

PRIMER PARCIAL – LABORATORIO III

1 cuat. 2024

Aclaraciones:

Las partes se corregirán de manera secuencial (ascendentemente). Si están bien todos los puntos de una parte, habilita la corrección de la parte posterior.

Se debe crear un archivo por cada entidad de TypeScript. Todos los métodos deben estar declarados dentro de clases.

Toda comunicación con el backend se realizará con AJAX/FETCH. Todo el pasaje de datos se hará con JSON.

Las vistas (páginas .php o .html) se entregan por parte del profesor y se deberán alojar en el 'localhost'.

Se deben respetar los nombres de los archivos, de las clases, de los métodos y de los parámetros de las peticiones.

Todas las referencias a archivos y/o recursos, deben estar de manera relativa.

El backend lo provee el profesor (usted debe respetar nombres y tipos de parámetros) e interactúa con la base de datos **buzon** (sobres y postales).

Parte 1 FRONTEND – HTML5 y TypeScript (hasta 5)

Crear la siguiente clase en **Typescript** en el namespace **Apellido** (del alumno):

- **Sobre**, posee como atributos protegidos:
 - direccion_destinatario (cadena)
 - remitente (cadena)
 - precio_estampilla (numérico)

Un constructor (que inicialice los atributos), un método de instancia **toJSON()**, que retornará los datos de la instancia (en una cadena con formato **JSON**).

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

AgregarSobre. Obtiene la **direccion_destinatario**, el **remitente** y el **precio_estampilla** desde la página **sobres.html** y se enviará (por AJAX/FETCH) hacia **"http://localhost:2024/sobre"** por el método POST. Retornará un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Informar por consola y alert el mensaje recibido.

MostrarSobres. Recuperará (por AJAX/FETCH) todos los sobres de la tabla **sobres** de la base de datos **buzon**, invocando a **"http://localhost:2024/sobre"**, que recibe la petición (por GET) y retornará un JSON (éxito:true/false, sobres:array/null) para crear un listado dinámico (en el FRONTEND) que mostrará toda la información de cada uno de los sobres.

VerificarSobre. Verifica que el sobre exista. Para ello, invocará (por AJAX/FETCH) a **"http://localhost:2024/sobre/remitente"**. Se recibe por GET (cómo parámetro de ruta) el **remitente** y retornará un JSON (éxito:true/false, sobres:array/null) para crear un listado dinámico (en el FRONTEND) que mostrará toda la información de cada uno de los sobres (Este listado **no** tendrá la columna **'Acciones'**).

Si no existe ningún sobre, se mostrará (por consola y alert) lo acontecido.

Apellido y nombre del alumno

CRUD - Sobres

Id	<input type="text"/>
Dirección destinatario	<input type="text"/>
Remitente	<input type="text"/>
Precio estampilla	<input type="text"/>

Agregar

Mostrar

Verificar

Modificar

tabla aquí...

sobres.html

NOTA: vincular los eventos 'click' de cada botón a la función correspondiente.

Parte 2 FRONTEND – HTML5 y TypeScript (hasta 6)

Del listado generado dinámicamente por el método **MostrarSobres**, agregar una columna (**Acciones**) que permitan: **Eliminar** y **Modificar** el sobre elegido.

Para ello, agregue dos botones (input [type=button]) que invoquen a las funciones (de la clase Manejadora), EliminarSobre, Modificar y ModificarSobre, respectivamente.

Modificar. Mostrará todos los datos del usuario que recibe por parámetro (objeto JSON), en el formulario principal. Permitirá modificar cualquier campo, a excepción del **id**, dejarlo como de sólo lectura.

Al pulsar el botón **Modificar** se invocará al método **ModificarSobre** que enviará (por AJAX/FETCH) a **"http://localhost:2024/sobre"**, que recibirán por PUT los siguientes valores: **sobre_json** (id, direccion_destinatario, remitente y precio_estampilla, en formato de cadena JSON), para modificar un sobre en la base de datos.

Retornará 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.

