**Semana 1 y 2: Conceptos Básicos de JavaScript**

**- Conceptos básicos:** JavaScript es un lenguaje de programación interpretado y dinámico utilizado principalmente en el desarrollo web. Es esencial para la creación de páginas web interactivas y dinámicas.

**- Tipos de datos:** Incluyen números, cadenas, booleanos, null, undefined, objetos y símbolos. Los tipos de datos permiten manipular y almacenar diferentes formas de información en JavaScript.

**- Variables y constantes**: Se declaran utilizando var, let y const. let y const tienen un alcance de bloque, mientras que var tiene un alcance de función.

**- Sintaxis:** El uso de punto y coma para finalizar declaraciones es opcional pero recomendado para evitar errores. Los bloques de código se delimitan con llaves {}.

**- Estructuras de control:** Incluyen if, else, switch, for, while, y do-while, las cuales permiten controlar el flujo del programa basado en condiciones y repeticiones.

**- Funciones:** Pueden ser declarativas (function nombre() {}) o expresiones (const nombre = function() {}). Las funciones flecha (const nombre = () => {}) tienen una sintaxis más concisa y no tienen su propio this.

**- Ejecución de eventos:** Se maneja mediante addEventListener, que permite asociar funciones a eventos específicos de elementos del DOM, como clics o movimientos del ratón.

**- Salidas:** Métodos como console.log() para depuración, alert() para mostrar mensajes emergentes, y manipulación del DOM ('Modelo de Objetos del Documento') para actualizar la interfaz de usuario.

**- DOM:** 'Modelo de Objetos del Documento' es una API de programación para documentos HTML, que permite que los programas pueden acceder y modificar el contenido, estructura y estilo.

**Semana 6: Node.js y TypeScript**

**- Node.js:** Es un entorno de ejecución de JavaScript en el lado del servidor, basado en el motor V8 de Google Chrome. Permite ejecutar JavaScript fuera del navegador.

**- Instalación:** Se realiza usando nvm (Node Version Manager) y npm (Node Package Manager), los cuales facilitan la instalación y gestión de versiones de Node.js y sus paquetes.

**- Ejecución desde la consola:** Se utiliza el comando node seguido del nombre del archivo JavaScript para ejecutarlo en la consola.

**- TypeScript:** Es un superconjunto de JavaScript que agrega tipos estáticos. Mejora la robustez del código y facilita el desarrollo de aplicaciones a gran escala. Se instala y configura mediante npm (npm install typescript).

**Semana 7, 8 y 9: Angular, Elementos y Servicios**

**- Angular**: Es un framework para desarrollar aplicaciones web de una sola página. Facilita la creación de aplicaciones robustas y mantenibles.

**- Instalación:** Usando Angular CLI (ng), que permite crear, desarrollar y mantener proyectos Angular de manera eficiente.

**- Creación de proyectos:** Se realiza con el comando ng new nombre-del-proyecto, que configura un nuevo proyecto Angular con una estructura inicial.

**- Directivas:** Son instrucciones en el DOM que Angular interpreta y transforma. Incluyen ngIf (condicionales), ngFor (iteraciones), ngClass y ngStyle (manipulación de clases y estilos).

**- Navegación**: Se implementa con RouterModule, que permite definir rutas y manejar la navegación entre diferentes vistas de la aplicación.

**- Componentes**: En Angular, los componentes son partes lógicas de una aplicación que representan un trozo de la pantalla. Son una combinación de una plantilla HTML, estilos CSS y lógica TypeScript.

**- Pipes:** Transforman los datos en las plantillas, permitiendo formatear y manipular la presentación de datos directamente en la vista.

**- Formularios:** Angular ofrece formularios reactivos y basados en plantillas (ReactiveFormsModule, FormsModule), facilitando la gestión y validación de formularios   
complejos.

**- Back-end en JSON:** Uso de json-server para simular un servidor REST y facilitar el desarrollo frontend con datos mock.

**- Métodos REST:** Incluyen GET, POST, PUT, DELETE para interactuar con APIs. Estos métodos permiten crear, leer, actualizar y eliminar recursos en un servidor.

**- Herramientas de prueba:** Postman se utiliza para probar APIs y verificar su funcionalidad mediante peticiones HTTP.

**- Servicios:** Implementan la lógica de negocio y el acceso a datos, utilizando HttpClient para consumir APIs REST. La inyección de dependencias facilita la gestión de servicios en Angular.

**Semana 11, 12, 13 y 14: React, Elementos, Hooks y Servicios**

**- React:** Es una biblioteca para construir interfaces de usuario, desarrollada por Facebook. Permite crear componentes reutilizables y gestionar el estado de la aplicación eficientemente.

**- Instalación:** Se realiza con create-react-app, que configura un entorno de desarrollo con todas las herramientas necesarias (npx create-react-app nombre-del-proyecto).

**- JSX y TSX:** Son extensiones de JavaScript que permiten escribir HTML dentro de archivos JavaScript (JSX) o TypeScript (TSX), facilitando la creación de componentes de interfaz de usuario.

**- Navegación:** Se gestiona con react-router-dom, que permite definir rutas y navegar entre diferentes vistas de la aplicación.

**- Componentes:** Pueden ser funcionales o de clase. Los componentes funcionales son más simples y, con los hooks, pueden manejar el estado y los efectos secundarios.

**- Formularios:** Se gestionan mediante el estado (useState) y validaciones, permitiendo una interacción dinámica con los usuarios.

**- State Hooks:** useState se utiliza para manejar el estado en componentes funcionales, permitiendo actualizar y gestionar el estado interno.

**- Context Hooks:** useContext se usa para acceder a contextos, proporcionando una forma de pasar datos globales a través de la aplicación sin necesidad de props.

**- Ref Hooks:** useRef permite crear referencias a elementos DOM y mantener valores persistentes entre renderizaciones sin causar nuevas renderizaciones.

**- Resource Hooks:** Librerías como SWR o React Query facilitan la gestión de datos asíncronos y el estado de las peticiones.

**- Fetch API:** Es el método nativo para realizar peticiones HTTP en JavaScript, permitiendo interactuar con APIs.

**- Axios:** Es una librería popular para realizar peticiones HTTP, ofreciendo características adicionales como la configuración global y el manejo de interceptores.

**Semana 16: Vue.js y Next.js**

**- Vue.js:** Es un framework progresivo para construir interfaces de usuario. Es flexible y fácil de integrar con otros proyectos o bibliotecas.

**- Next.js:** Es un framework de desarrollo web basado en React que permite crear aplicaciones web y sitios web estáticos.

**- Comparación:** Vue.js es más flexible y fácil de aprender, mientras que Next.js ofrece capacidades avanzadas de SSR para aplicaciones React.

¿Qué es un API?  
 API: Es una interfaz que permite que diferentes sistemas o aplicaciones se comuniquen entre sí, facilitando el intercambio de datos y funcionalidades.  
¿Qué son los hooks?

 **Hooks**: Son herramientas en React que nos ayudan a agregar funciones como recordar datos o manejar acciones en componentes simples