* La aplicación mostrará un formulario donde el usuario deberá de introducir los datos correspondientes a cada campo según se indique.
* El sistema validará los datos introducidos por el usuario.
* En caso de que el usuario haya introducido algún dato erróneo, sea por el formato del correo, por insertar un correo ya existente en la base de datos o por no introducir la contraseña correctamente en ambos campos destinados para ello, mostrará un mensaje de error.
* En caso de que los datos sean correctos y el servidor haga una validación de dichos datos correcta, se mostrará un mensaje de éxito en el registro.
* La aplicación finalmente le enviará a la pantalla principal de perfil.

##### 3.1.3.- Inicio de sesión.

* La aplicación mostrará un formulario con dos campos para introducir el correo con el que se registró y la contraseña.
* El sistema validará los datos introducidos por el usuario.
* Una vez comprobados, si los datos son correctos, el será enviado a la pantalla principal del perfil.
* En caso de que sean erróneos, sea por insertar mal el formato del correo, por introducir un correo no registrado o una contraseña que no se corresponde, mostrará un mensaje de error.

##### 3.1.4.- Cierre de sesión.

* En todo momento, el usuario loggeado tendrá disponible un botón para poder cerrar sesión y salir de la aplicación.
* En el caso de que el usuario presione el botón cerrar sesión, la sesión finalizará y el usuario será redirigido a la pantalla principal de MySky.

##### 3.1.5.- pantalla principal del perfil.

* Se presentarán cuatro botones, a aparte del botón de cierre de sesión.
* El botón *Elegir archivos* servirá para acceder al gestor de archivos de su equipo para seleccionar uno o más archivos que el usuario quiera almacenar en el servidor.
* El otro botón *subir* cargará el archivo o archivos seleccionados en el servidor, dentro de un directorio llamado /uploads. Una vez pulsa éste botón, el usuario es redireccionado a una nueva pagina indicando que la subida ha sido exitosa y automáticamente, pasados dos segundos será redireccionado de nuevo al perfil principal.
* El botón *ver archivos* redireccionará al usuario un end point llamado /files donde se listara un JSON con los archivos subidos en el servidor actualmente.
* En todo momento, el logo de MySky hace de botón de inicio.
* En todas las páginas por las que el usuario puede navegar en todo momento dispondrá de un enlace en el nombre del creador de la web, donde le redireccionará al perfil de github del propietario.

#### 3.2.- Requisitos NO funcionales.

##### 3.2.1.- DOCUMENTACIÓN.

Documentación sobre el sitio web para informar al usuario del servicio que realiza la aplicación.

##### 3.2.2.- SEGURIDAD.

Seguridad de la aplicación. Solo aquellos usuarios registrados podrán utilizar el servicio de MySky. La contraseña del usuario pasa por un algoritmo de encriptación a la hora de ser guardada en la base de datos.

##### 3.2.3.- PORTABILIDAD.

La aplicación estará contenida en un contenedor de Docker, lo que permitirá instalarse en cualquier dispositivo que acepte la tecnología de Docker y de esta forma, utilizarse como servidor.

##### 3.2.4.- Interfaz.

La aplicación estará construida bajo una interfaz elegante, minimalista e intuitiva. El usuario no tendrá ninguna dificultad para saber cómo manejar la aplicación.

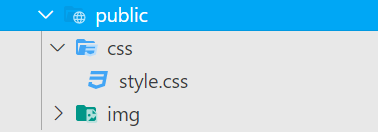
##### 3.2.4.- eficiencia.

La aplicación no consume muchos recursos. Es bastante ligera y además contenida en un contenedor de Docker.

* 1. DISEÑO

Estructura de directorios de la ap­­­licación

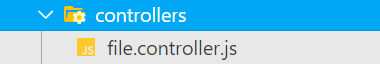
La estructura de la aplicación se compone de tres directorios principales:

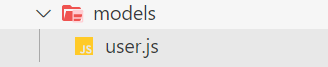
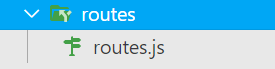
* /public: En este directorio se encuentran todos los archivos estáticos de configuración de vistas, como la hoja de estilos css así como todas las imágenes utilizadas para la creación de dichas vistas.
* /config: En este directorio se guardan todos los archivos de configuración relacionados con la conexión a la base de datos de MongoDB y los archivos encargados de garantizar la comunicación entre las rutas del servidor.

