

Guía de ejercicios - Persistencia



¡Hola! Te damos la bienvenida a esta nueva guía de estudio.

¿En qué consiste esta guía?

La siguiente guía de estudio tiene como objetivo practicar y ejercitar los contenidos que hemos visto en clase.

¡Vamos con todo!



Tabla de contenidos

Consumiendo API REST	2
Actividad guiada: Persistiendo datos de una API REST	2
Almacenando datos de una API REST	3
¡Manos a la obra! - Almacenamiento de datos personales en archivo JSON con Node.js	5
¡Manos a la obra! - Impresión de datos almacenados	5
¡Manos a la obra! - Creación de un servidor para almacenar datos de un nuevo juego	5
¡Manos a la obra! - Creación de un servidor para acumular datos de varios videojuegos	5
¡Manos a la obra! - Creación de un servidor para consultar y almacenar usuarios desde una API	5
Soluciones	6



¡Comencemos!

Consumiendo API REST

Podemos consultar APIs en cualquier parte de nuestra aplicación con las herramientas ya conocidas (fetch, AJAX, axios) y sabiendo esto podríamos incluir a nuestro servidor una ruta para el almacenamiento de datos provenientes de una API REST.



Actividad guiada: Persistiendo datos de una API REST

Continuando con la temática de registro de usuarios, podemos ocupar la API de [randomuser](https://randomuser.me/) para guardar un usuario aleatorio y hacer una aplicación que consulte un servicio externo y persista la información obtenida en esa consulta.

Instala axios en tu proyecto y prosigue con los siguientes pasos para crear un ruta “/random” que consiga lo previamente dicho:

- **Paso 1:** Incluye la importación del paquete axios.
- **Paso 2:** Crea una ruta **GET /random**
- **Paso 3:** Utiliza axios para consultar la API en su end point: <https://randomuser.me/api/>
- **Paso 4:** Imprime por consola la data obtenida y concluye la consulta.

```
// Paso 1
const axios = require('axios')
```

```
// Paso 2
app.get("/random", async (req, res) => {
  // Paso 3
  const { data } = await axios.get("https://randomuser.me/api")
  // Paso 4
  console.log(data)
  res.send()
})
```

Ahora si consultas la siguiente dirección: <http://localhost:3000/random> y revisas la consola verás lo que se muestra en la siguiente imagen.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

[Symbol(kOutHeaders)]: [Object: null prototype] {
  accept: [Array],
  'user-agent': [Array],
  host: [Array]
},
data: {
  results: [ [Object] ],
  info: { seed: 'bb46bd515cbd8096', results: 1, page: 1, version: '1.3' }
}
```

Imagen 1. Data de axios mostrada por consola como resultado de una consulta al servidor.
Fuente: Desafío Latam

Como puedes apreciar, estamos viendo la data que por defecto nos trae axios y dentro de la propiedad "data" están las propiedades de randomuser.

Almacenando datos de una API REST

Bien, ahora que logramos consultar la API de randomuser a partir de la consulta a una ruta de nuestro servidor, lo que sigue es almacenar estos datos en nuestro Usuarios.json para persistir esta data consultada.

Para esto sigue los siguientes pasos:

- **Paso 1:** Guardar en una variable el objeto que estará en el índice 0 del arreglo results. Este objeto representa el usuario random que se está consultando y contiene en sus propiedades la información del mismo.
- **Paso 2:** Crea una variable usuario con los valores primer nombre, apellido y email del usuario random.
- **Paso 3:** Ingresa al arreglo usuarios, el usuario random.
- **Paso 4:** Sobrescribe el archivo "Usuarios.json" con la data nueva

```
app.get("/random", async (req, res) => {
  const { data } = await axios.get("https://randomuser.me/api")
  // Paso 1
  const randomUser = data.results[0];
  // Paso 2
  const usuario = {
    name: randomUser.name.first,
```

```
    lastname: randomUser.name.last,  
    email: randomUser.email,  
  };  
  // Paso 3  
  const { usuarios } = JSON.parse(fs.readFileSync("Usuarios.json",  
"utf8"));  
  usuarios.push(usuario);  
  // Paso 4  
  fs.writeFileSync("Usuarios.json", JSON.stringify({ usuarios }));  
  res.send();  
})
```

Ahora si consultas la siguiente dirección: <http://localhost:3000/random> y revisas el documento Usuarios.json podrás ver que se agregó un nuevo usuario random, tal como se muestra en la siguiente imagen:



