Evaluación 1 módulo 3

1.- Informe de Investigación: Generalidades del Lenguaje JavaScript:

- Desarrolla un informe que cubra los siguientes aspectos:

- **Historia de JavaScript:**
  - Fue creada en 1995 por Brendan Eich en tan solo 10 dias, como un lenguaje de scripting para Netscape navigator para darle interactivad básica a los sitios web. Hoy es fundamental en el desarrollo web dado que permite crear interfaces dinámicas e interactivas, es compatible con todos los navegadores y además permite crear servidores con tecnologías como node.js

- **Uso de JavaScript en Navegadores Web:**
  - JavaScript es el único lenguaje de programación que los navegadores ejecutan de forma nativa para manejar interactividad y el DOM, permitiendo modificar contenido en tiempo real, responder a eventos, validar formularios, hacer peticiones, etc.

- **Entornos Virtuales de JavaScript:**
  - Puede ejecutarse en entornos virtuales del lado del cliente, en este caso los navegadores como: Google Chrome, mozilla, Edge, etc. Así como en entornos virtuales del lado del servidor con node.js

- **Diferencias entre JavaScript y otros lenguajes:**
  - Es de **tipado dinámico** a diferencia de C++ / Java.
  - Es **interpretado** a diferencia de C / C++ / Java.
  - Es **multiparadigma** dado que soporta programación imperativa, funcional u orientada a objetos.

- **Fortalezas y debilidades de JavaScript:**

    - **Fortalezas:**
        - **Asincronía:** para ejecutar peticiones de eventos en hilos secundarios sin bloquear al principal
        - **Extensibilidad:** dado los múltiples frameworks, librerías y bibliotecas disponibles: React, Next, Node.js, Nest , etc.
        - **Flexibilidad:** gracias a su tipado dinámico permite flexibilidad para trabajar con datos y estructuras.

    - **Debilidades y/o limitaciones**:
        - **Manejo de errores:** debido al tipado dinámico se pueden provocar errores difíciles de rastrear dado que variables pueden cambiar de tipo en cualquier parte del script.
        - **Inconsistencia entre navegadores:** si bien todos los navegadores soportan JavaScript, existen diferencias en la interpretación del código de cada navegador.

- **JavaScript como lenguaje asíncrono:**
    - Javascript usa event loop para gestionar tareas como peticiones HTTP (esenciales para el propósito de un sitio web) sin bloquear el hilo principal. Esto es fundamental para que el programa siga corriendo mientras se espera la respuesta del servidor

        - **Callbacks**: son funciones que se pasan como argumento a otras funciones y se ejecutan una vez que la tarea asíncrona este terminada.
        - **Promises**: permite gestionar una petición asíncrona, dado que tiene 3 estados: pendiente, cumplida y rechazada.
        - **Async/await:** con async se define una función como asíncrona para que este preparada a tener que ejecutar en un hilo secundario alguna petición, mientras que con await se le indica a JavaScript que debe esperar por la resolución de la petición mientras el programa se sigue ejecutando.

2. Evolución del Lenguaje JavaScript y el Estándar ECMAScript (2 puntos)

- Incluye en el informe un análisis sobre:

- **Lenguaje Interpretado vs. Compilado:**
    - **Lenguajes interpretados:**
        - JavaScript es interpretado, el código se ejecuta línea por línea en tiempo real, esto permite flexibilidad, pero puede ser más lento.
    - **Lenguajes compilados:**
        - El código fuente se compila a lenguaje máquina. Esto requiere un paso adicional respecto al interpretado, pero hace que programa se ejecute más rápido.
- **Evolución del Estándar ECMAScript:**
    - ES5(2009): introdujo archivos JSON nativos, strict mode y mejoras en manipulación de objetos
    - ES6(2015): Introdujo las variables let y const, módulos, clases, promeras y arrow function
    - ES9(2018): Mejoras en asincronía, promise .finally y operadores spread
- **JavaScript vs. ECMAScript:**
    - JavaScript está basado en el estándar ECMAScript, lo que significa que sigue este estándar en navegadores y entornos de node.js
    - ECMAScript define el comportamiento y funcionalidades del lenguaje, cada versión de ECMA mejora características del lenguaje. Como las mencionadas en el apartado anterior.

