«Talento Tech»

Desarrollo Web 4

Clase 03

Buenos Aires Ciudad

Buenos Aires aprende
Agencia de Habilidades para el Futuro

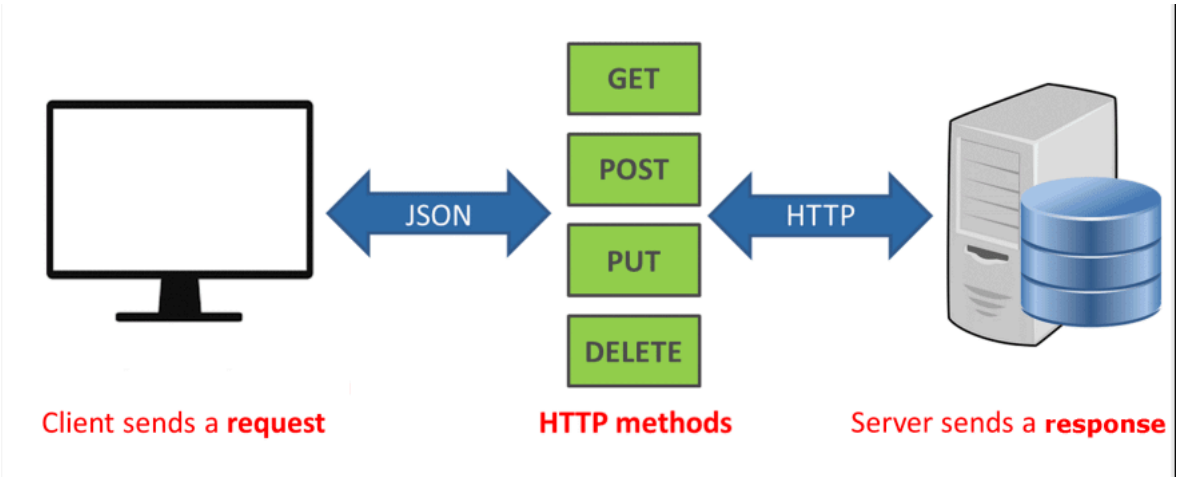
Clase N° 3 | Middlewares y Postman

Temario:

- Continuamos con Express- Middleware, POST, DELETE
- Postman: Testar nuevos métodos (POST y DELETE)

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Continuamos con Express - Middleware, POST, DELETE



Método POST y DELETE

¿Te acordás que la clase pasada estuvimos codeando una api usando express?
Bueno ahora vamos a continuar el proyecto implementando nuevos métodos. Ellos van a ser **POST** y **DELETE**.

El método **POST** se utiliza para enviar datos al servidor para ser procesados. Su función principal es solicitar que el servidor acepte y almacene los datos enviados en el cuerpo de la solicitud. Es comúnmente utilizado para crear nuevos recursos en el servidor.

El verbo **DELETE** se utiliza para eliminar un recurso identificado por una URL específica. Al enviar una solicitud **DELETE**, estás solicitando que el servidor elimine el recurso correspondiente.
Por ejemplo, puedes usar **DELETE** para eliminar un usuario de una base de datos o un archivo de un servidor.

Vamos a hacer un repaso para poder terminar nuestra **Rest API.**

⚠ *Recuerden tener instalado **Express y Body-Parser.** En el caso de no tenerlos instalados recuerden que había que implementar los comandos*

- Para poder inicializar el proyecto:

```
npm init-y
```

- Para poder inicializar el servidor:

```
npm i express body-parser
```

Hasta ahora teníamos un código similar a este:

```
const express = require('express');
const  bodyParser = require ('body-parser');

const  app = express();
app.use(bodyParser.json());

//Variable global que almacena los objetos
let datos = [
{ id:1, nombre: 'Ejemplo 1'},
{ id:2, nombre: 'Ejemplo 2'},
{ id:3, nombre: 'Ejemplo 3'},
];
//Ruta para tener todos los datos
app.get('/datos', (req, res) => {
            res.json(datos);
});

const puerto = 3000;
app.listen (puerto, () => {
      console.log(`Servidor Express escuchando en el puerto
${puerto}`);
});
```

