Express y cómo funciona (mas o menos)

express es un framework que funciona en conjunto con node.js después de hacer la instalación de node.js dentro del terminal con “npm i express”, express sirve para agilizar el código dentro de la aplicación usando funciones ya hechas dentro del framework a diferencia de otros frameworks donde hay que seguir una estructura aquí solo se dan las funciones y el usuario puede estructurar como sea el código, funcionando con requests y responses lo cual es óptimo para su utilización en la base de datos, normalmente este se encuentra en stacks como podría ser el stack de [mongodb, nodejs, express y react], se usa express para crear endpoints y interactuar con la base de datos, pudiendo juntar aplicaciones móviles y webs(óptimo para el proyecto actual).

ejemplo con node.js normal:

```
const http = require('http');
const fs = require('fs');

const server = http.createServer((req, res) => {
    const read = fs.createReadStream('./static/index.html')
    read.pipe(res);
})
server.listen(3000)
console.log('Server listening on port 3000');
```

-aqui se importa el módulo http de nodejs para crear el servidor
-se usa fs para poder manipular los archivos del sistema, como podría ser leer o editar un html y enviarlo al usuario
-se crea el servidos con http.create server req la solicitud del cliente y el res la respuesta del servidor
-se lee la carpeta donde se encuentra el html
-después de leerse se envía inmediatamente al servidor
-el server escucha el puerto 3000 que se va a usar para buscarlo como http (el localhost)
-muestra el mensaje en consola de qué está escuchando

ejemplo con express:

```

const express = require('express');
const app = express();

app.get('/', (req, res) => {
    res.sendFile(__dirname + '/static/index.html');})

app.listen(3000)
console.log('Server listening on port 3000');
```

-se importa el express

-se inicializa el express y esta será la app que responda las peticiones en cambio a http(app es lo mismo que server)
-se saca el html que será usado para el localhost junto con los parámetros del request y response(el “/” es para definir la ruta por ejemplo “/perfil” hará que el localhost:3000/perfil lleve al html designado)
-se define qué archivo se va a enviar y el directorio de este como respuesta
-de por si el express cuenta con readstream que es que vaya leyendo de a poco los elementos para agilizar la carga en la página
- se define el port

---

como se puede ver aquí no se hace uso directamente del http, se tiene el readstream integrado al igual que el pipe, el app ya cuenta con funciones como app.listen, app.request o app.delete que aligeran el uso de constantes y sea más legible, como se puede ver se usan funciones ayudan a reducir la cantidad de código que existe en un proyecto pequeño se puede ver ínfimo pero en proyectos grandes como el que se hará ayudarán enormemente a reducir la cantidad de información almacenada.

en general siento que se ve mucho más limpio y entendible a comparación del solo uso de nodejs, como tiene funciones internas que de por si hacen el trabajo como podría ser el readstream saca la carga de tener que ponerlo en todas las veces que se quiera llamar a una url al igual que errores como olvidarse de poner ciertos elementos del código, para un código pequeño el cambio se nota y se vé que aligerará el trabajo realizado en el backend.