

# Taller Express y otras cosas



Nombre: Leonardo Mario Alberto Guillen Soria

Matrícula: A00574110

Película favorita y razón: HOME - Siempre hay un buen amigo

**El objetivo de la actividad es explorar y conocer más sobre el desarrollo web del lado del backend.**

**Al final de la actividad deberás hacer un PDF con este documento. El PDF deberá tener un enlace en supertarea.html**

- **Ejercicio 0: El pasado**

En las semanas previas vimos la forma de consumir un API desde el cliente (el navegador). Ahora consumirás el **API desde el servidor**. Usa javascript y nodejs.

Selecciona un API que no requiera autenticación.

El servidor deberá consumir el API, procesarlo de alguna manera (ejemplo: seleccionar un dato) y posteriormente entregar ese dato (o datos) al cliente (navegador).

Sube tu archivo a tu repositorio.

Escribe aquí abajo el enlace al archivo en tu repositorio:

<https://github.com/leonardo-guillen/leonardo-guillen.github.io/tree/main/act5/Ejercicio0>

¿Cuál fue el principal problema para resolver este ejercicio? ¿Cuál fue el problema que tuvo una de las personas en tu equipo? Explica cómo resolvió esa persona ese problema.

**API: <https://f1api.dev/api/2010/drivers>**

**Docs: <https://f1api.dev/docs/drivers/year-drivers>**

**El principal problema fue encontrar un API libre y comprender cómo utilizarla, lo cual solucioné leyendo detalladamente su documentación.**

**Alejandro: Mi compañero Alejandro tuvo problemas al conectarse con el API, pero lo solucionó leyendo un poco la documentación.**

¿ Recuerdas cómo **instalar un módulo** en node.js? Escribe la instrucción que se usa:

**Si, para instalar un módulo de node tendremos que utilizar un instalador como npm o pnpm agregando 'install' y el nombre del módulo, en este caso express para poder trabajar nuestro script como servidor.**

## **pnpm install express**

Investiga qué es express (en el contexto de desarrollo web en nodejs). Explica a un **niñ@** qué es express, cómo se usa, para qué sirve. Escribe ese texto aquí. Incluye también **las referencias** de tu investigación.

**Express es una herramienta que nos permite escribir mediante texto lo que hace una aplicación por detrás.**

**Express no es la única herramienta que nos permite esto, pero nos facilita la forma de realizarlo y además hace que nuestras instrucciones sean más ordenadas y claras, para que a ti te llegue lo que quieras ver y no otra cosa como un video de bob esponja cuando tu querias ver uno de ben10.**

**Esto quiere decir que, tomando como ejemplo Youtube que lo utilizas bastante. Yo como programador puede escribir una indicación que está siempre esperando para poder responderte, un ejemplo es cuando ver un video, a ti te sale con su imagen y su título, si tu presionas ese botón lo que hará será enviar una señal a mi código escrito que será como decirle 'HEY HEY HEY, quiero ver este video, por favor :3'.**

**Entonces, mi código lo que hará será buscar ese video que tu quieras ver y enviarlo hacia tu teléfono solo para que lo puedas ver, pero esto no descargara el video en tu celular, solo irá enviando el video conforme tu lo vayas viendo.**

**Entonces, express me facilita a mí la posibilidad de crear instrucciones que puedan manejar datos o mensajes enviados por ti como persona y luego regresarte otros datos, en este caso el vídeo que quieras ver.**

**Su uso es interesante, ya que puedes escribir las instrucciones, pero tendrás que seguir una estructura, así como tu cuando estudias divisiones, tienes que ir paso a paso respetandolos, porque si no el resultado de la división sería incorrecto.**

Dile a una persona de la clase (que no esté en tu mesa) que te explique qué es express y qué ventajas te podría dar su uso. Graba el audio de la explicación.

**Me ayudó mi compañero ángel.**

¿Crees que realizó una buena explicación? ¿Por qué?

**Realizó una explicación muy buena, sabe bastante sobre el porqué es bueno utilizar express para nuestras páginas y el porque es una excelente opción.**

**Me platico sobre la facilidad que tiene Express para su implementación, ya que nos hace más fácil su entendimiento e implementación, además de ser muy ligero y tener una rapidez muy buena, así como la facilidad del enrutamiento de las páginas.**

- **Ejercicio 3: Desde un libro**

Investiga una definición en un libro (digital o físico) **de una API**. Incluye la referencia del libro.

Escribe un ejemplo de un endpoint de tu proyecto.

**Un endpoint en mi proyecto serán las rutas con su respectiva funcionalidad. Un ejemplo es cuando una escuela busque a algún donador, la web enviará el input a un endpoint específico en el API y está tratará este input para buscar los posibles donadores y después retorna la información de cada uno de los donadores.**

**De esta manera el cliente recibirá la información y se le mostrará en forma de tarjetas.**

**Express instalado: pnpm install express**

Escribe la instrucción que se usa para **instalar el módulo de express:**

**pnpm || npm install express**

- **Ejercicio 5: Uso de express**

Ejecuta y explica que hace el siguiente código:

```
import express from 'express';

const app = express();

app.listen(1984, () => {
  console.log('Up and up');
});
```

**Este código lo que realiza es primeramente importar el módulo express que nos permitirá inicializar nuestro servidor, después utilizamos express() para iniciar nuestra aplicación y finalmente con el método listen(), dándole como primer argumento el puerto, inicializará nuestro servidor en escucha en el puerto 1984, finalmente como instrucción nos imprimirá el mensaje en la terminal.**

- **Ejercicio 6: Uso de express++**

Ejecuta y explica que hace el siguiente código:

```
app.get('/bienvenida', (req, res) => {
  res.send('Esto no es una página html');
});

app.get('/otraBienvenida', (req, res) => {
  res.sendFile('bienvenida.html');
});
```

Este código genera dos endpoints que son para ser utilizados mediante peticiones GET, de esta manera si es llamada a '/bienvenida' nos retorna el mensaje 'Esto no es una página html.'

En cuanto a `res.sendFile()`, nos permite enviar archivos.

En caso de que alguno de estos te marque error, soluciona el problema y explica la forma de solucionarlo:

En cuanto a `res.sendFile()`, sucede un error debido a que nuestro servidor no sabe en dónde buscar el archivo, es por ello que aquí entra en juego otra dependencia la cual es 'path' mediante el método `join()`, pasándole como primer argumento el path del directorio donde deseamos buscar el archivo y como segundo argumento el nombre del archivo.

- **Ejercicio 7: Recuerdo de imagen**

Recuerdas que la imagen no se podía ver en <https://github.com/sgiomatec/act2025>. Un servidor entrega respuestas a solicitudes realizadas por un cliente. Cuando el

cliente solicita una imagen (u otro tipo de archivo) el servidor tiene que saber responder.

Express nos propone crear un directorio para los archivos que queremos usar, por ejemplo imágenes o archivos de hojas de estilo.

Investiga qué es express.static. Escribe tu investigación.

Escribe tu investigación

**Express.static() es una función de expresión que nos permite servir los archivos estáticos como PDF, CSS, JavaScript o HTML en una ruta.**

**Si tenemos una carpeta llamada 'public'**

**Podremos utilizar app.use(express.static('public'))**

**De esta manera esta carpeta será accesible desde el navegador.**

- **Ejercicio 8: Película**

Pregunta a una persona de otra mesa su película favorita y sus razones.

¿Ya viste esa película?. ¿Te gusta? ¿La verías si no la has visto? ¿Por qué?

**Nombre: Sebastian**

**Pelicula: Interstellar**

**¿Por qué le gusta?**

**Por su historia, actuaciones, banda sonora, pero principalmente la interpretación del espacio y física aplicadas, ya que es algo que no ha comprobado pero que podría ser posible.**

**Yo:**

**Es una película que ya he visto y me parece muy interesante, especialmente por la forma en que dan a mostrar la física y el espacio, ya que lo muestran de una forma muy interesante y que además es algo que podría ser posible.**

- **Ejercicio 9: Transformación**

Usa express para transformar todo el código de servidor.js, nombra el archivo ahora servidor\_express.js

Transformacion del servidor creado con http a servidor con express:

<https://github.com/leonardo-guillen/leonardo-guillen.github.io/blob/main/act5/servidor.js>

Escribe aquí abajo el enlace al archivo en tu repositorio:

- **Ejercicio 10: Verbos**

En el contexto de desarrollo web, explica los siguientes verbos GET, POST, PUT, DELETE

**GET ->** Nos permite realizar peticiones a un API de una forma en la que el cliente que realiza la petición esperara datos de vuelta sin enviar datos.

**POST ->** Nos permite realizar peticiones a un API de una forma en la que el cliente puede enviar datos al servidor para darle contexto de la situación y finalmente el servidor o API retorna un código de estado. Un ejemplo común es utilizarlo para un login.

**PUT ->** Nos permite realizar peticiones en las cuales enviamos datos al servidor o API y estos datos serán utilizados para generar nuevos datos. Un caso de uso es para generar o almacenar nuevos datos en nuestra base de datos.

**DELETE ->** Nos permite realizar peticiones para eliminar información que ya tengamos almacenada en nuestro servidor o base de datos.

Explica los siguientes códigos, en el contexto de desarrollo web:

**1xx**

1xx se usa para respuestas informativas, en casos para indicarle al usuario que se está procesando la información que se ha enviado pero puede seguir enviando más información sin problemas.

#### Códigos de estado usuales:

**100 continue ->** El servidor ha recibido los encabezados y el usuario puede seguir enviando el cuerpo de la solicitud.

**2xx**

2xx se utiliza para respuestas exitosas, que esto se ve representado en un caso en el que un API o servidor devuelve datos o una confirmación exitosa de ejecución.

### Códigos de estado usuales:

**200 OK** -> Toda la petición se ejecutó correctamente.

**201 Created** -> Se generó un nuevo recurso correctamente.

**204 No Content** -> La solicitud fue exitosa, pero no existe contenido que deba devolverse.

### 3xx

3xx se utiliza para las redirecciones, usualmente se generan las redirecciones para enviar a los usuarios a otras páginas, ayudando a optimizar el rendimiento de la caché.

### Códigos de estado usuales:

**301 Moved Permanently** -> Cuando la URL cambia de forma definitiva.

**302 Found** -> Cuando se generó una redirección temporal, como cuando un usuario no está logueado y es enviado al login.

**304 Not Modified** -> Se usa cuando el recurso no ha cambiado y por ende el navegador puede utilizar el mismo cache.

### 4xx

4xx se utiliza para contemplar los errores del cliente, ayudando a llevar el manejo de errores en formularios, accesos restringidos y páginas inexistentes.

### Códigos de estado usuales:

**400 Bad Request** -> Se genera cuando una solicitud es inválida, con los datos incorrectos o generados de forma incorrecta.

**401 Unauthorized** -> Se utiliza cuando se requiere de autenticación.

**403 Forbidden** -> Se utiliza cuando un acceso es denegado.

**404 Not Found** -> Se utiliza de forma típica cuando un recurso no existe, como una página.

### 5xx

5xx se utiliza para contemplar los errores que se den en el servidor y los cuales requieren de depuración.

### Códigos de estado usuales:

**500 Internal Server Error** -> Se utiliza cuando se ocasiona un error en el servidor.

**502 Bad Gateway** -> Se utiliza cuando el servidor recibe una respuesta inválida de otro servidor.

**503 Service Unavailable** -> Se utiliza cuando el servidor está sobrecargado o en mantenimiento.

Explica con un ejemplo (relacionado con tu proyecto/reto) cómo se atendería una solicitud de cada uno de esos verbos usando express. Incluye al menos una respuesta 404, 201 y 200.

**404** -> Mostrará al cliente que la página por la que se está buscando no existe, es el código de error típico para indicar que una página no existe.

**200** -> Se considera un código de estado correcto, donde el cliente si puede acceder a esta página sin ningún inconveniente y por ende le será mostrada.

**201** -> Se considera un código de estado correcto en caso de utilizar el método POST, como cuando se genera un nuevo registro de inicio de sesión, si todo sale correctamente se generará un código de estado 201.

### • Ejercicio 11: Postman

En equipo con tu mesa, instalen postman en cada una de sus computadoras. Tomen una misma foto en la que aparezcan todas las pantallas con postman abierto.

Foto aquí:



Explica para qué sirve Postman

Es una herramienta que nos permite probar de forma sencilla las API, de tal manera que no tendríamos que escribir código y podríamos enviar solicitudes de los distintos métodos existentes y podremos ver las respuestas a nuestras peticiones sin necesidad de escribir código.

- **Ejercicio 12: Parámetros**

Inventa un ejercicio (que esté relacionado con escuelas/donantes) para que uses parámetros en las rutas (por ejemplo /getEscuelas)

**Obtener el nombre de un usuario en base a su ID almacenado en la base de datos.**

Soluciona ese ejercicio

**La forma en la que lo solucione es generando una ruta la cual reciba un id 'getUser/:id' de esta forma mi API puede recuperar este id 'req.params.id' y con ello podremos consultar la informacion, en este caso de un json y enviar la informacion del nombre al cliente.**

**El cliente solo tendra que colocar el id de usuario en la ruta: '/getUser/1', haciendo referencia al id 1 en este caso.**

Escribe aquí abajo el enlace al archivo en tu repositorio:

Link:

Compara ese ejercicio con el de otra persona de la clase. ¿Qué ejercicio ayuda mejor a comprender el uso de parámetros y por qué?

**Justamente este ejercicio ya que es utilizado de forma mucho más sencilla y se entiende correctamente lo que realiza y el propósito de implementarlo de esta manera.**

- **Ejercicio 13: npx**

Investiga con alguien más lo siguiente: npx express-generator. Importante haz un directorio diferente si quieres ejecutar el comando.

Escribe el nombre de ese alguien más:

Alejandro

Explica lo que genera y hace ese comando.

**Es una herramienta que nos genera rápidamente la estructura de un proyecto con expressjs, lo que no evita el tener que configurarlo manualmente.**

**De esta manera ya podremos empezar a trabajar con él, similar a cuando utilizamos un framework para el front, ya que nos otorgan una configuración inicial.**

Explica de manera básica y conceptualmente qué archivos se generaron.

**Se generó la siguiente estructura en nuestro proyecto con Express.js**

**bin/** -> Tiene el archivo que ayudará a inicializar el servidor  
**public/** -> Puede almacenar los archivos estáticos tales como html, css, js, etc.  
**routes/** -> Almacena las rutas de nuestra API en archivos individuales, lo que permite tener una mejor estructura y control.  
**views/** -> Esta carpeta nos ayuda a almacenar plantillas, tal como en el front, que podemos hacer elementos o layouts con html.  
**app.js** -> Este archivo almacena la configuración principal e inicial de nuestro proyecto con express  
**Packge.json** -> Aquí se almacena la configuración de la referencia hacia nuestras dependencias y scripts de npm, tales como las dependencias que instalamos para el front-end, como react-router-dom, o react. Pero en el caso de back-end se utilizarán dependencias distintas.

- **Ejercicio 14: Explicar**

Explica a una niña o un niño de 10 años para qué sirven frameworks como Angular, React, Vue, o Svelte.

**Un framework de este tipo podríamos considerarlo como una plantilla, ya que esta nos da una base sobre la cual podremos construir una página de internet.**

**Lo que esto llamado framework nos proporciona es una facilidad de implementar muchas cosas para mejorar la funcionalidad de nuestra página, herramientas que nos permiten colocar elementos más movidos como leones voladores en una página de internet o incluso algo más sencillo como solo dejar una imagen aburrida de fondo.**

**Básicamente un framework a un programador nos ayuda a tener una base construida para empezar a generar nuestra página.**

**Sería algo así como, cuando vamos a construir un castillo de arena en la playa, que nuestra tierra ya esté plana y por ende podremos construirlo sin problemas, pero si la tierra no está para nada plana, por ende sería algo con mucho más trabajo e incluso cansado.**

**Entonces un framework nos otorga esta facilidad y base para empezar a construir nuestra página.**

Ref

<https://kinsta.com/es/base-de-conocimiento/que-es-express/> [2]