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Aplicación del Método POST

Luego vamos a tener que crear el verbo **POST** el cual nos va a permitir poder agregar elementos a nuestra base de datos.

```
//Ruta para crear un nuevo dato
app.post ('/datos', (req, res) => {
    const nuevoDato = req.body;
              nuevoDato.id = datos.length + 1;
              datos.push(nuevoDato);
              res.json(nuevoDato);
    });
```

Ahora vamos a explicar un poco de que se trata:

Definición de la Ruta

```
app.post ('datos', …)
```

Configura una ruta en Express para manejar solicitudes HTTP con el método POST en la URL /datos. Esto significa que esta ruta está diseñada para recibir datos y crear un nuevo recurso en el servidor.

Manejo de la Solicitud POST:

```
(req, res) => {...}:
```

Este es el controlador de la ruta que se ejecuta cuando se recibe una solicitud POST en la ruta /datos. req representa la solicitud (request) y res representa la respuesta (response).

Obtención de los Datos del Cuerpo de la Solicitud:

```
const nuevoDato = req-body;
```

Se obtiene el cuerpo de la solicitud utilizando req.body. Este paso asume que el middleware adecuado, como body-parser, ha sido configurado para analizar los datos del cuerpo en formato JSON.

Asignación de un ID Único al Nuevo Dato:

```
nuevoDato.id= datos.length + 1;
```

Asigna un ID único al nuevo dato. En este caso, simplemente se asigna un ID incremental basado en la longitud actual del array datos. Es común utilizar bases de datos que generen IDs automáticamente, pero aquí se muestra un enfoque básico.

Inclusión del Nuevo Dato en el Array de Datos:

```
datos.push(nuevoDato);
```

Se agrega el nuevo dato al array datos. Este array es asumido como una variable global que almacena todos los datos.

Respuesta JSON al Cliente:

```
res.json(nuevoDato);
```

Se responde al cliente con el nuevo dato creado en formato JSON. Esta respuesta generalmente indica que la operación de creación fue exitosa y devuelve los detalles del nuevo dato creado, incluido el ID asignado.

Aplicación del Método DELETE

Ahora vamos a ver como aplicar el método DELETE

```
//Ruta para eliminar un dato por su ID
app.delete('/datos/:id', (req, res) => {
    const id = parseInt(req.params.id);
    datos = datos.filter((item) => item.id !== id);
    res.json ({ mensaje: 'Dato eliminado exitosamente' });
});
```

Ahora vamos a explicar un poco el código:

Ruta y Método HTTP:

```
app.delete('/datos/:id' , …)
```

Configura una ruta que responde a solicitudes **HTTP** con el método **DELETE** en la URL /datos/:id. Al igual que antes, :id es un parámetro de ruta que captura el valor del ID del recurso.

Parseo del ID:

```
const id = parseInt(req.params.id);
```

Extraemos el ID de la solicitud. req.params.id contiene el valor del parámetro :id de la URL, y lo convertimos a un número entero usando parseInt.

Filtrado y Actualización del Array:

```
datos = datos.filter ((item) => item.id !== id);
```

Filtramos el array datos, eliminando el objeto que tiene el mismo **ID** que el proporcionado en la solicitud. El método de arrays filter crea un nuevo array que contiene sólo los elementos que cumplen la condición.
Este código elimina el dato con el **ID** correspondiente del array.

Respuesta JSON:

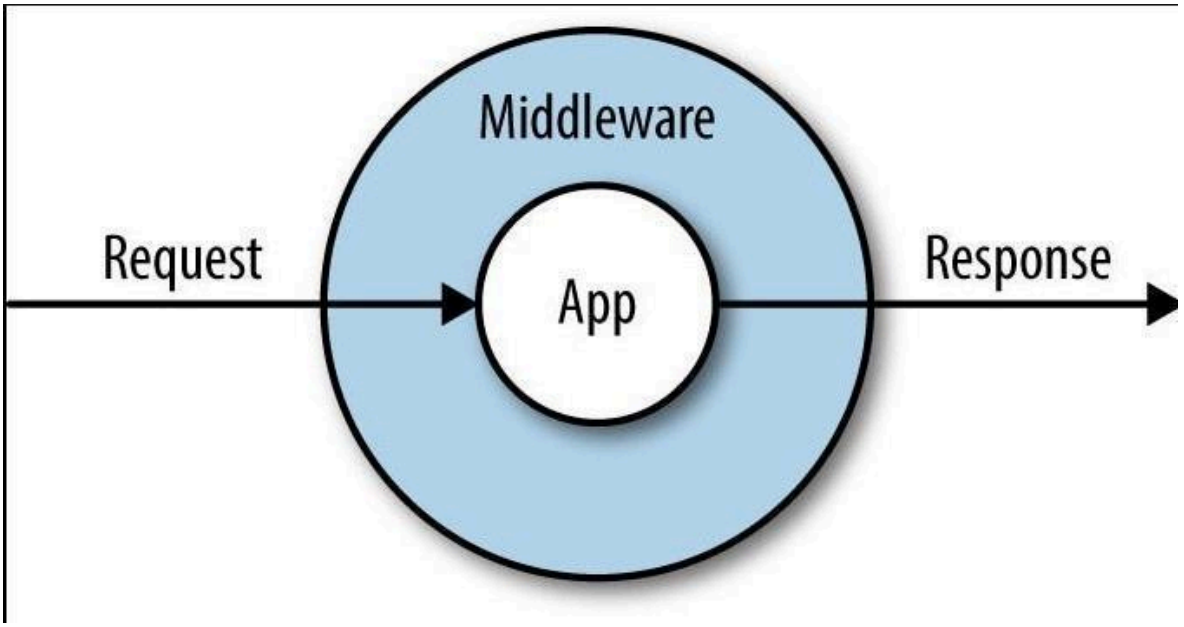
```
    res.json ({ mensaje: 'Dato eliminado exitosamente' });
```

Respondemos con un objeto **JSON** que contiene un mensaje indicando que el dato fue eliminado con éxito.

En resumen, esta ruta **DELETE** permite eliminar un dato del array datos según el ID proporcionado en la **URL**. El array se actualiza después de eliminar el dato, y se responde con un mensaje **JSON** indicando que la eliminación fue exitosa.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

¿Qué es un middleware?



En el código tuvimos que instalar body-parser e implementarlo pero un poco no entendíamos de qué se trataba.
Body-parser es un **middleware** y básicamente un middleware es un tipo de software que actúa como una capa intermedia entre distintos componentes de un sistema. Su función principal es facilitar la comunicación y la interacción entre estos componentes.

Podés pensar en un middleware como un intermediario que ayuda a gestionar y controlar el flujo de datos entre diferentes partes de una aplicación.

En el contexto de frameworks web como Express.js, los middlewares son funciones que se ejecutan durante el procesamiento de una solicitud HTTP. Por ejemplo, el middleware puede analizar el cuerpo de una solicitud, manejar la autenticación, o comprimir la respuesta antes de enviarla al cliente.

Postman: Testear nuevos métodos (POST y DELETE)

Paso 1: Iniciar el Servidor Express

Asegúrate de tener Node.js instalado. Luego, guarda el código del servidor en un archivo llamado app.js y ejecuta el siguiente comando en la terminal:

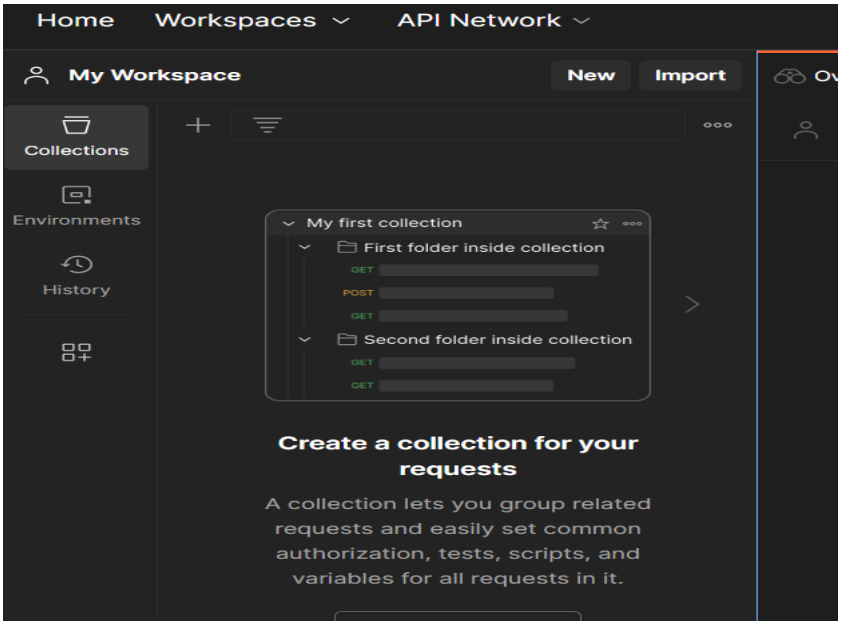
```
node app.js
```

Esto iniciará el servidor Express en el puerto 3000.

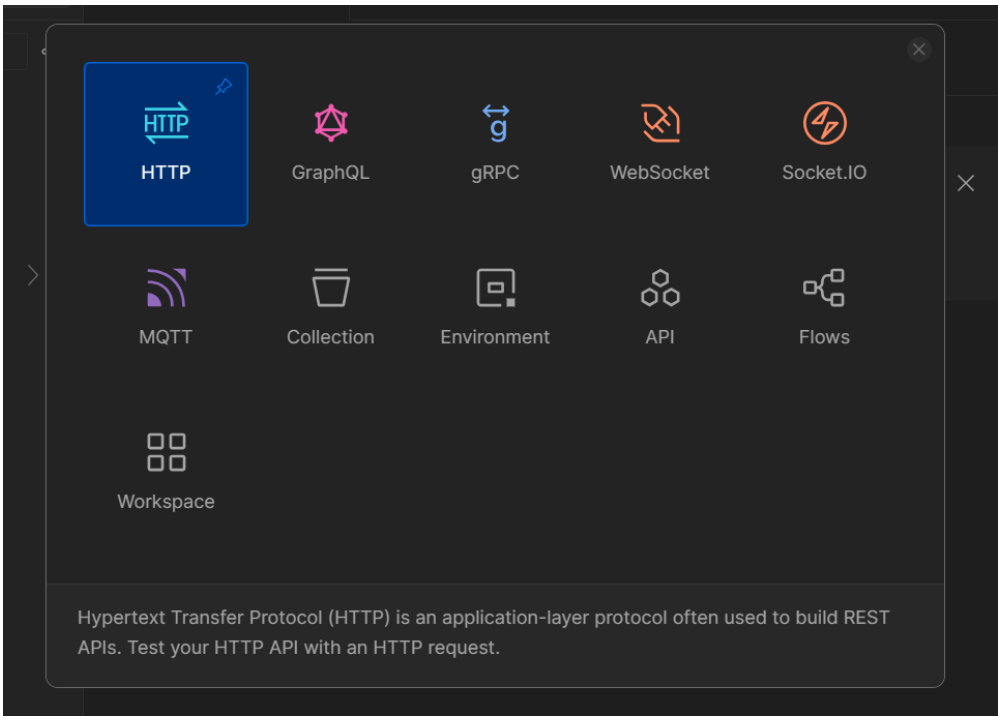
Paso 2: Abrir Postman

Abre Postman en su página oficial: https://www.postman.com/. Tendrás que logearte para ingresar e ir.