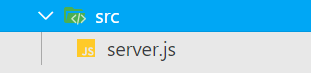


* /controllers: En este directorio se almacenan los archivos encargados de hacer

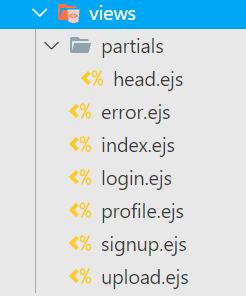
el control de gestión de la visualización de archivos desde el servidor.



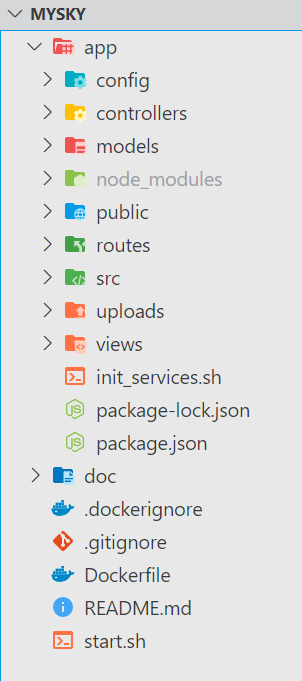
* /models: En este directorio se guardan los archivos que contienen los modelos que componen las colecciones de la base de datos.
* /routes: En este directorio se guardan los archivos encargados de realizar la comunicación de los archivos mediante la declaración de las distintas rutas (end points) que tendrá nuestro servidor.
* /src: Este directorio contiene el archivo principal que contiene toda la configuración del servidor. Este es el archivo que ejecuta la aplicación.



* /views : En este directorio se guardan todos los archivos referentes a las vistas. Esto es: al front end de la aplicación, es decir, la interfaz de usuario.



El directorio completo de la aplicación es el siguiente:

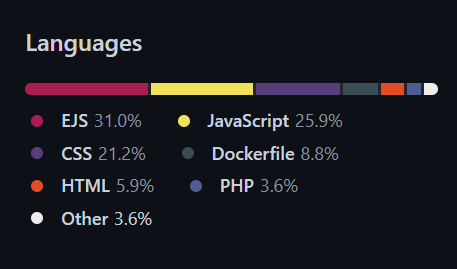


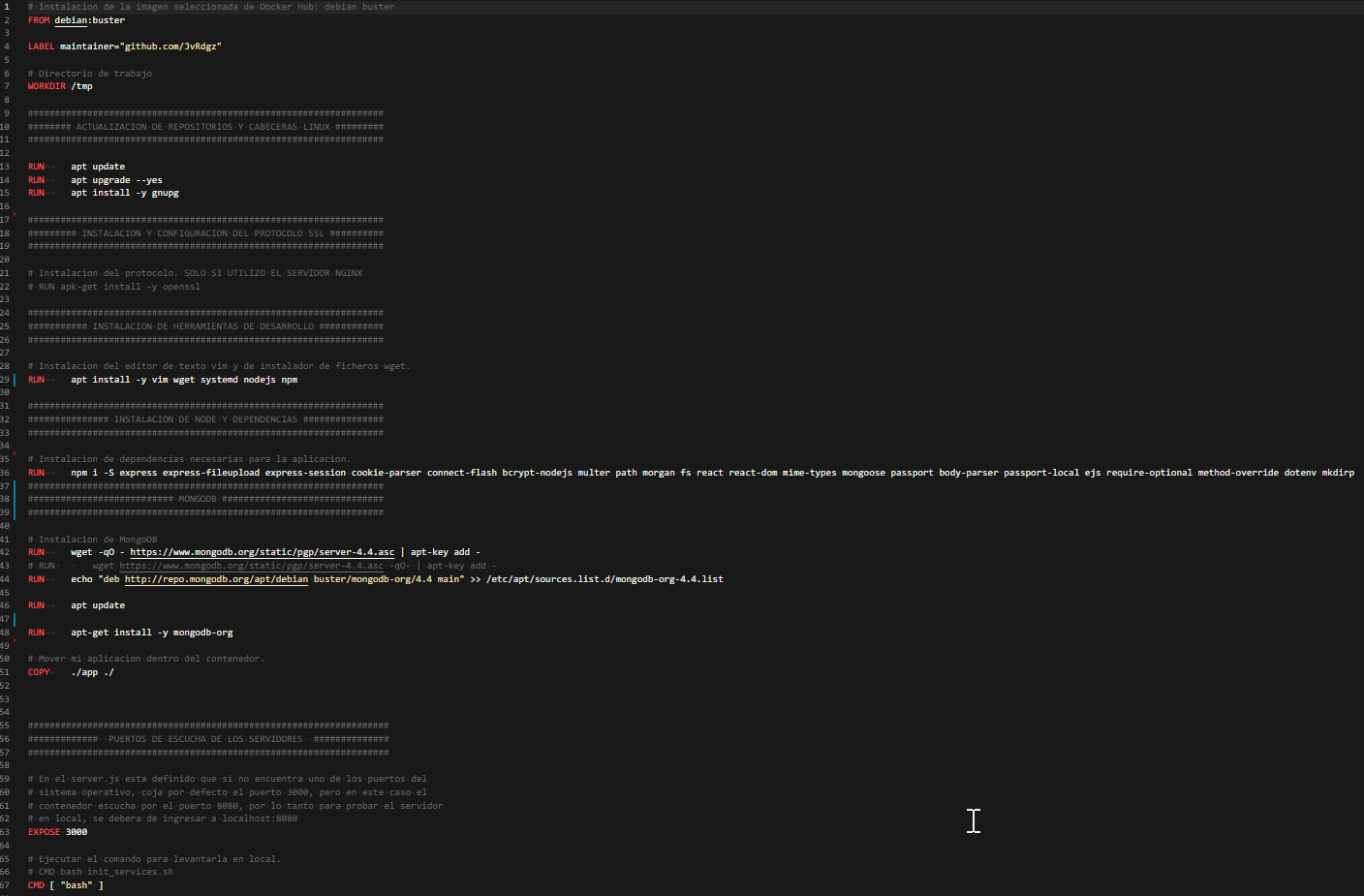
En el directorio principal, a parte de los directorios descritos anteriormente que componen la aplicación, también encontramos el directorio de dependencias */node\_modules* que contiene todos los archivos de configuración de dependencias propias de NodeJs. También se encuentras los archivos *package.json* y *package-lock.json* que contienen la configuración de la aplicación y las versiones de las dependencias de NodeJs utilizadas. En el mismo directorio principal, también encontramos un archivo llamado init\_services.sh el cual contiene una configuración muy básica para iniciar los servicios de forma automática mediante bash, que es ejecutado por el archivo de configuración *Dockerfile*, una vez éste haya construido previamente nuestro contenedor y descargado nuestra imagen.

También contiene un directorio llamado */uploads* donde se guardarán los archivos que el usuario quiera guardar en el servidor.

* 1. PROGRAMACIÓN

La aplicación está construida mediante el lenguaje de programación Nodejs que se posiciona desde el lado del servidor, y mediante el motor de vistas EJS y Javascript que se posicionan ambos desde el lado del cliente. Podemos verlo en esta imagen:

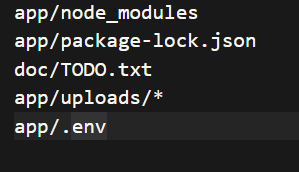


También interactúan otra serie de tecnologías como es Docker, donde se ha utilizado para crear una hoja de instrucciones Dockerfile para definir la estructura de la imagen que vamos a crear de nuestra aplicación. Allí se incluyen los comandos necesarios para instalar las herramientas necesarias para poder ejecutar nuestra aplicación sobre el contenedor de Docker.

