

Recuperatorio 2 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.

POO: obligatorio.

Toda comunicación con el backend se realizará con AJAX.

Todo pasaje de datos se hará con JSON.

La Api Rest que se utilizará en el parcial la proveerá el alumno.

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) /login

Se envía un JSON → **user** (correo y clave) y retorna un JSON (éxito: true/false; jwt: JWT (con todos los datos del usuario) / 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 es true, se guardará en el LocalStorage el JWT obtenido y se redireccionará hacia **principal.php**.

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

A login form UI mockup with a pink background. At the top, the word "LOGIN" is written in blue. Below it, there are two input fields: "Correo" with an envelope icon and "Clave" with a key icon. Below the "Clave" field are two buttons: "Enviar" in blue and "Limpiar" in yellow. At the bottom, there is a green button that says "Quiero registrarme!".

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.

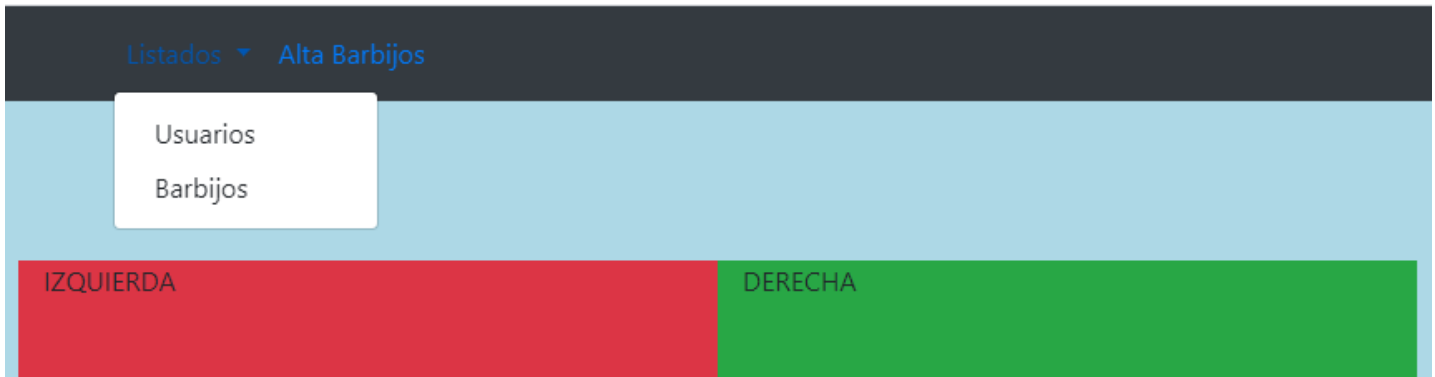
Si es true, se redirigirá hacia **login.html**.

The image shows a web form titled "REGISTRO" in a large, bold, blue font. The form is contained within a light grey box with a blue border. It features several input fields, each with a corresponding icon in a small square to its left: an envelope icon for "Correo", a key icon for "Clave", a person icon for "Nombre", another person icon for "Apellido", a document icon for a dropdown menu labeled "Seleccionar perfil", and a camera icon for a file upload area. The file upload area includes a button labeled "Seleccionar archivo" and a text field containing "No se eligió archivo". At the bottom of the form are two buttons: a green "Registrar" button and a yellow "Limpiar" button.

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 el submenú *Usuarios* del menú Listados, se mostrará el listado de los usuarios en una tabla de BOOTSTRAP (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. Mostrará el listado completo de los usuarios (array JSON).
Retorna un JSON (éxito: true/false; mensaje: string; tabla: stringJSON; 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 tabla 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	

NOTA: en cada interacción, se debe verificar la autenticidad del usuario, para ello, se tiene que invocar al verbo GET (ruta **/login**, de la ApiRest).

*(GET) Se envía el JWT → **token** (en el header) y se verifica. En caso exitoso, retorna un JSON (éxito: true/false; mensaje: string; status: 200/403).*

*Si el **JWT** no es válido, redirigir al **login.html**.*

Al pulsar el submenú *Barbijos* del menú Listados, se mostrará el listado de los barbijos en una tabla de BOOTSTRAP (elegir otro estilo).

Invocar al verbo GET (ruta **/barbijos**, de la Api Rest).

(GET) Listado de barbijos. Mostrará el listado completo de los barbijos (array JSON).

Retorna un JSON (éxito: true/false; mensaje: string; tabla: stringJSON; 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 tabla del json de retorno. Mostrarlo en la sección izquierda.

COLOR	TIPO	PRECIO
blanco	transparente	1230
gris	liso	505
rojo	estampado	953

PARTE 2 (hasta un 6)

Agregar al listado de barbijos, dos columnas extras, cada una con un botón.

El primer botón (btn-danger), permitirá el borrado del barbijos, previa confirmación del usuario.

Si se confirma, se eliminará el barbijos, invocando al verbo DELETE (de la Api Rest).

(DELETE) Borrado de barbijos por ID.

*Recibe el ID del barbijos a ser borrado → **id_barbijos** más el JWT → **token** (en el header).*

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 barbijos.

El segundo botón (btn-info), permitirá la modificación del barbijos seleccionado. Para ello, se cargarán todos los datos del barbijos en el formulario (cómo muestra la imagen, mostrarlo en la sección derecha).

Asociar al evento click del botón **btnModificar** una función que recupere el color, el tipo y el precio e invoque al verbo PUT (de la Api Rest).

(PUT) Modifica barbijos por ID.

Recibe un JSON → **barbijo** (formado con todos los datos del barbijo más el ID) más el JWT → **token** (en el header).

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 barbijos.

NOTA: en cada interacción, se debe verificar la autenticidad del usuario, para ello, se tiene que invocar al verbo GET (ruta **/login**, de la ApiRest).

(GET) Se envía el JWT → **token** (en el header) y se verifica. En caso exitoso, retorna un JSON (éxito: true/false; mensaje: string; status: 200/403).

Si el **JWT** no es válido, redirigir al **login.html**.

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

PARTE 3

Al pulsar la opción de menú **Alta Barbijos**, se mostrará el formulario de alta de barbijo (modificar lo necesario del formulario utilizado para la modificación).

Al pulsar el botón, se debe invocar al verbo POST (nivel de aplicación de la Api Rest).

(POST) Alta de barbijos. Se agregará un nuevo registro en la tabla barbijos *.

Se envía un JSON → **barbijo** (color, tipo ** y precio).

* ID auto-incremental. ** liso, estampado y transparente.

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.
Refrescar el listado de barbijos.

NOTA: en cada interacción, se debe verificar la autenticidad del usuario, para ello, se tiene que invocar al verbo GET (ruta **/login**, de la ApiRest).

(GET) Se envía el JWT → **token** (en el header) y se verifica. En caso exitoso, retorna un JSON (éxito: true/false; mensaje: string; status: 200/403).

Si el **JWT** no es válido, redirigir al **login.html**.