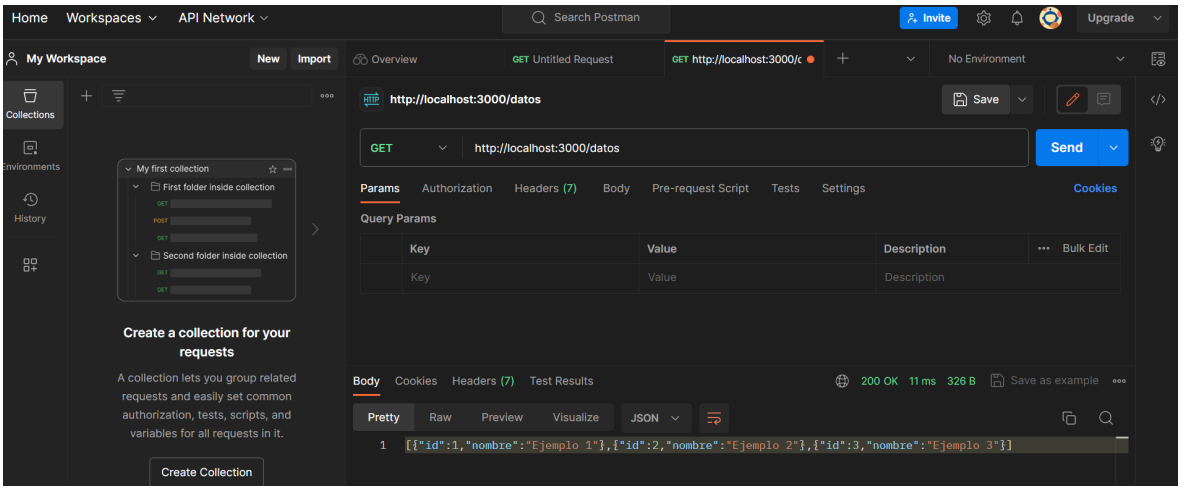
También deberás dirigirte al workspace que ya existe y generar una petición clickeando en 'New' y después en 'HTTP'. Tendrás que usar clickear en 'New' por cada uno de los metodos a usar.



Paso 3: Realizar Solicitudes HTTP



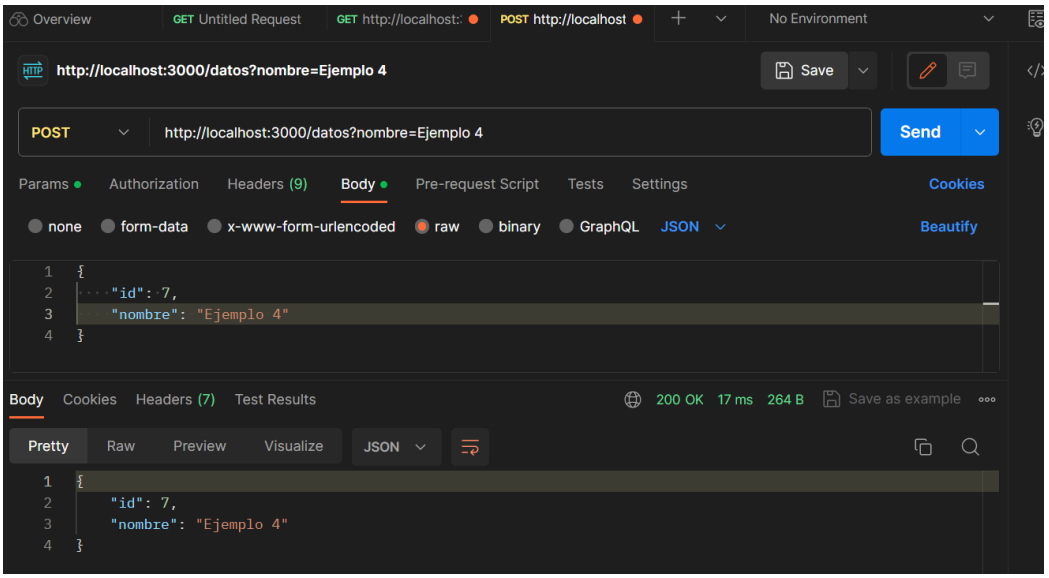
Obtener todos los datos (GET):



Buenos Aires aprende
Agencia de Habilidades para el Futuro

Método: GET
URL: http://localhost:3000/datos/1 (o cualquier otro ID existente)
Haz clic en "Send" para obtener un dato específico por su ID.