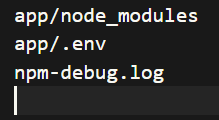
El archivo *start.sh* es ejecutado desde el terminal Linux del ordenador/dispositivo que vaya a hacer de servidor. La aplicación está automatizada, es decir, que simplemente ejecutando ese archivo comenzará el despliegue automático de la misma. El comando para ejecutarlo es el siguiente: *sh start.sh*. Es un pequeño script que contiene los comandos necesarios para instalar el contenedor y ejecutar la imagen:

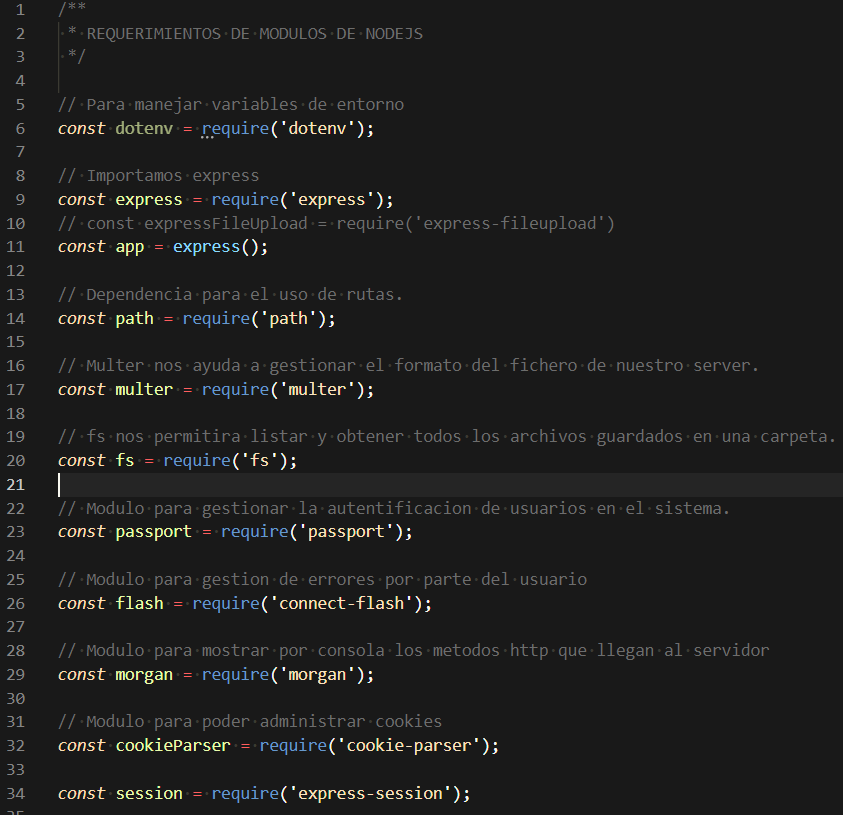
Los archivos .gitignore y .dockerignore simplemente contienen unas instrucciones que evitan que se carguen a los respectivos repositorios carpetas innecesarias o información sensible.

Archivo .gitignore:

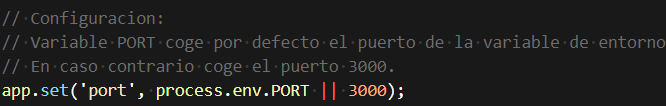


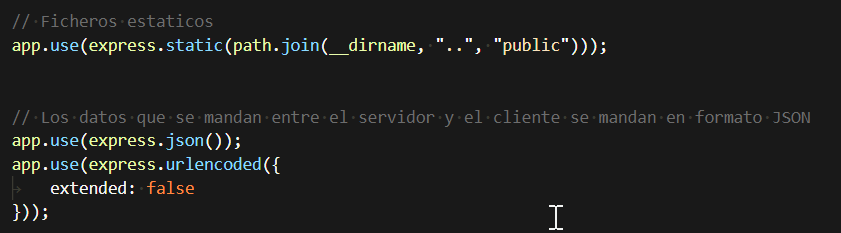
Archivo .dockerignore:



El código principal que habilita al servidor, es decir, el código donde se crea el servidor que escucha las peticiones se encuentra en el archivo *server.js* dentro de /src. En el se llaman a todos los módulos que va a requerir nuestra aplicación.

