Trabajo Práctico Final

Laboratorio 3 - TUP - Profesores: Castillo - Barbero

Introducción

En este trabajo final se validarán los conocimientos de HTML, CSS y JavaScript en el maquetado y consumo de API, especialmente orientada a un ABM (Alta, Baja y Modificación) de productos. Se utilizarán los verbos HTTP GET, POST, PATCH y DELETE para interactuar con los recursos de productos.
La modalidad es individual.

Requerimientos funcionales

Para aprobar el trabajo, deberán satisfacerse todos los requerimientos funcionales sin excepción. Una vez satisfechos, queda a libertad de cada alumno el agregar requerimientos adicionales.

- **Sitio web:** debe contener al menos 2 páginas.
- **Maqueta HTML completa:** debe contener encabezado, contenido y pie.
- **Utilización de API:** ABM completo (Alta-Baja-Modificación)
    - Tabla de productos.
    - Formulario de alta.
    - Formulario de edición.
    - Consultar antes de borrar.
    - Filtro de la tabla (opcional).
- **Tecnologías a utilizar:**
    - HTML
    - CSS (desktop y mobile)
    - JavaScript
    - VUE (opcional)
- **Entrega:** proyecto en un repositorio público de GitHub.
- **Fecha límite de entrega:** 1 Julio de 2024.

Endpoints y métodos HTTP

Todos los endpoints devuelven un string “OK” en caso de éxito, de lo contrario, devolverán una explicación breve sobre errores en las propiedades enviadas. En caso de estar enviándole todas las propiedades correctamente, pueden existir errores de conexión o base de datos donde la API devolverá “KO”.

URL base

Dentro de la URL, tendremos como primer parámetro el número de legajo del alumno. De esta forma, cada dato que interactúe con la API, será registrado bajo el legajo de cada alumno pudiendo guardar, editar y borrar de manera independiente. En la URL deberán cambiar el texto “<legajo>” por su legajo de alumno de 5 números.

URL base de la API:

- https://api.yumserver.com/<legajo>/products

Obtener todos los productos

```
GET /<legajo>/products
```

Este endpoint devuelve todos los productos disponibles. Devuelve una colección de objetos JSON con las siguientes propiedades: idcod, titulo, precioPeso, precioDolar, fecha.

Crear un nuevo producto

```
POST /<legajo>/products
```

Este endpoint crea un nuevo producto. Debe recibir un objeto JSON con las siguientes propiedades: titulo, precioPeso, precioDolar, fecha. El servidor automáticamente completará el campo idcod.

Modificar un producto existente

```
PATCH /<legajo>/products
```

Este endpoint modifica un producto existente. Debe recibir un objeto JSON con las siguientes propiedades: idcod, titulo, precioPeso, precioDolar, fecha.

Eliminar un producto

```
DELETE /<legajo>/products
```

Este endpoint elimina un producto existente identificado por su idcod.

Parámetros de solicitud

Al interactuar con la API, debemos enviar un objeto JSON con las siguientes propiedades dependiendo la acción que debamos realizar.

Crear un nuevo producto

- **titulo:** Título del producto (cadena de texto).
- **precioPeso:** Precio del producto en pesos (número).
- **precioDolar:** Precio del producto en dólares (número).
- **fecha:** Fecha del producto (cadena de texto en formato YYYY-MM-DD).

Modificar un producto existente

- **idcod:** ID del producto a modificar (cadena de texto).
- **titulo:** Nuevo o mismo título del producto (cadena de texto).
- **precioPeso:** Nuevo o mismo precio del producto en pesos (número).
- **precioDolar:** Nuevo o mismo precio del producto en dólares (número).
- f**echa:** Nueva o misma fecha del producto (cadena de texto en formato YYYY-MM-DD).

Eliminar un producto

- **idcod:** ID del producto a eliminar (cadena de texto).