EliminarSobre. Recibe como parámetro al objeto JSON que se ha de eliminar. Pedir confirmación, mostrando la *dirección del destinatario* y el *remitente*, antes de eliminar.

Si se confirma se invocará (por AJAX/FETCH) a **"http://localhost:2024/sobre"** pasándole como parámetro el **id** por DELETE y se deberá borrar el usuario.

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.

Apellido y nombre del alumno

CRUD - Sobres

Id

Dirección destinatario

Remitente

Precio estampilla

Agregar

Mostrar

Verificar

Modificar

Id	Dirección Destinatario	Remitente	Precio estampilla	Acciones	
2	bernal	venecia	215	Eliminar	Modificar
3	lomas	solano	1455	Eliminar	Modificar
7	novaresio 988	juan alcorta 2233	10.66	Eliminar	Modificar

Agregar la siguiente clase en **Typescript** en el namespace **Apellido** (del alumno):

- **Postal**, que deriva de Sobre y posee como atributo público:
 - imagen (cadena)

Un constructor (que inicialice los atributos), un método de instancia **toJSON()**, que retornará los datos de la instancia (en una cadena con formato **JSON**).

Crear la clase **ManejadoraPostales** agregándole los siguientes métodos:

AgregarPostal y MostrarPostales.

Al cargar la página **postales.html**, se deberá cargar el listado de postales obtenidas desde la base de datos, para ello se invocará al método **MostrarPostales** que enviará (desde AJAX/FETCH) hacia **"http://localhost:2024/postal"**, una petición por GET, que retornará un JSON (éxito:true/false, postales:array/null) para crear un listado dinámico (en el FRONTEND).

Nota: preparar la tabla (HTML) con una columna extra para que muestre la imagen de la postal (50px X 50px).


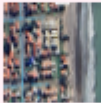
Mostrar el listado en la página (div id='divTablaPostales').

AgregarPostal. Obtiene los datos de la postal (incluyendo la imagen) desde la página **postales.html** y se enviará (por AJAX/FETCH) hacia **"http://localhost:2024/postal"** que recibirá por POST **obj_postal** (direccion_destinatario, remitente y precio_estampilla, en formato de cadena JSON) e **imagen** para registrar una postal en la base de datos.

Se retornará un JSON que contendrá: éxito(bool) y mensaje(string) indicando lo acontecido.

Informar por consola y alert el mensaje recibido.

Refrescar el listado de las postales.

Id	Dirección Destinatario	Remitente	Precio estampilla	Imagen	Acciones
7	Destinatario 123	Remitente 123	808		<div>Eliminar</div> <div>Modificar</div>
8	otro destinatario	otro remitente	202		<div>Eliminar</div> <div>Modificar</div>

Parte 3 FRONTEND – HTML5 y TypeScript

En la clase **ManejadoraPostales**, agregar los siguientes métodos:

EliminarPostal, Modificar y ModificarPostal.

Modificar. Mostrará todos los datos de la postal que recibe por parámetro (**objeto JSON**), en el formulario principal, incluida la imagen (mostrarla en "imgImagen").

Permitirá modificar cualquier campo, a excepción del **id**, dejarlo como de sólo lectura.

Al pulsar el botón **Modificar postal** se invocará al método **ModificarPostal** que enviará la petición (por AJAX/FETCH) a **"http://localhost:2024/postal/id"**, que recibirán por PUT **postal_json** (direccion_destinatario, remitente y precio_estampilla, en formato de cadena JSON) e **imagen** (para modificar una postal en la base de datos).

Nota: El valor del id, será el id de la postal 'original', mientras que el resto de los valores serán los de la postal modificada. Dicho valor se pasará como parámetro de ruta.

Refrescar el listado sólo si se pudo modificar, caso contrario, informar (por alert y consola) de lo acontecido.

EliminarPostal. Recibe como parámetro al objeto JSON que se ha de eliminar. Pedir confirmación, mostrando la *dirección del destinatario* y el *remitente*, antes de eliminar.

Si se confirma se invocará (por AJAX/FETCH) a "***http://localhost:2024/postal/id***", donde id será el parámetro de ruta por DELETE. 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.