Conceptos importantes:

**Axios** es una biblioteca de JavaScript que se utiliza para hacer **solicitudes HTTP** desde el navegador o un entorno Node.js. Es muy popular en aplicaciones web porque facilita la comunicación entre el frontend y el backend, lo que permite enviar y recibir datos de un servidor.

**Principales usos de Axios**

1. **Enviar solicitudes al servidor:**

* **GET:** Obtener datos del servidor (por ejemplo, recuperar una lista de productos).
* **POST:** Enviar datos al servidor (por ejemplo, registrar un usuario).
* **PUT/PATCH:** Actualizar recursos existentes (por ejemplo, modificar un perfil de usuario).
* **DELETE:** Eliminar recursos (por ejemplo, borrar un producto).

**BACK END:**

Para inicializar un proyecto de back, debemos instalar lo siguiente:

1. Npm init -y
2. Npm install express ejs Morgan sequelize mysql2
3. Npm install –save-dev nodemon para reiniciar automáticamente el servidor
4. Verificar si esta funcionando correctamente:

Explicación de package-json del back-end:

{

"name": "back",

"version": "1.0.0",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1",

"dev": "nodemon ./src/index.js"

},

"keywords": [],

"author": "",

"license": "ISC",

"description": "",

"dependencies": {

"bcrypt": "^5.1.1",

"cookie-parser": "^1.4.7",

"cors": "^2.8.5",

"dotenv": "^16.4.5",

"ejs": "^3.1.10",

"express": "^4.21.1",

"express-session": "^1.18.1",

"inquirer": "^12.0.1",

"morgan": "^1.10.0",

"mysql2": "^3.11.3",

"sequelize": "^6.37.4"

}

}

**Dependencias de tu proyecto:**

1. **bcrypt:**  
   Biblioteca para encriptar contraseñas. Se utiliza para guardar contraseñas de forma segura en la base de datos.
2. **cookie-parser:**  
   Middleware de Express para analizar cookies enviadas en las solicitudes HTTP. Ayuda a trabajar con cookies fácilmente.
3. **cors:**  
   Permite configurar **CORS (Cross-Origin Resource Sharing)**, que habilita o restringe solicitudes HTTP desde diferentes dominios o puertos. Es esencial para aplicaciones front-back que interactúan desde distintos servidores.
4. **dotenv:**  
   Permite cargar variables de entorno desde un archivo .env. Es útil para gestionar configuraciones sensibles como contraseñas, claves API, etc.
5. **ejs:**  
   Un motor de plantillas que permite generar HTML dinámico desde el servidor usando JavaScript. Es ideal para aplicaciones que combinan front-end y back-end.
6. **express:**  
   Un marco de trabajo para Node.js que simplifica la creación de servidores y manejo de rutas. Es la base para tu aplicación.
7. **express-session:**  
   Middleware de Express que maneja sesiones de usuario en el servidor, permitiendo almacenar datos específicos para cada cliente entre solicitudes.
8. **inquirer:**  
   Biblioteca para crear interfaces interactivas de línea de comandos. Se usa para capturar entrada del usuario en la terminal.
9. **morgan:**  
   Middleware de registro de solicitudes HTTP en la consola. Es útil para depuración y monitoreo de solicitudes.
10. **mysql2:**  
    Cliente para conectarse y ejecutar consultas en bases de datos MySQL desde Node.js. Es más rápido y compatible que el paquete original mysql.
11. **sequelize:**  
    Un ORM (Object-Relational Mapper) que facilita la interacción con bases de datos relacionales (como MySQL). Permite trabajar con datos como objetos en lugar de escribir consultas SQL directamente.

**Script adicional:**

* "dev": "nodemon ./src/index.js":  
  Inicia tu aplicación utilizando **Nodemon**, que reinicia automáticamente el servidor cada vez que detecta cambios en los archivos.

**PRUEBA EN INDEX.JS:**

const express = require('express');

const app = express();

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

res.send('Servidor funcionando correctamente');

});

app.listen(3000, () => {

console.log('Servidor corriendo en http://localhost:3000');

});

**TRY CATCH:**

Se usa comúnmente para capturar errores en funciones asíncronas (como peticiones a una API) y asegurarse de que la aplicación no se detenga si ocurre un problema.

**¿Qué hace Try...Catch?**

1. **try:** Aquí colocas el código que podría fallar, como una solicitud a un servidor.
2. **catch:** Si algo dentro del bloque try falla (lanza un error), el bloque catch lo captura y puedes manejar ese error, como mostrar un mensaje al usuario.

**que es deploy?**

**ChatGPT dijo:**

ChatGPT

El **deploy** (o "despliegue") es el proceso de tomar una aplicación o proyecto que has desarrollado y configurarlo para que esté disponible y funcionando en un entorno real o de producción. Esto permite que los usuarios finales puedan acceder y utilizar la aplicación, generalmente a través de internet.

**En el contexto de desarrollo web, el deploy implica:**

1. **Preparación del código**:
   * Asegurarte de que tu aplicación esté lista para ejecutarse fuera del entorno de desarrollo.
   * Minimizar archivos (como CSS y JavaScript).
   * Configurar variables de entorno (como claves de API, bases de datos, etc.).
2. **Subida a un servidor**:
   * Copiar los archivos del proyecto al servidor donde se alojará la aplicación.
   * Esto puede incluir servicios como AWS, Heroku, Vercel, Netlify, o un servidor privado.
3. **Configuración del servidor**:
   * Instalar dependencias necesarias (como Node.js, Python, etc.).
   * Configurar la base de datos, si es necesaria.
   * Asegurarse de que el servidor pueda ejecutar tu aplicación correctamente.
4. **Publicación**:
   * Asociar tu aplicación a un dominio (por ejemplo, www.tusitio.com).
   * Configurar HTTPS para garantizar conexiones seguras.
5. **Mantenimiento**:
   * Actualizar el código según sea necesario.
   * Monitorear el rendimiento de la aplicación.

**Ejemplo práctico:**

Si estás desarrollando un e-commerce como el de tu proyecto, el proceso de **deploy** consistiría en:

* Subir el frontend (React) a un servicio como Netlify o Vercel.
* Subir el backend (Node.js + Express) a un servidor como AWS, Heroku o Render.
* Configurar una base de datos como MySQL en un servicio en la nube.
* Enlazar el dominio de tu negocio (por ejemplo, happyfeet.com).

Una vez completado, tu aplicación estará accesible para los usuarios en internet. 😊