```
1  {  
2    "usuarios": [  
3      {  
4        "name": "Akira",  
5        "lastname": "Toriyama",  
6        "email": "akayama@example.com",  
7        "password": "1234"  
8      },  
9      {  
10       "name": "Gichin",  
11       "lastname": "Funakoshi",  
12       "email": "gfunakoshi@example.com",  
13       "password": "abcd"  
14     },  
15     {  
16       "name": "Yann",  
17       "lastname": "Blanc",  
18       "email": "yann.blanc@example.com"  
19     }  
20   ]  
21 }
```

Imagen 2. Almacenando varios usuarios en un arreglo dentro del documento JSON.

Fuente: Desafío Latam



¡Manos a la obra! - Almacenamiento de datos personales en archivo JSON con Node.js

Desarrollar una aplicación Node que al ser ejecutada almacene un objeto con tu nombre y apellido en un archivo JSON.



¡Manos a la obra! - Impresión de datos almacenados

Imprime por consola tu nombre y apellido almacenados en el documento JSON.



¡Manos a la obra! - Creación de un servidor para almacenar datos de un nuevo juego

Desarrolla un servidor que almacene en un JSON los parámetros de la siguiente URL

<http://localhost:3000/nuevoJuego?nombre=Halo&consola=Xbox>



¡Manos a la obra! - Creación de un servidor para acumular datos de varios videojuegos

Desarrollar un servidor que acumule en un JSON varios videojuegos siguiendo los parámetros de la siguiente URL:

<http://localhost:3000/nuevoJuego?nombre=Halo&consola=Xbox>



¡Manos a la obra! - Creación de un servidor para consultar y almacenar usuarios desde una API

Desarrollar un servidor que al consultar una ruta /usuario almacene en un JSON los usuarios obtenido de la siguiente API <https://jsonplaceholder.typicode.com/users>

Soluciones

1. Desarrollar una aplicación Node que al ser ejecutada que almacene un objeto con tu nombre y apellido en un archivo JSON

```
const fs = require('fs')
let yo= {
  nombre: 'Brian',
  apellido: 'Habib',
}
fs.writeFileSync('Yo.json', JSON.stringify(yo))
```

2. Imprimir por consola tu nombre y apellido almacenados en el documento JSON

```
const fs = require("fs");
fs.readFile("Yo.json", "utf8", function (e, data) {
  const yo = JSON.parse(data);
  console.log(yo.nombre + " " + yo.apellido);
});
```

3. Desarrollar un servidor que almacene en un JSON los parámetros de la siguiente URL
<http://localhost:3000/nuevoJuego?nombre=Halo&consola=Xbox>

```
const express = require('express')
const app = express()
const fs = require('fs').promises

app.listen(3000, console.log("SERVER ON"))

app.get("/nuevoJuego", async (req, res) => {
  const { nombre, consola } = req.query;
  const juego= { nombre, consola };
  await fs.writeFile("juego.json", JSON.stringify(juego));
  res.send()
})
```

4. Desarrollar un servidor pueda acumular a través de una ruta /nuevoJuego en un JSON varios videojuegos siguiendo los parámetros de la siguiente URL
<http://localhost:3000/nuevoJuego?nombre=Halo&consola=Xbox>

```
{
```

```
"juegos": []  
}
```

```
app.get("/nuevoJuego", async (req, res) => {  
  const { nombre, consola } = req.query;  
  const juego = { nombre, consola };  
  const { juegos } = JSON.parse(await fs.readFile("Juegos.json"));  
  juegos.push(juego);  
  await fs.writeFile("Juegos.json", JSON.stringify({juego}));  
  res.send()  
})
```

5. Desarrollar un servidor que al consultar una ruta /usuarios almacene en un JSON los usuarios obtenido de la siguiente api <https://jsonplaceholder.typicode.com/users>

```
const express = require('express')  
const app = express()  
const fs = require('fs').promises  
const axios = require('axios')  
  
app.listen(3000, console.log("SERVER ON"))  
  
app.get("/usuarios", async (req, res) => {  
  const { data } = await  
  axios.get("https://jsonplaceholder.typicode.com/users")  
  await fs.writeFile('Users.json', JSON.stringify(data))  
  res.send();  
})
```