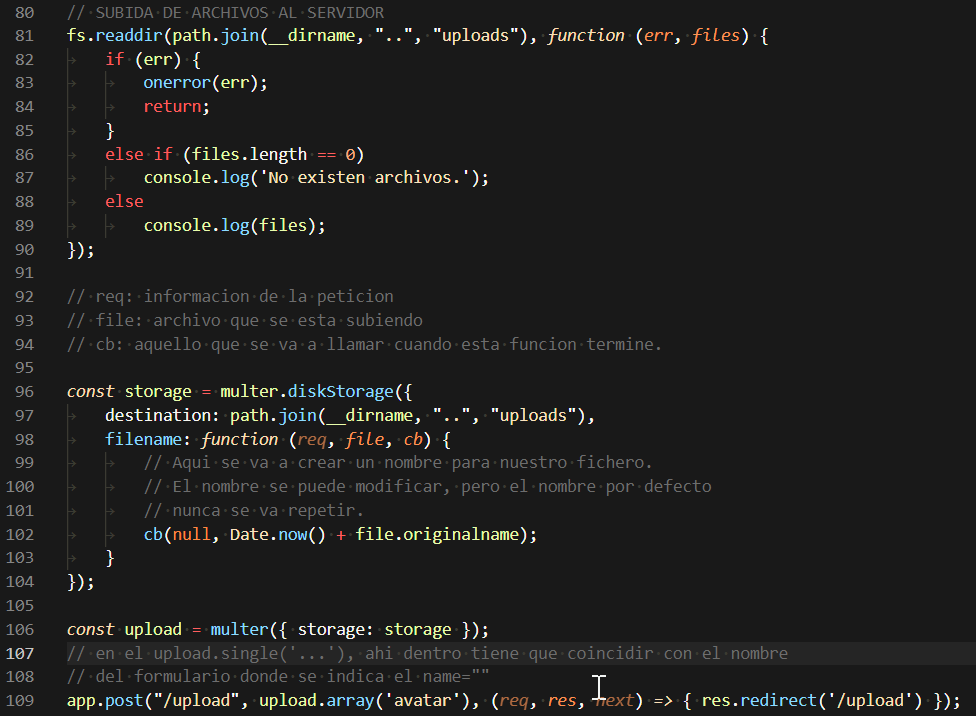
Mas adelante, se asigna el puerto de escucha, que oscilará entre el puerto 3000 o el puerto por defecto declarado en la variable de entorno. En este caso cogerá el puerto 3000 dado que no hemos asignado ninguna variable de entorno.

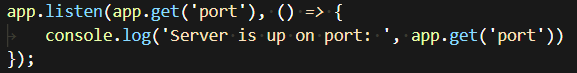


A continuación, se hace la configuración del módulo requerido *express* el cual nos facilitará todo lo que necesitamos para desarrollar el servidor de una forma más cómoda y eficiente. Indicamos que los datos que se manden del servidor al cliente y del cliente al servidor se manden en formato JSON.

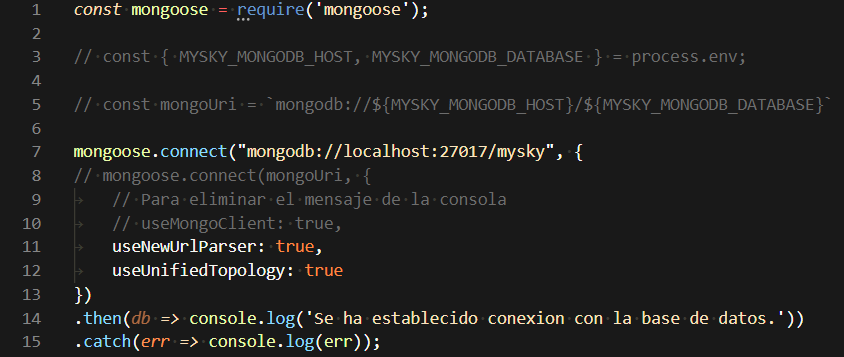
Se crea la configuración del uso de cookies gracias al módulo *cookie-parser* y a *express-session.* Habilitamos el uso del módulo Passport que nos ayudará a gestionar la autentificación del usuario en la aplicación.

**

Más tarde, mediante el módulo multer, se realiza la gestión para la subida de archivos a nuestro servidor, y se realiza la petición POST para la carga.

Finalmente, se habilita la escucha del servidor en el puerto configurado previamente.

