Modelo - PRIMER PARCIAL – LAB. III *2022*

Se debe crear un archivo por cada entidad de TypeScript. Todos los métodos deben estar declarados dentro de clases.
Toda la comunicación con el backend, se realizará con AJAX. El pasaje de datos, con JSON.
El backend lo debe proveer el alumno (respetando nombres y tipos de parámetros).
Se deben respetar los nombres de los archivos, de las clases, de los métodos y de los parámetros.

Parte FRONTEND

Crear la siguiente jerarquía de clases en **Typescript** (en el namespace **Entidades**):

a. ***Persona***: nombre(cadena), correo(cadena) y clave(cadena) como atributos. Un constructor que reciba dos parámetros. Un método, *ToString():string,* que retorne la representación de la clase en formato **cadena** (preparar la cadena para que, al juntarse con el método ToJSON, forme una cadena JSON válida).

b. ***Usuario***, hereda de Persona, posee como atributo id(entero), id_perfil(entero) y perfil(cadena). Un constructor para inicializar los atributos. Un método *ToJSON():JSON,* que retornará la representación del objeto en formato **JSON**. Se debe de reutilizar el método *ToString* de la clase Persona.

c. ***Empleado***, hereda de Usuario, posee como atributos id(entero), sueldo(numérico) y foto(cadena). Un constructor para inicializar los atributos.

Crear en **TypeScript** la clase ***Manejadora*** (en el namespace **ModeloParcial**) que posea los siguientes métodos y funcionalidades:

AgregarUsuarioJSON. Obtiene el nombre, el correo y la clave desde la página ***usuario_json.html*** y se enviará (por ***AJAX***) hacia ***"./BACKEND/AltaUsuarioJSON.php"*** *que invocará al método de instancia GuardarEnArchivo(), que agregará al usuario en **./backend/archivos**/usuarios.json. Retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido*.
Informar por **consola** y **alert** el mensaje recibido.

MostrarUsuariosJSON. Recuperará (por ***AJAX***) todos los usuarios del archivo ***usuarios.json*** y generará un listado dinámico, crear una tabla HTML con cabecera (en el ***FRONTEND***) que mostrará toda la información de cada uno de los usuarios. Invocar a ***"./BACKEND/ListadoUsuariosJSON.php"****, recibe la petición (por GET) y retornará el listado de todos los usuarios en formato JSON.*

VerificarUsuarioJSON. Verifica que el usuario exista. Para ello, invocará (por ***AJAX***) a ***"./BACKEND/VerificarUsuarioJSON.php"***. *Se recibe por POST parámetro **usuario_json** (correo y clave, en formato de cadena JSON).*
*Retornar un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido (agregar el mensaje obtenido del método de clase).*
Se mostrará (por **consola** y **alert**) lo acontecido.

AgregarUsuario. Obtiene el nombre, el correo, la clave y el id_perfil desde la página ***usuario.html*** y se enviará (por ***AJAX***) hacia ***"./BACKEND/AltaUsuario.php"*** *que recibe por POST el **nombre**, el **correo**, la **clave** e **id_perfil**. Se invocará al método Agregar.*
*Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.*
Informar por **consola** y **alert** el mensaje recibido.

MostrarUsuarios. Recuperará (por ***AJAX***) todas los usuarios de la base de datos, invocando a ***"./BACKEND/ListadoUsuarios.php"***, *recibe la petición (por GET) y mostrará el listado **completo** de las usuarios (obtenidas de la base de datos) en una tabla (HTML con cabecera). Invocar al método Traer (retorna un array de objetos de tipo Usuario, recuperados de la base de datos.).*
Informar por **consola** y **alert** el mensaje recibido y mostrar el listado en la página (*div id='divTabla'*).

ModificarUsuario. Mostrará todos los datos del usuario que recibe por parámetro (objeto ***JSON***), en el formulario. Permitirá modificar cualquier campo, a excepción del id, dejarlo como de sólo lectura.
Al pulsar el botón Modificar usuario se invocará (por ***AJAX***) a ***"./BACKEND/ModificarUsuario.php"***, *que recibirán por POST los siguientes valores: **usuario_json** (id, nombre, correo, clave y id_perfil, en formato de cadena JSON), para modificar un usuario en la base de datos. Invocar al método Modificar.*
Retornar un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.
Refrescar el listado sólo si se pudo modificar, caso contrario, informar (por alert y consola) de lo acontecido.

