

MODELO SEGUNDO PARCIAL – LABORATORIO III

Aclaración:

Toda comunicación con el backend se realizará con AJAX. Todo el pasaje de datos se hará con JSON. Las vistas (páginas .php o .html) deben respetar fielmente las formas, colores, iconos, etc..

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

La Api Rest se creará en Node.js. La misma tendrá que interactuar con la base de datos productos_usuarios_node(productos – usuarios) y gestionar JWT.

Parte 1 BACKEND – Node.js (hasta un 4)

Diseñar un servidor Node.js (**http://localhost:4321**) con las siguientes rutas y middlewares:

verificar_usuario (MW): Recibe el legajo y el apellido en un JSON y retorna toda la información del registro al siguiente callable. Si el usuario no existe, retornará un JSON (éxito: false; mensaje: string; jwt: null; status: 401).

login (POST): Se envía el legajo y el apellido en un JSON y creará un JWT retornando otro JSON (éxito: true/false; mensaje: string; jwt: JWT (*) / null; status: 200/401).

Agregar el middleware anterior.

(*) el payload del JWT tendrá la siguiente estructura:

- usuario: con todos los datos del usuario, a excepción de la clave.
- api: nombre de la base de datos.

El token debe expirar en 5 minutos.

verificar_jwt (MW): Recibe el JWT (en el **Bearer**) y retorna toda la información del JWT decodificada al siguiente callable. Si el JWT no es válido, retornará un JSON (éxito: false; mensaje: string; status: 403).

verificar_token (GET): Se envía el JWT (en el **Bearer**) y retorna un JSON (éxito: true/false; mensaje: string; payload: datos (*) / null; status: 200/403).

Agregar el middleware anterior.

(*) datos completos del payload

Parte 1 FRONTEND – HTML5 y TypeScript (hasta un 5)

