FRONT-END PARA NAVEGADORES WEB

SEGUNDA PRÁCTICA – PROCESO DE REGISTRO Y RESERVAS – JAVASCRIPT, JQUERY Y ANGULAR

El objetivo de la segunda práctica es que el alumno amplíe sus conocimientos en el diseño e implementación de la parte de conexión con el servidor utilizando tecnologías asíncronas al mismo tiempo puede utilizar un entorno de desarrollo específico como es Angular.

En esta práctica se deberán resolver otros de los procesos asociados a dos de las opciones del menú principal mostrado en la primera práctica y correspondiente a la gestión del club de pádel. Los procesos a resolver son los de registro y el de reservas. Al igual que en la primera práctica, el sistema debe ser una aplicación "single page" (o de página única), y se deberá desarrollar usando el framework ***Angular***.

ATENCIÓN:

Al igual que en la primera práctica, se utilizará una API REST que proporciona el servicio de "back-end" a la aplicación (gestión de usuarios y de reservas de pistas para poder jugar al pádel). Esta API REST está a disposición de los alumnos para que puedan probar y validar el código generado antes de ser presentado para su evaluación. Las características básicas de la API REST son:

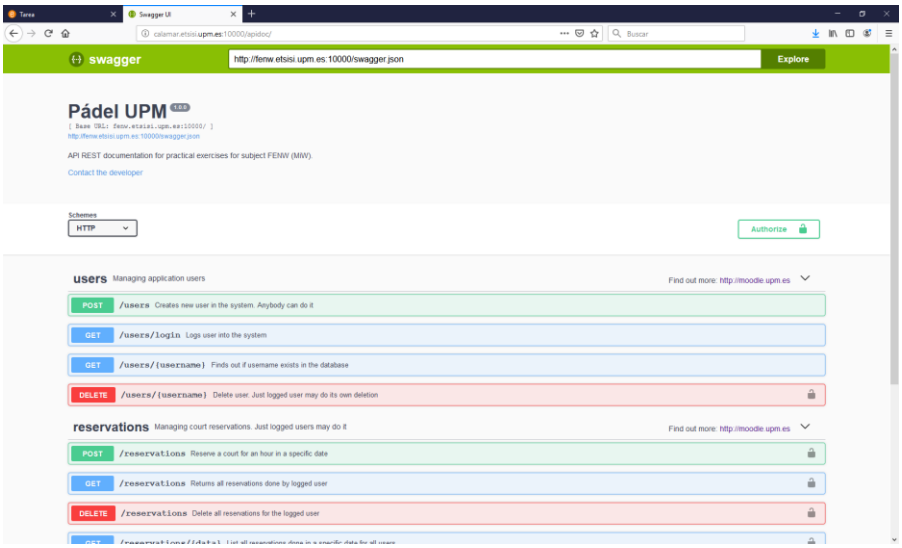
URL base: **http://fenw.etsisi.upm.es:10000** **(Nótese el uso del puerto 10000)**

La API se encuentra suficientemente documentada a través de la herramienta "swagger" en la url http://fenw.etsisi.upm.es:10000/apidoc

Esta página de especificación permite, por una parte, comprobar las rutas y métodos accesibles; por otra, comprobar los parámetros necesarios para su ejecución así como los resultados que la misma proporciona como respuesta; y por último, probar que se entiende su funcionamiento a través de la ejecución de los métodos disponibles en la misma página.

Resulta **importante** destacar que, en aquellas solicitudes de alta, utilizando el método POST, los parámetros pasados a la API, en el **body**, se pueden pasar como parámetros independientes (urlencoded), (**username=nuevonombre&email=nuevoemail&password=nuevapasswd)** o como un objeto json cuyas propiedades son los nombres de los parámetros citados (**{username:nuevonombre, email:nuevoemail, password:nuevapassword}**). Sin embargo, todos las respuestas recibidas desde la API REST estarán en formato **json** (incluso aunque sea una simple cadena de caracteres o un número).

Todas las solicitudes a las rutas que conciernen a la gestión de reservas así como el borrado de un usuario necesitan, de manera obligatoria, que se le pase como contenido de la cabecera "***Authorization***" la palabra "***Bearer***" y el ***token*** de autorización obtenido en la operación de *login* (separados por un espacio en blanco). Dado que el *token* tiene una validez de 10 minutos, la API REST, cada vez que realiza una operación**, actualiza y proporciona un nuevo *token* con otros 10 minutos de validez**, de tal manera que caduca solamente si el usuario no solicita ninguna operación durante ese tiempo.



NOTA: El alumno rehará la primera práctica de tal manera que todo el código, incluida la opción de "*login*" sea desarrollada con el *framework Angular*.

Funcionalidad de la opción de "Registro"

Esta opción permitirá al usuario proceder al registro necesario para poder conectarse al sistema y **no se podrá ejecutar** si el usuario se encuentra ya conectado al mismo (ha realizado ya la operación de *login* con éxito y por tanto dispone de un *token* válido de conexión). Una vez elegida la opción, el sistema deberá mostrar el formulario diseñado a tal efecto en la primera práctica. El usuario deberá rellenar el formulario, siendo obligatorios todos los campos excepto el de la fecha de nacimiento.

El sistema deberá comprobar la unicidad del nombre de usuario **antes** de que se proceda al envío de los datos por parte del usuario. Para ello, cuando el usuario pase de un campo a otro (el elemento en cuestión pierda el foco), se deberá proceder a ejecutar una llamada asíncrona a la API REST disponible pasándole el valor del identificador **username** como parte de la URL:

Método: **get** url: http://fenw.etsisi.upm.es:10000/users/**{username}**

Las diferentes respuestas a esta petición están disponibles en la documentación indicada

Una vez pasadas estas comprobaciones (que en caso de no producir errores pasan inadvertidas para el usuario), el usuario debe poder mandar (al pulsar el botón de "enviar") todos los datos: "**username**", "**email**", "**password**" (todas cadenas de caracteres) y "**birthDate**" que será la fecha de nacimiento implementada en base a un número entero que se corresponde con el número de milisegundos transcurridos desde el 1 de enero de 1970 (*getTime() en tipo Date()*).

Este envío se realizará, otra vez, a la API REST

Método: **post** url: http://fenw.etsisi.upm.es:10000/users

Las diferentes respuestas a esta petición están disponibles en la documentación indicada.

El sistema debe informar del resultado del proceso final al usuario mediante un pequeño mensaje.

En este apartado se podrá incorporar, de manera **optativa**, un sistema de garantía de que el *registro* no se está efectuando mediante un programa "robot" con el desplazamiento de una imagen de una caja a otra por "drag&drop" (o sistema similar). En caso de usar una librería existente, el alumno debe asegurarse que no tenga necesidades especiales de funcionamiento (ni instalación de software ni requisitos de ejecución diferentes al framework).

Funcionalidad de la opción de "Reserva"

La opción de "Reserva" se refiere a la posibilidad que tiene el usuario de reservar la utilización de una pista de pádel en una hora determinada de una fecha concreta siempre y cuando no se encuentre ya reservada. **Solamente podrá realizar una reserva si previamente el usuario ha realizado la operación de autenticación (*login*) con éxito (dispone de un *token* válido)**. Todas las peticiones a las rutas concernientes a la gestión de reservar requieren que se les pase en la cabecera "*Authorization*" el *token* correspondiente. Un usuario concreto solamente podrá efectuar un máximo de **cuatro** reservas (hecho del que se encarga la API REST y por tanto no hay que comprobar en el cliente).

El alumno podrá elegir el interfaz que se presentará al usuario para poder efectuar las reservas de pistas. Por ejemplo, se puede optar por presentar las reservas ya efectuadas (pistas ocupadas) a lo largo de toda la semana (y por tanto las que no están ocupadas estarán libres para ser reservadas) o elegir presentar solamente las reservas de un determinado día. Por lo tanto, la funcionalidad podría ser **algo similar a**:

1) El usuario se identifica mediante la opción *login* y obtiene su *token* de identificación
2) El usuario elige, de un calendario (un elemento tipo "*date*" o "*datepicker*"), una **fecha** y la envía a la ruta correspondiente de la API REST junto con el *token* que le autoriza. Hay que indicar que, según se puede ver en la documentación, la fecha se deberá pasar como un número entero de formato *getTime*() de javascript (milisegundos transcurridos desde el 1 de enero de 1970).

   El sistema recibe las reservas ya efectuadas para la fecha (o sea, pistas **NO** disponibles) y muestra las **pistas** (atención: el club dispone **de cuatro pistas**, **numeradas del 1 al 4**) y **horas** disponibles en esa fecha, entre las que el usuario efectuará su elección. Una vez elegida, se enviará a la ruta correspondiente la pista elegida y la fecha en número entero, que esta vez incluirá la hora (será el *getTime* de **fecha y hora**), además del *token* que le autoriza.

   ATENCIÓN: La reservas **solamente** se podrán **hacer por horas completas, a la hora en punto y en horario de 10:00 a 21:00** (o sea, de 10:00 a 11:00, o de 13:00 a 14:00 o de 16:00 a 17:00, etc)

   Las diferentes respuestas a esta petición están disponibles en la documentación indicada.

El alumno es libre de elegir aquella representación en pantalla que considere oportuna en todo el proceso, valorándose el diseño de la pantalla de presentación de las reservas disponibles en determinado espacio de tiempo (por días o semanas, etc).

SE PIDE:

Se debe entregar (subir a la plataforma) un fichero comprimido que contenga todos aquellos elementos para que, una vez descomprimido, la práctica pueda visualizarse y ejecutarse con normalidad. La descompresión del fichero debe dar lugar a la jerarquía de directorios necesaria para situar cada fichero donde corresponda, incluyendo aquellos que puedan pertenecer a las librerías si no se utiliza CDN para su incorporación.

CALIFICACIÓN

- ✓ La corrección de la práctica se realizará valorando los aspectos indicados en la rúbrica correspondiente publicada en la plataforma.