02. API REST with Express

Creamos un nuevo proyecto

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Instalamos 3 paquetes

Texto

Descripción generada automáticamente

Creamos nuestro server.js e importamos las dependencias

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Creamos nuestros 4 endpoints que corresponderían con un CRUD básico.

Texto

Descripción generada automáticamente

GET

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico, Sitio web

Descripción generada automáticamente

POST

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

UPDATE

Ponemos el id:0 correspondiente al user creado previamente, para enviárselo por params

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Y actualizamos su nombre

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Comprobamos con el GET que se ha actualizado

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

DELETE

Borramos el usuario con id:0

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Comprobamos con el GET que se ha borrado y vemos que no hay nada en el array de accounts

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

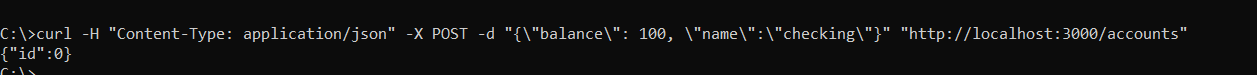
Descripción generada automáticamente

Ahora probaremos nuestra aplicación con CURL

Después de configurar el Windows siguiendo la guía para que funcione con el comando CURL

Probamos ya nuestra api

POST

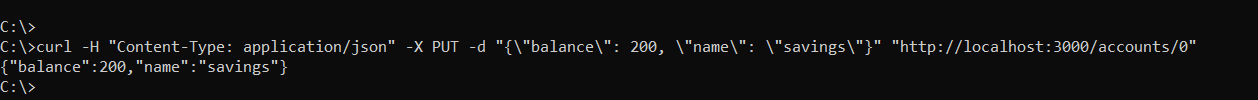


comprobación con GET para probar que se haya ejecutado bien

Texto

Descripción generada automáticamente

Hacemos el UPDATE



Comprobamos con el GET

Texto

Descripción generada automáticamente

Y ahora borramos con el DELETE y comprobamos con GET

Texto

Descripción generada automáticamente

Ahora vamos a comprobar que nos muestra la petición de tipo GET poniendo -iv delante

Texto

Descripción generada automáticamente

Con esta opción lo que tenemos es que en la petición con el **-i** obtenemos solo los encabezados de la respuesta y no el cuerpo del contenido y con la opción **-v** la usamos para mostrar información detallada de la solicitud HTTP incluyendo la solicitud enviada y la recibida.

**Una vez hecho el tutorial pasamos a la actividad propuesta A21**

**Responde a estas preguntas:**

**El código utiliza let en lugar de var para declarar variables. ¿Cuál es la diferencia?**

La diferencia entre LET y VAR principalmente esta en el SCOPE de la variable, es decir limita el alcance de la variable en nuestra app.

**¿En qué parte de la aplicación se realizan las importaciones? ¿Dónde se lanza express()? ¿Dónde se utiliza el middleware? Añade comentarios al código donde se ven estas partes.**

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**En el código se devuelven varios códigos de estado HTTP. ¿Cuáles son? Explica su significado.**

En el código vemos varios códigos de estado HTTP

**200:** Indica que la solicitud ha tenido éxito.

**201:** Indica que la solicitud ha tenido éxito y ha llevado a la creación de un recurso.

**204:** Indica que la solicitud se ha realizado correctamente, pero que el cliente no necesita salir de la página en la que se encuentra.

Ahora probamos a crear una nueva ruta para entrar directamente a una cuenta con un ID

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente