**Estructura de carpetas para el proyecto formativo.**

**Front end (Client):**

/public #Archivos públicos accesibles, como el index.html que es el punto de entrada de la aplicación. En public se almacenan archivos totalmente estáticos, hay que mantener lo netamente necesario aquí. También hay otros archivos como el favicon.ico (logo que aparece en la pestaña del navegador).

/src #código fuente del front-end

/assets #Imagenes, fuentes y archivos estáticos que son utilizados por los componentes.

/components #Componentes de React que son reutilizables a lo largo de la aplicación. Ejemplo: botones, formularios, tarjetas.

/pages #Representa las vistas completas de las rutas de la aplicación. Aquí irían las pantallas principales como “Login”, “Dashboard”, “ProductList”.

/context #React Contexto API. Se usa para el manejo de temas o estilos globales, autenticación y estado de usuarios, configuraciones globales.

/hooks #Custom Hooks. La idea principal de un custom hook es extraer lógica común que pueda ser compartida entre varios componentes, haciendo que el código sea más limpio y fácil de mantener.

/services #Conexiones con el backend (API requests usando axios)

/utils #Funciones auxiliarías que pueden ser utilizadas en varias partes de la aplicación. Las funciones suelen ser independientes del estado y la lógica de la UI o la lógica de negocio. En esencia son pequeñas piezas de código reutilizable que cumplen funciones comunes, como validación de datos, manipulación de fechas, formato de texto.

App.js #Componente principal del front-end, se manejan las rutas y la estructura general de la aplicación.

Index.js & Index.css #Punto de entrada del front-end y estilos globales que se aplican a toda la aplicación. También permite el modo estricto, que ayuda a seguir buenas prácticas durante el desarrollo.

package.json #Dependencias y scripts del cliente

.env #Variables de entorno del front-end, por ejemplo la url del api.

**Backend (server):**

/config #configuraciones globales, normalmente esta la conexión en la base de datos.

/controllers #Lógica de los controladores para las rutas de la API.

/middlewares #Middlewares personalizados (auth, manejo de errores, etc)

/models #Definiciones de modelos de la base de datos.  
  
/routes #Definición de rutas para ser accedidas desde la API.

/services #Logica de negocio.

/utils #Al igual que en el front end, se utiliza para almacenar funciones utilitarias para ser reutilizadas en la aplicación. (como formateo de datos, validaciones de formularios, respuestas de la api)

server.js #es el punto de entrada del servidor de express. Aquí se configura el servidor, rutas, middlewares y conexiones a la base de datos.

.env #almacena las variables de entorno del servidor. Incluye información sensible y de configuración, como credenciales de base de datos y tokens secretos.

package.json #almacena las dependencias.

**Migración BD (db):**

/migrations #Scripts que permiten crear y actualizar las tablas de la base de datos.

/seeds #Contiene scripts para insertar datos iniciales o de prueba en la base de datos. Esto es útil para tener datos precargados para pruebas o para la instalación inicial de la aplicación

schema.sql #Este archivo contiene la estructura de la base de datos, incluyendo las tablas, relaciones y restricciones. Es la manera de definir la base de datos de manera textual y puede ser utilizado para crear la base desde cero.