Ejemplos de solicitud y respuesta

Obtener todos los productos

Solicitud:

Se debe reemplazar *<legajo>* por su número de legajo de estudiante.

```
GET https://api.yumserver.com/<legajo>/products
```

Respuesta:

```
[
  {
    "idcod": "1",
    "titulo": "Producto 1",
    "precioPeso": 1000,
    "precioDolar": 1,
    "fecha": "2024-05-13"
  },
  {
    "idcod": "2",
    "titulo": "Producto 2",
    "precioPeso": 2000,
    "precioDolar": 2,
    "fecha": "2024-05-14"
  }
]
```

Crear un nuevo producto

Solicitud:

Se debe reemplazar *<legajo>* por su número de legajo de estudiante.

```
POST https://api.yumserver.com/<legajo>/products
Body:
{
  "titulo": "Nuevo Producto",
  "precioPeso": 150,
  "precioDolar": 15,
  "fecha": "2024-05-15"
}
```

Respuesta:

En caso de éxito, devolverá un string “OK”.

```
OK
```

En caso de error, devolverá un string con información sobre el error, por ejemplo:

```
El título no puede estar vacío.
```

Modificar un producto existente

Solicitud:

Se debe reemplazar *<legajo>* por su número de legajo de estudiante.

```
PATCH https://api.yumserver.com/<legajo>/products/
Body:
{
  "idcod": "b23bUIAC32Aw",
  "titulo": "Producto Modificado",
  "precioPeso": 150,
  "precioDolar": 15,
  "fecha": "2024-05-15"
}
```

Respuesta:

En caso de éxito, devolverá un string “OK”.

```
OK
```

En caso de error, devolverá un string con información sobre el error, por ejemplo:

```
El título no puede estar vacío.
```

Eliminar un producto

Solicitud:

Se debe reemplazar *<legajo>* por su número de legajo de estudiante.

```
DELETE https://api.yumserver.com/<legajo>/products
Body:
{
  "idcod": "b23bUIAC32Aw"
}
```

Respuesta:

En caso de éxito, devolverá un string “OK”.

```
OK
```

En caso de error, devolverá un string con información sobre el error, por ejemplo:

```
El idcod no es válido.
```

Códigos de ejemplo

Se debe reemplazar *<legajo>* por su número de legajo de estudiante.

```
// Obtener todos los productos
fetch('https://api.yumserver.com/<legajo>/products')
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));

// Crear un nuevo producto
fetch('https://api.yumserver.com/<legajo>/products', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
      titulo: 'Nuevo Producto',
      precioPeso: 15000,
      precioDolar: 15,
      fecha: '2024-05-15'
    })
  })
  .then(response => response.text())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```

Utilidades de JSON

Conversión de Objeto JSON a String

```
let miObjetoJSON = {
  idcod: "abc"
};

let cadenaDeTexto = JSON.stringify(miObjetoJSON);
```

Conversión de String a Objeto JSON

```
let cadenaDeTexto = "{idcod: \"abc\"}";

let miObjetoJSON = JSON.parse(cadenaDeTexto);
```

Entidades genéricas

Se pueden utilizar diferentes entidades (productos es una) para crear distintas utilidades y cualquier tipo de proyecto.

La URL genérica permite interactuar con nombres de entidades por ruta y con propiedades también genéricas. Las propiedades (variables a enviar y recibir) son 10 y están nombradas desde *param1* hasta *param10*, de esta forma pueden utilizarlas para cualquier tipo de estructura de dato.

```
BODY:

{
    param1: 'string',
    param2: 'string',
    param3: 'string',
    param4: 'string',
    param5: 'string',
    param6: 'string',
    param7: 'string',
    param8: 'string',
    param9: 'string',
    param10: 'string'
}
```

Demás está aclarar que pueden utilizarse la cantidad de parámetros a gusto.

URL Genérica:

```
https://api.yumserver.com/<legajo>/generic/<entidad>
```

Se debe reemplazar *<legajo>* por su número de legajo de estudiante y *<entidad>* por la que desee utilizar.

Ejemplo de interacción con una entidad ficticia Clientes

```
// Obtener todos los clientes
fetch('https://api.yumserver.com/00000/generic/clientes)
  .then(response => response.json())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));

// Crear un nuevo cliente
fetch('https://api.yumserver.com/00000/generic/clientes, {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({
      param1: 'Nuevo cliente,
      param2: 15000,
      param3: 15,
      param4: '2024-05-15'
    })
  })
  .then(response => response.text())
  .then(data => console.log(data))
  .catch(error => console.error('Error:', error));
```