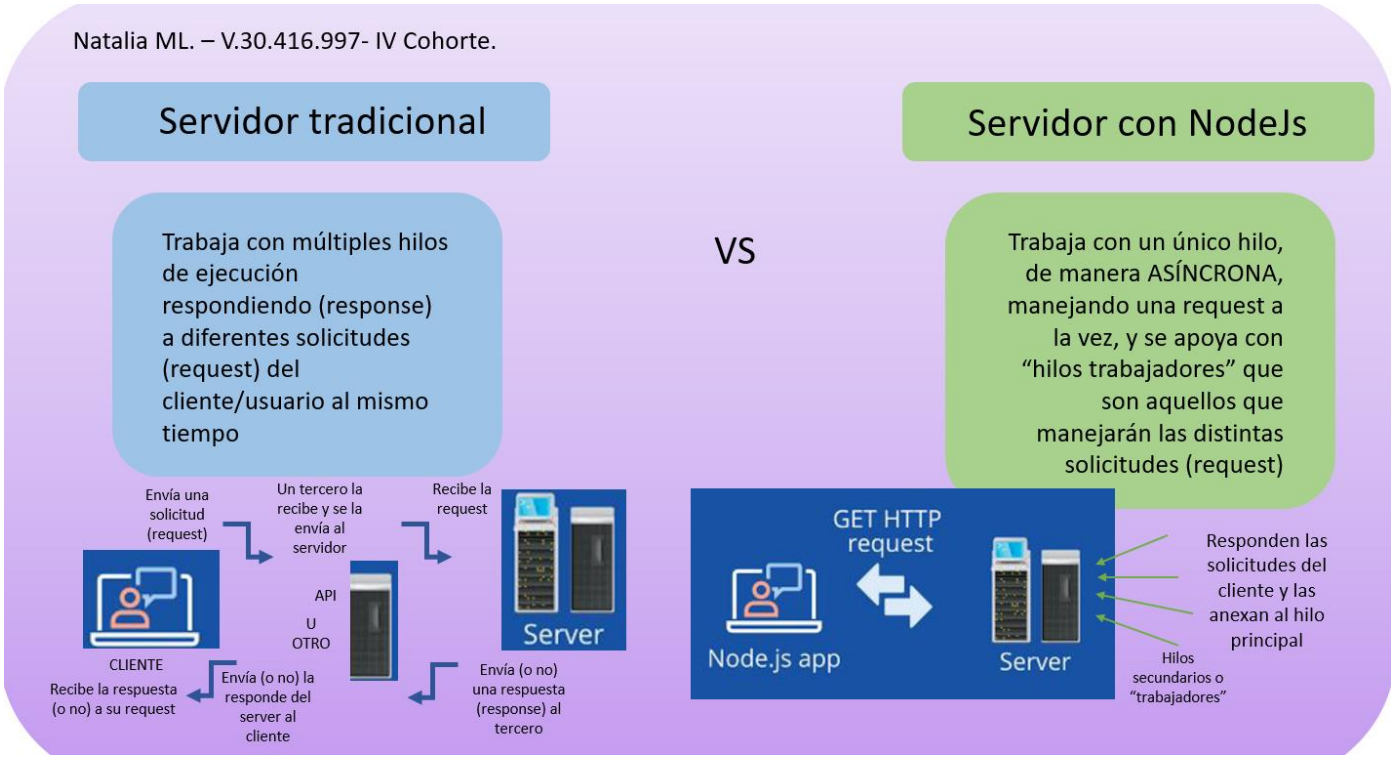
Natalia Mora L. – V.30.416.997 – IV Cohorte.

Evaluación 1 (30 puntos)

I. Desarrolle:

A. Elabore un diagrama del funcionamiento de un servidor desarrollado en NodeJs y describa su diferencia con un servidor tradicional. (5 puntos)

- Nota!: Entiéndase como respuesta a esta pregunta, de acuerdo a lo explicado por el docente, que la misma puede ser; un diagrama de conceptos, mapa conceptual o una descripción que explique el funcionamiento de un servidor desarrollado con NodeJs a un servidor tradicional



B. Describa la importancia y el uso de los módulos en Node Js, de un ejemplo de cómo construir un modulo y cómo usar dichos módulos dentro de la aplicación. (5 puntos)

- B.1) La importancia en usar los módulos de NodeJs se encuentra en precisamente, NodeJs permite usar y crear módulos desde cero.