Crear un nuevo dato (POST):



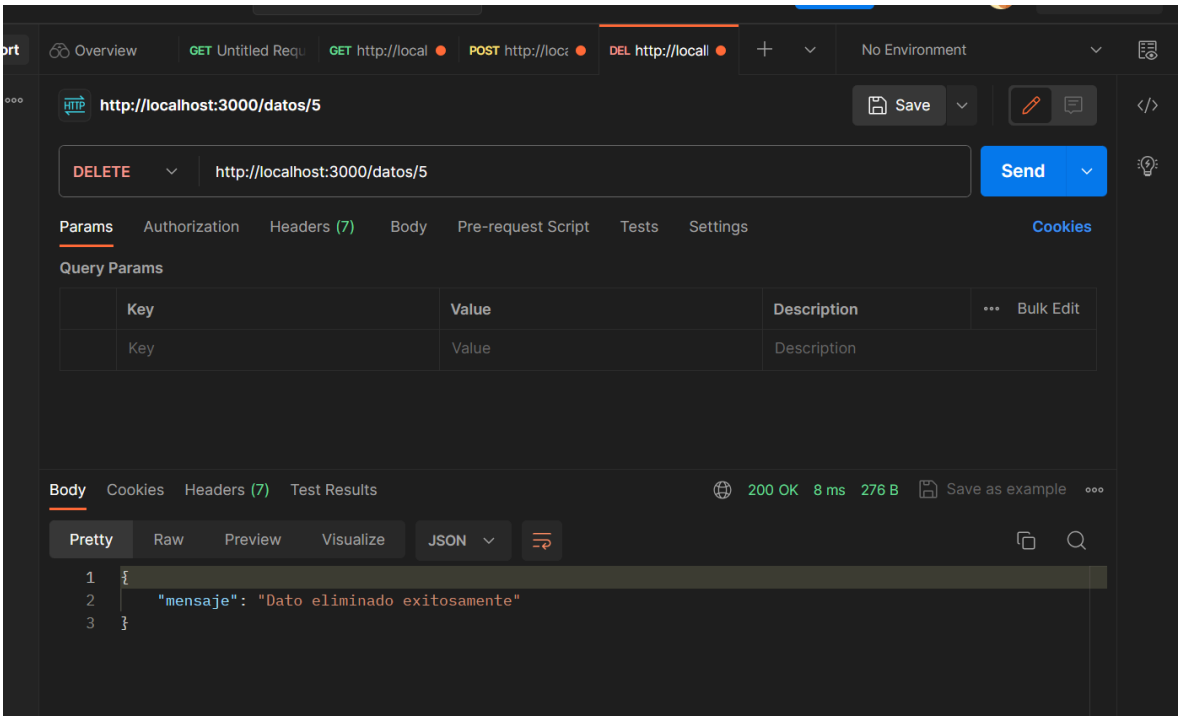
Método: POST
URL: http://localhost:3000/datos
Cuerpo (seleccione "raw" y "JSON (application/json)"):

```
{
“nombre”: “Nuevo Ejemplo”
}
```

Haz clic en "**Send**" para crear un nuevo dato.
Actualizar un dato por su ID (PUT):
Haz clic en "Send" para actualizar un dato por su ID.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Eliminar un dato por su ID (DELETE):



Método: DELETE
URL: http://localhost:3000/datos/1 (o cualquier otro ID existente)
Haz clic en "Send" para eliminar un dato por su ID.

Paso 4: Verificar Resultados
Asegúrate de que los alumnos observen los resultados de cada solicitud y entiendan cómo interactúa Postman con el servidor Express para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en la colección de datos.

Al utilizar este enfoque, ya podes crear y probar las solicitudes directamente desde el sitio web de Postman sin importar nada desde el exterior. Te permite familiarizarse con la interfaz web y realizar pruebas de API de manera rápida y eficiente.Este paso a paso los debería ayudar a probar todas las funcionalidades del servidor utilizando Postman.


Buenos Aires
aprende
Buenos
Aires
Ciudad