Segundo parcial de Laboratorio III

Aclaración:
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 pueden usar templates o reutilizar código hecho, pero en ningún caso, se debe incluir código obsoleto o que no cumpla ninguna función dentro del parcial.
Toda comunicación con el backend se realizará con AJAX. Todo pasaje de datos se hará con JSON.
Las vistas (páginas .php o .html) deben respetar fielmente las formas, colores, iconos, etc. Utilizar Bootstrap 4.
Se deben respetar los nombres de los archivos, de las clases, de los métodos y de los parámetros de las peticiones.
Respetar la estructura de directorios (index.php → ./public; fotos → ./src/fotos; clases en ./src/poo; vistas en ./src/views;).
La Api Rest que se utilizará en el parcial la proveerá el alumno. Se sugiere utilizar como base, la ya realizada en el segundo parcial de programación III -> APi Rest (Slim 4) para la concesionaria de autos Scaloneta, que interactúe con la clase Auto, la clase Usuario y la base de datos concesionaria_bd (autos - usuarios).
Agregar a la Api los siguientes verbos GET (a nivel de ruta): /front-end-login, /front-end-registro y /front-end-principal, para acceder a login.html, registro.html y principal.php respectivamente.

PARTE 1 (hasta un 5)

login.html – login.ts

Asociar al evento click del botón ***btnEnviar*** una función que recupere el correo y la clave para luego invocar al verbo POST de la ruta **/login** (de la Api Rest).

(POST) Se envía un JSON → user (correo y clave) y retorna un JSON (éxito: true/false; usuario: JSON (con todos los datos del usuario, a excepción de la clave) / null; status: 200/403)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - danger) un mensaje que indique lo acontecido.
Si la respuesta posee un 'status' 409, también mostrar el mensaje de lo acontecido (en un alert de BOOTSTRAP - danger).
Si es true, se guardará en el LocalStorage el usuario obtenido y se redireccionará hacia ***principal.php***.

El botón 'Quiero Registrarme!' llevará al usuario hacia la página ***registro.html***.

Colores e iconos: lightpink; lightgrey – fas fa-envelope; fas fa-key;

registro.html – registro.ts

Se registrarán los datos de un nuevo usuario.
Asociar al evento click del botón ***btnRegistrar*** una función que recupere el correo, la clave, el nombre, el apellido, el perfil y la foto. Invocar al verbo POST de la ruta **/usuarios** (de la Api Rest).

