¿Qué hace cada parte del código?

Importación de módulos:

const express = require('express');

const fs = require('fs');

const path = require('path');

Estos son los módulos que use:

express: Permite crear el servidor y definir rutas.

fs: Nos deja leer y escribir archivos.

path: Sirve para manejar rutas de forma correcta en cualquier sistema operativo.

Configuración de la aplicación:

const app = express();

const PORT = 3000;

Aquí se crea la aplicación Express (app) y se define que usaremos el puerto 3000 para ejecutar el servidor.

Middleware para aceptar JSON:

app.use(express.json());

Este middleware permite que la aplicación pueda entender los datos enviados en formato JSON, que es como suelen enviarse desde herramientas como Postman.

Crear la ruta POST /alumno:

app.post('/alumno', (req, res) => {

Esta línea define una ruta de tipo POST con la dirección /alumno. Esta ruta será usada para enviar datos de un nuevo alumno.

Obtener los datos enviados:

const { cuenta, nombre, promedio, grado, grupo } = req.body;

Se extraen los datos del alumno desde el cuerpo de la solicitud (req.body).

Validación de los campos:

if (!cuenta || !nombre || promedio === undefined || !grado || !grupo) {

  return res.status(400).json({ error: 'Todos los campos son obligatorios.' });

}

Aquí se comprueba que todos los campos estén completos. Si falta alguno, se responde con un error 400 y no se guarda nada.

Crear el objeto del alumno:

const nuevoAlumno = { cuenta, nombre, promedio, grado, grupo };

Los datos del alumno se agrupan en un solo objeto llamado nuevoAlumno.

Definir el archivo donde se guardarán los datos:

const archivo = path.join(__dirname, 'alumnos.txt');

Esta línea crea una ruta correcta hacia el archivo alumnos.txt, que estará en la misma carpeta que app.js.

Leer los datos anteriores si el archivo ya existe:

```
let listaAlumnos = [];
if (fs.existsSync(archivo)) {
  const datos = fs.readFileSync(archivo, 'utf8');
  try {
    listaAlumnos = JSON.parse(datos);
  } catch (error) {
    listaAlumnos = [];
  }
}
```

Si el archivo alumnos.txt ya existe, se leen sus datos. Si no se puede leer correctamente, se crea una lista vacía.

Agregar el nuevo alumno:

```
listaAlumnos.push(nuevoAlumno);
```

Se añade el nuevo alumno a la lista de alumnos.

Guardar la lista actualizada en el archivo:

```
fs.writeFileSync(archivo, JSON.stringify(listaAlumnos, null, 2), 'utf8');
```

Se guarda toda la lista (incluyendo el nuevo alumno) en el archivo alumnos.txt, en formato JSON. El null, 2 sirve para que el texto se vea ordenado y fácil de leer.

Responder que todo salió bien:

```
res.status(201).json({ mensaje: 'Alumno guardado correctamente.' });
```

El servidor responde con un mensaje indicando que el alumno se guardó correctamente. El código 201 significa que algo fue "creado".

Iniciar el servidor:

```
app.listen(PORT, () => {
  console.log(`Servidor iniciado en http://localhost:${PORT}`);
});
```

Esta línea pone en marcha el servidor. Si abrimos la terminal, veremos el mensaje:

```
Servidor iniciado en http://localhost:3000
```

Conclusión

Este archivo app.js representa un ejemplo básico de cómo funciona una API en Node.js con Express. Nos permite enviar datos de alumnos y guardarlos de forma permanente en un archivo de texto como si fuera una pequeña base de datos.