EliminarUsuario. Recibe como parámetro al objeto ***JSON*** que se ha de eliminar. Pedir confirmación, mostrando nombre y correo, antes de eliminar.
Si se confirma se invocará (por ***AJAX***) a ***"./BACKEND/EliminarUsuario.php"*** *pasándole cómo parámetro el **id** por POST y se deberá borrar el usuario (invocando al método Eliminar).*
*Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.*
Informar por consola y alert lo acontecido. Refrescar el listado para visualizar los cambios.

NOTA:
*Agregar una columna (Acciones) al listado de usuarios que permita: **Eliminar** y **Modificar** al usuario elegido. Para ello, agregue dos botones (input [type=button]) que invoquen a las funciones EliminarUsuario y ModificarUsuario, respectivamente.*

En la clase Manejadora, agregar los siguientes métodos:

AgregarEmpleado, MostrarEmpleado, EliminarEmpleado y ModificarEmpleado.

Al cargar la página empleado.html, se deberá cargar el listado de empleados obtenidos desde la base de datos, para ello se invocará al método **MostrarEmpleados** que enviará (desde AJAX) hacia **"./BACKEND/ListadoEmpleados.php"**, una petición por GET, que mostrará el listado **completo** de los empleados (obtenidos de la base de datos) en una tabla (HTML con cabecera). Invocar al método TraerTodos.
***Nota**: preparar la tabla (HTML) con una columna extra para que muestre la imagen de la foto (50px X 50px).*
Mostrar el listado en la página (*div id='divTablaEmpleados'*).

AgregarEmpleado. Obtiene los datos del empleado (incluyendo la foto) desde la página ***empleado.html*** y se enviará (por ***AJAX***) hacia ***"./BACKEND/AltaEmpleado.php"*** *que recibirá por POST todos los valores: **nombre**, **correo**, **clave**, **id_perfil**, **sueldo** y **foto** para registrar un empleado en la base de datos..*
*Se retornará un **JSON** que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.*
***Nota:** La foto guardarla en "./backend/empleados/fotos/", con el nombre formado por el nombre punto tipo punto hora, minutos y segundos del alta (Ejemplo: juan.105905.jpg).*
Informar por **consola** y **alert** el mensaje recibido.

Refrescar el listado de empleados.

ModificarEmpleado. Mostrará todos los datos del usuario que recibe por parámetro (objeto ***JSON***), en el formulario, incluida la foto (mostrarla en ***"imgFoto"***). Permitirá modificar cualquier campo, a excepción del id, dejarlo como de sólo lectura.
Al pulsar el botón Modificar empleado se invocará (por ***AJAX***) a ***"./BACKEND/ModificarEmpleado.php"***, *que recibirán por POST los siguientes valores:* ***empleado_json*** *(id,* nombre, correo, clave, id_perfil, sueldo y pathFoto, en formato de cadena JSON) *y* ***foto*** (*para modificar un empleado en la base de datos. Invocar al método Modificar.*
Nota*: El valor del id, será el id del empleado 'original', mientras que el resto de los valores serán los del empleado modificado.*
Refrescar el listado sólo si se pudo modificar, caso contrario, informar (por alert y consola) de lo acontecido.

EliminarEmpleado. Recibe como parámetro al objeto ***JSON*** que se ha de eliminar. Pedir confirmación, mostrando nombre y sueldo, antes de eliminar.
Si se confirma se invocará (por ***AJAX***) a ***"./BACKEND/EliminarEmpleado.php"*** *pasándole cómo parámetro el* ***id*** *por POST y se deberá borrar el empleado (invocando al método Eliminar).*
Se retornará un ***JSON*** *que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.*
Informar por consola y alert lo acontecido. Refrescar el listado para visualizar los cambios.