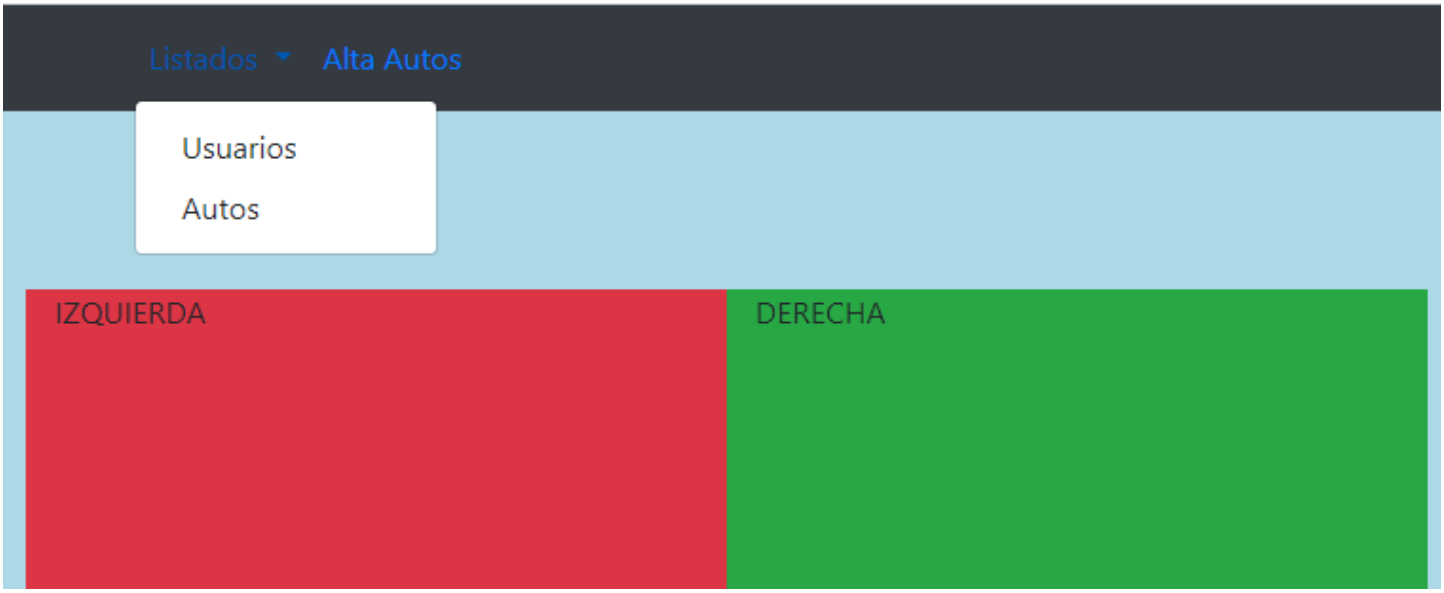
*(POST) Alta de usuarios. Se agregará un nuevo registro en la tabla usuarios *.*
*Se envía un JSON → usuario (correo, clave, nombre, apellido, perfil**) y foto.*
** ID auto-incremental. ** propietario, encargado y empleado.*
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - danger) un mensaje que indique lo acontecido.
Si es true, se redirigirá hacia ***login.html***.

Colores e iconos: rgb(153, 167,184); lightgrey – fas fa-envelope; fas fa-key; fas fa-user; far fa-id-card; fas fa-camera;

principal.php – principal.ts

Debe tener un menú (cómo muestra la imagen) y tener dividido el cuerpo en dos secciones (izquierda y derecha).



Al pulsar la opción de menú *Alta Autos*, se mostrará el formulario de alta de auto (cómo muestra la imagen) en la sección izquierda.

Formulario alta / modificación

Colores e iconos: darkcyan – fas fa-trademark; fas fa-palette; fas fa-car; fas fa-dollar-sign;

Asociar al evento click del botón ***btnAgregar*** una función que recupere el color, la marca, el precio y el modelo. Invocar al verbo POST (nivel de ruta **/usuarios** de la Api Rest).

*(POST) Alta de autos. Se agregará un nuevo registro en la tabla autos *.*
Se envía un JSON → auto (color, marca, precio y modelo).
** ID auto-incremental.*
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - danger) el mensaje recibido.
Si es true, se mostrará un mensaje (alert de BOOTSTRAP – success) indicando lo sucedido.

Al pulsar el submenú *Usuarios* del menú Listados, se mostrará el listado de los usuarios en una tabla de BOOTSRTAP (elegir un estilo). Las fotos tendrán un ancho y un largo de 50px.
Invocar al verbo GET (nivel de aplicación, de la Api Rest).

(GET) Listado de usuarios. Obtendrá el listado completo de los usuarios (array JSON).
Retorna un JSON (éxito: true/false; mensaje: string; dato: arrayJSON; status: 200/424)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - danger) el mensaje recibido.
Si es true, se armará (en el frontend) el listado que proviene del atributo **dato** del json de retorno. Mostrarlo en la sección derecha.

CORREO	NOMBRE	APELLIDO	PERFIL	FOTO
a@a.com	a	a	propietario	
b@b.com	b	b	empleado	
c@c.com	c	c	supervisor	
d@d.com	d	d	empleado	
e@e.com	e	e	empleado	

Al pulsar el submenú *Autos* del menú Listados, se mostrará el listado de los autos en una tabla de BOOTSRTAP (elegir otro estilo).
Invocar al verbo GET (ruta **/autos**, de la Api Rest).

(GET) Listado de autos. Obtendrá el listado completo de los autos (array JSON).
Retorna un JSON (éxito: true/false; mensaje: string; dato: arrayJSON; status: 200/424)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - danger) el mensaje recibido.
Si es true, se armará (en el frontend) el listado que proviene del atributo **dato** del json de retorno. Mostrarlo en la sección izquierda.

MARCA	COLOR	MODELO	PRECIO
citroen	blanco	c4	123456
renault	gris	r9	50500
ford	rojo	fiesta	95300

PARTE 2 (hasta un 6)

Agregar al listado de autos, dos columnas extras, cada una con un botón.
El primero con un botón (btn-danger) que permitirá el borrado del auto, previa confirmación del usuario, preguntando si el auto con tal marca, color y modelo se quiere borrar de la base de datos.
Si se confirma, se eliminará el auto, invocando al grupo **/cars** con verbo DELETE (de la Api Rest).

(DELETE) Borrado de autos por ID.
Recibe el ID del auto a ser borrado (id_auto, cómo parámetro de ruta).
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - warning) el mensaje recibido.
Si es true, refrescar el listado de autos.

El segundo con botón (btn-info) que permitirá la modificación del auto seleccionado. Para ello, se cargarán todos los datos del auto en el formulario (formulario alta / modificación). Mostrarlo en la sección izquierda.
Al pulsar el botón ***btnModificar*** (cambiarlo en el formulario) se invocará al grupo **/cars** con verbo PUT (de la Api Rest).

(PUT) Modificar los autos por ID.
Recibe el JSON del auto a ser modificado → auto (id_auto, color, marca, precio y modelo)
cómo parámetro a nivel de ruta.
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - warning) el mensaje recibido.
Si es true, refrescar el listado de autos.

PARTE 3

Agregar al listado de usuarios, una columna extra con un botón (btn-danger) que permitirá el borrado del usuario, previa confirmación del usuario, preguntando si el usuario con tal nombre, apellido y perfil se quiere borrar de la base de datos.
Si se confirma, se eliminará el usuario, invocando al grupo **/users** ruta **/delete** con verbo POST (de la Api Rest).

(POST) Borrado de usuarios por ID.
Recibe el ID del usuario a ser borrado en un JSON → usuario (id_usuario) pasado cómo parámetro de la petición (NO cómo parámetro de ruta)
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - warning) el mensaje recibido.
Si es true, refrescar el listado de usuarios.

El segundo con botón (btn-info) que permitirá la modificación del auto seleccionado. Para ello, se cargarán todos los datos del auto en el formulario (formulario alta / modificación). Mostrarlo en la sección izquierda.
Al pulsar el botón ***btnModificar*** (cambiarlo en el formulario) se invocará al grupo **/users** ruta **/edit** con verbo POST (de la Api Rest).

(POST) Modificación de usuarios.
Se envía un JSON → usuario (id_usuario, correo, clave, nombre, apellido, perfil) y foto.
Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Si el atributo éxito del json de retorno es false, se mostrará (en un alert de BOOTSTRAP - warning) el mensaje recibido.
Si es true, refrescar el listado de usuarios.

PARTE 4

Utilizando los métodos MAP, REDUCE y FILTER, se pide:

Agregar una opción de menú que muestre el listado completo de los autos cuyo precio sea mayor a 199999 y el color distinto a 'rojo'.
Invocar al verbo GET (ruta **/autos**, de la Api Rest).
En el frontend, filtrar el listado obtenido, de acuerdo al precio indicado. Mostrar el listado en la sección derecha.

Agregar una opción de menú que muestre el promedio de precios de aquellos autos cuya marca comience con la letra 'F'.
Invocar al verbo GET (ruta **/autos**, de la Api Rest).
Mostrar el promedio obtenido en un alert de BOOTSTRAP – info.

Agregar una opción de menú que muestre el listado de usuarios sólo con el nombre y el apellido, de todos aquellos usuarios que sean ***empleados*** *o* ***supervisores***.
Invocar al verbo GET (nivel de aplicación, de la Api Rest).
Mostrar en la sección izquierda.