Para la conexión a la base de datos, se crea un archivo de configuración llamado *database.js* el cual contiene la configuración y a conexión a la base de datos local de mongodb mysky.

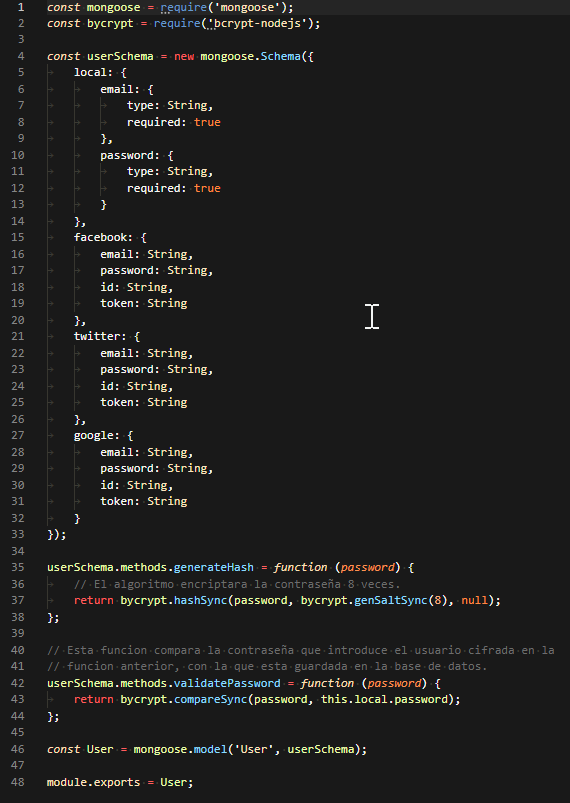


El archivo *Passport.js* se encarga de realizar la autentificación con el usuario y de guardar las credenciales del usuario en la base de datos a través de una colección llamada *users*.

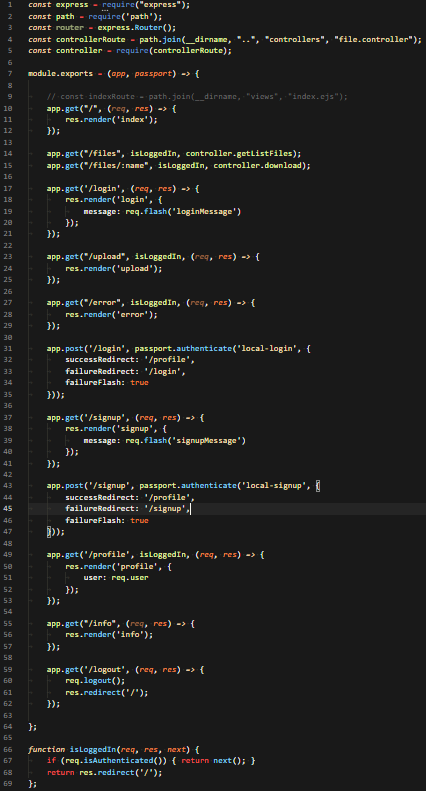
También existe un controlador de archivos que se encarga de gestionar los archivos ya subidos en el servidor y realiza la tarea de listarlos y de descargarlos (esta última función aún no está implementada, saldrá en próximas versiones).



El archivo user.js contiene el modelo del que se componen los usuarios que se van a guardar en la base de datos de forma local. (Está configurado para poder iniciar sesión mediante tokens de las distintas redes sociales y cuenta de Google, pero esa función no está implementada).



Para finalizar, el archivo *routes.js* se encarga de definir las diferentes rutas según el tipo de petición que se realizará. También contiene un control de seguridad para que solo se pueda acceder a rutas autorizadas exclusivamente si el usuario esta loggeado en la aplicación.



* 1. Conclusiones

Finalizamos este documento recalcando que se han cumplido todos los objetivos principales y mandatorios que se han descrito al principio de éste. La aplicación se mantendrá de cara a un futuro y se irán realizando mejoras e implementando nuevas funcionalidades. Se trata de un software ligero y eficiente para un uso personal o familiar, donde los usuarios registrados podrán compartir entre sí un espacio común, donde encontrar los distintos archivos cargados al servidor.

FIN.