- **TypeScript y sus Características:**

TypeScript es un superset de JavaScript que

  - Añade tipado estático opcional, lo que facilita la detección de errores.
  - TypeScript compila a JavaScript lo que lo hace compatibles en los mismos entornos que JavaScript.
  - Su importancia se enfatiza en proyectos grandes al detectar errores en tiempo de desarrollo, mantener el código escalable y robusto.
  - Es una alternativa a JavaScript que mejora la productividad y calidad de código en proyectos de cualquier escala.

- **Ventajas y desventajas de TypeScript:**

  - **Ventajas:**
    - **Tipado estático:** permite reducir errores
    - **Mejor mantenibilidad:** gracias a que todo está definido mediante tipado estático.
    - **Incorpora todas las mejoras de ECMAScript.**

  - **Desventajas:**
    - **Requiere compilación adicional:** proceso de desarrollo puede ser más lento debido al paso extra.
    - **Curva de aprendizaje:** en proyectos pequeños se hace menos necesario y a su vez, hace que el aprendizaje sea más complejo.

3. Análisis de la Pertinencia de Integrar JavaScript Avanzado o TypeScript en el Proyecto (3 puntos)

Ventajas de utilizar JavaScript avanzado o TypeScript en el proyecto:

- **Darle interactividad al proyecto**, a través de funciones avanzadas que mejoren la experiencia del usuario, como validaciones de formularios, peticiones al servidor, manejo de eventos, entre muchos otros.

- **Uso de asincronismo** para mantener el código funcionando mientras se resuelven peticiones HTTP u otras peticiones que puedan demorarse.

- **Conexión del proyecto,** el uso de JavaScript avanzado o TypeScript permitirá conectar el proyecto con APIs externas o un servidor propio para manejar el backend y las bbdd.

- **Uso de módulos /librerías de JavaScript,** el uso de JavaScript permitirá incluir librerías externas para las mas diversas aplicaciones como sweetalert (generar alertas), axios (conectar con servidores), etc.

Desventajas de utilizar JavaScript avanzado o TypeScript en el proyecto:

- **Curva de aprendizaje:** Si el desarrollador no tiene experiencia previa en JavaScript, se complejizará y ralentizará el desarrollo del proyecto

- **Tiempo y tamaño de desarrollo del proyecto:** al incluir las funcionalidades que podrían agregarse usando JavaScript avanzado o TypeScript el tamaño del proyecto crecerá de manera importante, lo cual debe revisarse si esta alineado con los objetivos del proyecto.

- **Costo del proyecto:** Al incluir más complejidad asociada a la implementación de JavaScript avanzado o TypeScript se requerirán más recursos como desarrolladores, test, servidores, etc.

Conclusión: ¿Es recomendable incluir JavaScript avanzado o TypeScript en el proyecto? Justifica tu respuesta.

Es totalmente recomendable incluir JavaScript avanzado o TypeScript , dado que nos permitirá llevar el proyecto a un siguiente nivel, en el cual se pueda por ejemplo hacer lo siguiente:

- Gestionar formularios para hacerle consultas al Hospital
- Permitir registro y login de usuarios
- Creación de un CRUD con peticiones HTTP para crear reservas, borrar reservas, modificar reservas o revisar reservas.
- Integrar una Api en la cual se guarden los profesionales y su disponibilidad horario para poder hacer escalable el ingreso de nuevos profesionales al hospital.
- Integrar APIS externas para diversas funcionalidades, como el clima, etc.
- Incorporar librerías que permitan mostrar alertas con sweet-alert o gestionar horas como moment.js
- En funcionalidades más avanzadas podríamos incluir un chat con IA que permita dar soporte a usuarios.