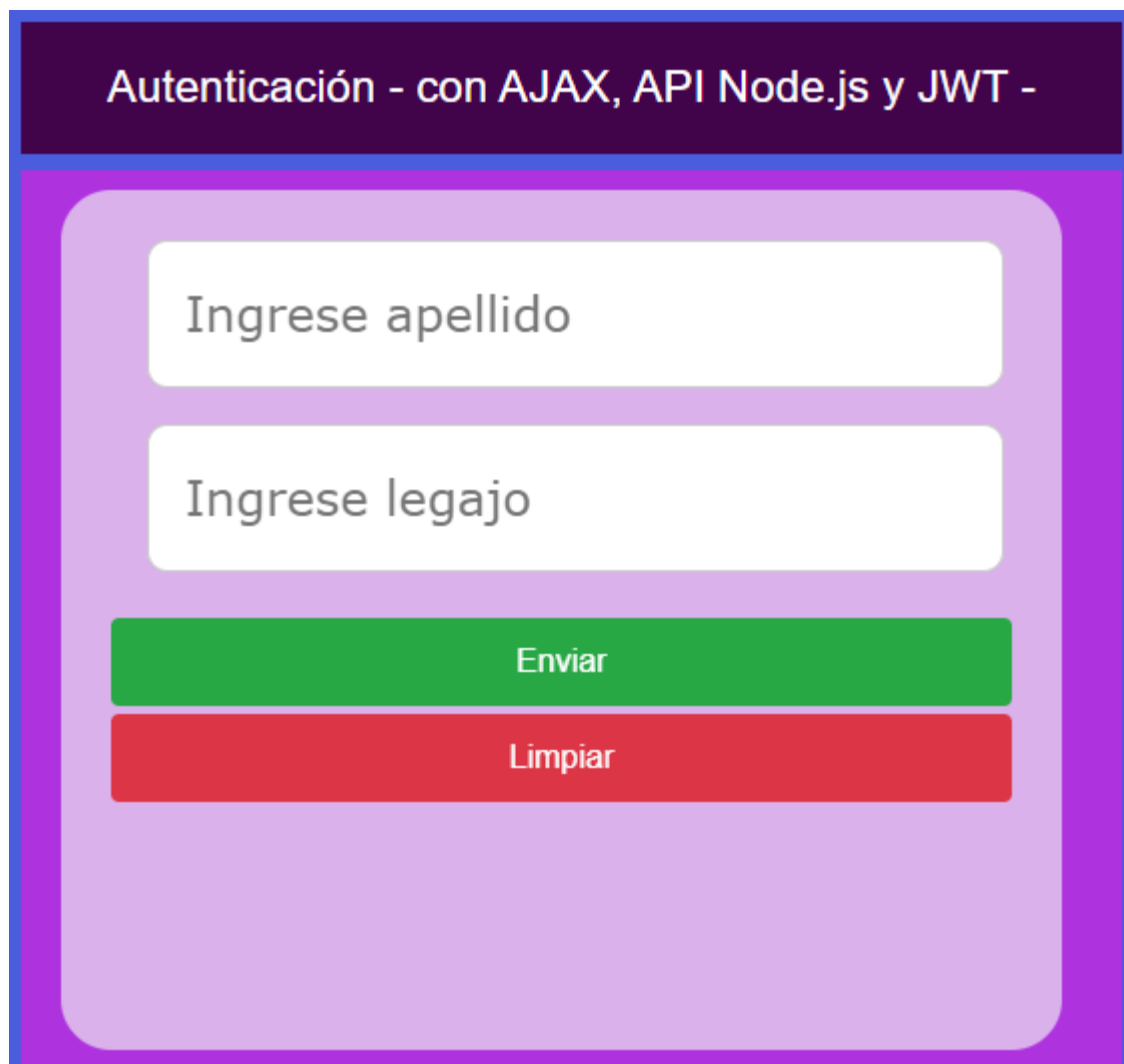
login.html – login.ts

Desarrollar un proyecto, que desde **http://localhost/lab_3/modelo_sp**, se acceda a las siguientes páginas que interactúen con el servidor Node.js.

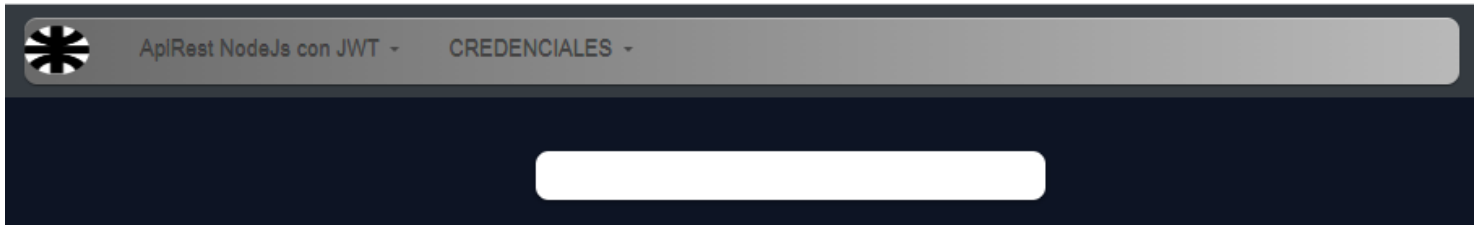
Asociar 'dinámicamente' al evento click del botón **btnEnviar**, de la página login.html, una función que recupere el legajo y el apellido para luego invocar (por **AJAX**) hacia al verbo POST de la ruta **/login** (de la Api Rest, "**http://localhost:4321/login**").

Si el atributo éxito del json de retorno es false, se mostrará un mensaje que indique lo acontecido (por consola y alert).

Si es true, se guardará en el LocalStorage el **JWT** obtenido y se redireccionará hacia **principal.php**.



The image shows a login form interface. At the top, there is a dark purple header bar with the text "Autenticación - con AJAX, API Node.js y JWT -" in white. Below the header, the form is contained within a light purple rounded rectangle. Inside this rectangle, there are two white input fields with rounded corners. The first input field is labeled "Apellido" and the second is labeled "Legajo". Below these input fields, there are two buttons: a green button labeled "Enviar" and a red button labeled "Limpiar". The entire form is set against a background of a purple-to-blue gradient.