- B.2) Se anexa a continuación un ejemplo de módulos (para crearlo y usarlo, en este caso con vscode se anexa el siguiente código)

Se entiende por módulo aquel espacio designado a contener código útil a usar en repetidas ocasiones, de los módulos manejados con NodeJs hasta el momento se conocen: 'fs' para manejo de archivos, 'http' y 'url'.

- Para construir un módulo; seleccionamos un directorio o espacio en donde se desarrollará el módulo en cuestión, creamos un archivo .js, importamos aquellas librerías, herramientas o módulos pre existentes que se requieran, (en este ejemplo 'fs/promises').
- Desarrollamos la función asíncrona a utilizar.

```
const { open} = require('fs/promises');
async function openFile(fileName, data) {
    try {
    const file = await open(fileName, 'w');
    await file.write(data);
    console.log(`Opened file ${fileName}`);
    } catch (error) {
    console.error(`Got an error trying to open the
    file: {error.message}`);
    }
}
```

- Exportamos el módulo creado luego de la func. async

```
module.exports = { openFile }
```

- Llamamos en nuestra app principal, a través de una importación, el módulo creado y usamos según se necesite.

C. ¿Para ti es recomendable usar nodejs para el desarrollo de aplicaciones? ¿*Cómo* definirías qué es Node Js? (5 puntos)

- C.1) Es recomendable usar NodeJS si se requiere desarrollar una app web con su backend y frontend en un mismo lugar, modularizado, que requiera crear y usar diferentes módulos y librerías, además de necesitar un servidor asíncrono, entonces sí es recomendable usar NodeJS.

- C.2) Node Js es un entorno de ejecución que ejecuta el motor de Chrome V8 pero no se limita a este, trabaja con un único hilo de ejecución que se apoya de hilos secundarios conocidos como “hilos trabajadores” para responder de manera asíncrona a las distintas solicitudes del usuario. Además, es rápido, escalable, de código abierto y multiplataforma, en concreto, es un servidor útil para aquellos desarrolladores que deseen trabajar con diferentes frameworks, librerías, de forma modularizada y con el desarrollo backend y frontend al mismo tiempo.

II. Ejercicio 15 puntos

Descripción:

En este proyecto, tendrás la tarea de crear un servidor HTTP personalizado utilizando los módulos y herramientas que hemos estudiado en clase. El servidor deberá tener una página de presentación incorporada, creada con HTML y estilos definidos en un archivo CSS vinculado desde el head de la página. Además, se requerirá la implementación de un sistema para mostrar y guardar información importante del autor y la fecha de inicio del servidor.

Requisitos:

1. **Creación del Servidor HTTP:**
   - Crea un servidor HTTP utilizando Node.js y los m6dulos necesarios para gestionar solicitudes y respuestas web.
2. **Página de Presentación:**
   - Crea una página de presentación (página de inicio) en HTML.
   - Agrega estilos a la página utilizando un archivo CSS separado.
   - La página de presentación debe mostrar información relevante, como el nombre del autor de la pagina y la fecha de inicio del servidor.
3. **Información del Autor:**
   - El nombre del autor deberá ser incorporado en la pagina de inicio.
   - La fecha de inicio del servidor deberá ser mostrada en formato "dd/MM/yyyy".
   - La fecha de inicio deberá guardarse en un archivo JSON llamado "autor.json" con la estructura {"autor": "Nombre del Autor", "fechalnicio": "dd/MM/yyyy"}.
4. **Método de Consulta:**
   - El servidor web debe implementar un método que permita a los usuarios consultar la información almacenada en el archivo "autor.json". Cuando un usuario accede a una **URL** específica (por ejemplo, /autor), el servidor debe responder con el contenido del archivo **JSON.**

Entregables:

- Código fuente del servidor **HTTP** en Node.js.
- Archivo **HTML** de la página de presentación.
- Archivo CSS para los estilos.
- Archivo "autor.json" con la información del autor y la fecha de inicio.
- Documentación detallada que explique cómo iniciar y acceder al servidor, cómo consultar la información del autor y cualquier otro detalle relevante.

Observaciones:

- Asegúrate de que tu servidor funcione correctamente y que los estilos se apliquen adecuadamente en la página de inicio.
- Verifica que el método de consulta de información del autor esté funcionando según lo requerido.