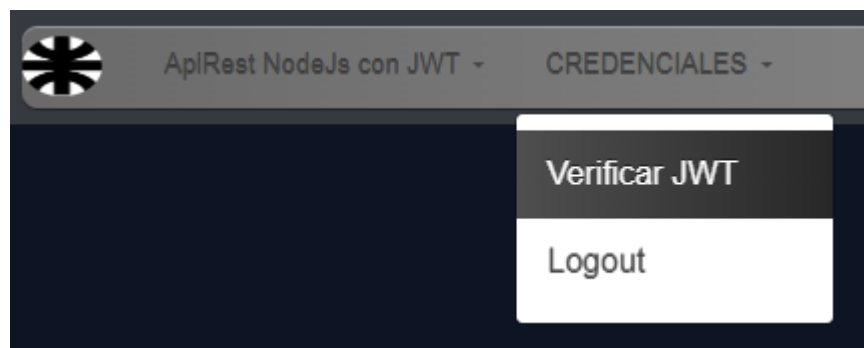


Principal.html debe tener un menú (cómo muestra la imagen) y poseer un `<div>` en el cuerpo.

Al pulsar el submenú Verificar JWT del menú Credenciales, se mostrará el `<div>` la información obtenida del servidor.

Invocar (por **AJAX**) al verbo GET (de la ruta **/verificar_token**, de la Api Rest). Si el atributo éxito del json de retorno es false, se mostrará el mensaje recibido (por consola y alert)

Nota: Si el JWT no es válido, redirigir hacia index.html.



Al pulsar el submenú Logout del menú Credenciales, se redirigirá hacia index.html (vaciar el LocalStorage)

Parte 2 BACKEND – Node.js (hasta un 6)

Agregar las siguientes rutas a la Api:

productos_bd (GET): mostrará el listado completo de los productos (obtenidos de la base de datos) en un array de Producto (en formato de cadena JSON).

Retorna un JSON (éxito: true/false; mensaje: string; dato: arrayJSON / null; status: 200/424)

Agregar el middleware **verificar_jwt**

productos_bd (POST): recibirá un JSON → **obj** (código, marca y precio) y **foto** para agregar un registro en la tabla **productos**, de la base de datos productos_usuarios_node.

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

Agregar el middleware **verificar_jwt**

Nota: La foto guardarla en **./public/fotos/**, con el nombre formado por código punto extensión. Ejemplo: 912.jpg

productos_bd/eliminar (POST): Borrado de productos por código.

Recibe el código del producto a ser borrado en un JSON más el JWT (en el **Bearer**).

Agregar el middleware **verificar_jwt**

Nota: La foto debe ser borrada.

Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

productos_bd/modificar (POST): Modificar productos.

Recibe el JSON del producto a ser modificado → **obj** (codigo, marca, precio), la **foto** y el JWT (en el **Bearer**).

El código será el del producto a ser modificado, mientras que el resto, serán los valores a ser modificados.

Agregar el middleware **verificar_jwt**

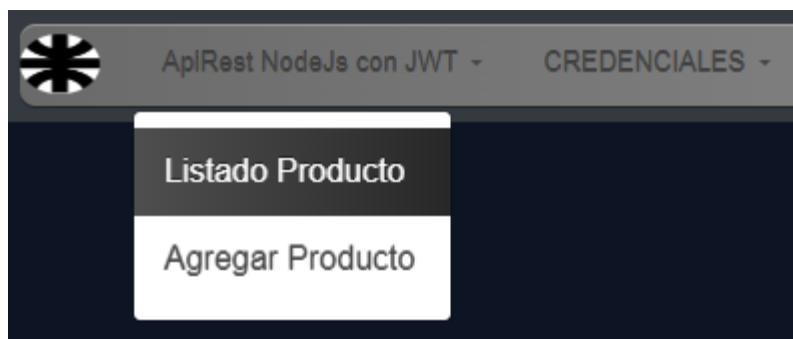
Nota: La foto guardarla en **./public/fotos/**, con el siguiente formato:

codigo.extension.

Ejemplo: **./public/fotos/91218.jpg**

Retorna un JSON (éxito: true/false; mensaje: string; status: 200/418)

Parte 2 FRONTEND – HTML5 y TypeScript (hasta un 7)



Al pulsar el submenú Listado Producto del menú ApiRest NodeJs con JWT, se mostrará el listado de los productos en una tabla de HTML.

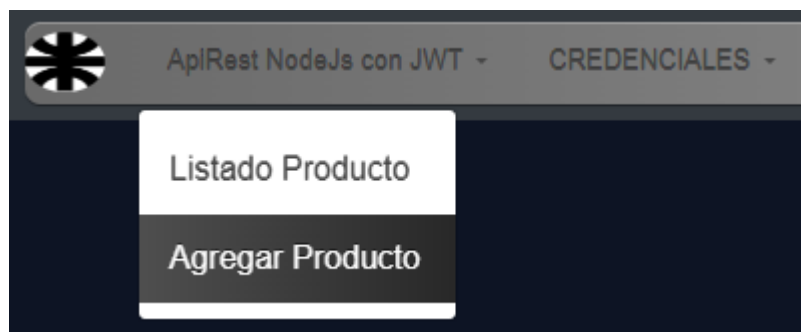
Invocar (por **AJAX**) al verbo GET (ruta **/productos_bd**, de la Api Rest). Si el atributo éxito del json de retorno es false, se mostrará (en un alert y en consola) el mensaje recibido.

Si es true, se armará (en el frontend) el listado que proviene del json de retorno. Mostrarlo en **<div>**. Las fotos tendrán un ancho y alto de 50px.

Nota: Si el JWT no es válido, redirigir hacia login.html.

CÓDIGO	MARCA	PRECIO	FOTO	ACCIONES
1	marca_uno_con_foto	100		 
222	nueva marca	500		 

Parte 3 FRONTEND – HTML5 y TypeScript (hasta un 8)



Al pulsar la opción de menú Agregar Producto, se mostrará el formulario de alta de productos (cómo muestra la imagen).

Formulario alta / modificación

AGREGAR PRODUCTO

Código:

Título:

OK.

Precio:

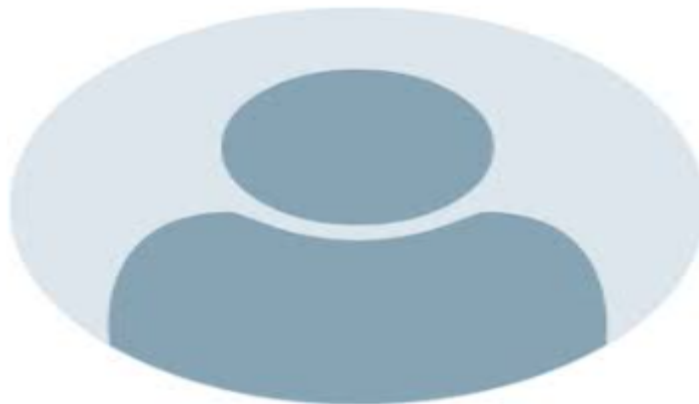


OK.

Foto:

 91218.webp

OK.



Cerrar

Aceptar

Asociar al evento click del botón **btnAgregar** una función que recupere el código, la marca, el precio y la foto. Invocar (por **AJAX**) al verbo POST (de la ruta **/productos_bd** de la Api Rest). Se mostrará (en un alert y por consola) el mensaje recibido.

Nota: Si el JWT no es válido, redirigir hacia login.html.

Parte 4 FRONTEND – HTML5 y TypeScript (hasta un 10)

Agregar al listado de productos, una columna extra, con dos botones.

El primer botón (btnUpdate) permitirá la modificación del producto seleccionado. Para ello, se cargarán todos los datos del producto en el formulario (formulario alta / modificación).

Al pulsar el botón **btnModificar** (cambiarlo en el formulario) se invocará (por **AJAX**) al verbo POST (de la ruta **/productos_bd/modificar** de la Api Rest).

Si el atributo éxito del json de retorno es false, se mostrará (en un alert y por consola) el mensaje recibido.

Si es true, refrescar el listado de juguetes.

Nota: Si el JWT no es válido, redirigir hacia login.html.

El segundo botón (btnDelete) permitirá el borrado del juguete, previa confirmación del usuario, preguntando si el juguete con tal marca y precio se quiere borrar de la base de datos. Si se confirma, se eliminará el juguete, invocando (por **AJAX**) al verbo POST (de la ruta **/productos_bd/eliminar** de la Api Rest).

Si el atributo éxito del json de retorno es false, se mostrará (en un alert y por consola) el mensaje recibido.

Si es true, refrescar el listado de juguetes.

Nota: Si el JWT no es válido, redirigir hacia login.html.