**Color Celeste: Lo que explicó Chaldu**

**Modelo Cliente Servidor Distribuido**

El modelo Cliente-Servidor-Distribuido es una arquitectura donde las tareas y responsabilidades se distribuyen entre distintos sistemas o nodos, permitiendo una interacción eficiente y escalable. En este contexto, el "cliente" se refiere a la interfaz de usuario, el "servidor" a la lógica de negocio y el "distribuido" indica que estos componentes pueden estar en diferentes ubicaciones, ya sea en la misma red o a través de Internet.

**Componentes Clave**

**Cliente (FrontEnd)**

* **Interfaz de Usuario (UI):** Implementada en HTML, CSS y JavaScript. La UI se presenta en el navegador del usuario, permitiendo la interacción con la aplicación.
* **Lógica de Cliente**: JavaScript maneja la interactividad, como responder a eventos (clics, formularios, etc.) y comunicarse con el servidor mediante llamadas a API.
* **Tiene su Dominio/Puerto**
* **Comunicación:** Para una API Rest, se comunica con el BackEnd utilizando peticiones HTTP.

**Servidor (BackEnd** En nuestro caso es JSON Server. Sirve solo en nuestra máquina. No se puede llevar a un hosting. Emulamos la funcionalidad del BackEnd**)**.

* **API (Interfaz de Programación de Aplicaciones):** El servidor expone una API, a menudo REST o GraphQL, que permite al cliente realizar operaciones como crear, leer, actualizar y eliminar datos.
* **Lógica de Negocio:** Aquí se procesan las solicitudes del cliente, aplicando reglas de negocio y realizando operaciones sobre la base de datos.
* **Gestión de Seguridad:** El servidor también maneja la autenticación y autorización, asegurando que los usuarios solo accedan a los datos permitidos.
* **Tiene su Dominio/Puerto**
* Procesa las solicitudes provenientes del FrontEnd y se comunica con la BD.
* **Comunicación:** Enviando respuestas a las peticiones que recibe.

**Base de Datos (Servidor de Datos)**.

* **Almacenamiento de Datos:** La base de datos puede ser SQL (como MySQL o PostgreSQL) o NoSQL (como MongoDB). Almacena la información que la aplicación utiliza, como usuarios, productos, o tareas.
* **Conexión con el Servidor:** El servidor se conecta a la base de datos para realizar operaciones, lo que implica realizar consultas y gestionar la persistencia de datos.
* **Tiene su Dominio/Puerto**
* **Almacenar Datos:** El BackEnd utiliza driver para comunicarse para hacer consultas, etc.
* **Comunicación:** La BD responde únicamente al BackEnd.

**Funcionamiento del Modelo**

**Flujo de Datos**

1. **Interacción del Usuario:** El usuario realiza una acción en la UI (por ejemplo, enviar un formulario).
2. Solicitud al Servidor: JavaScript envía una solicitud HTTP (GET, POST, PUT, DELETE) a un endpoint del servidor.
3. **Procesamiento en el Servidor**

* El servidor recibe la solicitud y la procesa.
* Puede validar los datos, aplicar lógica de negocio y realizar consultas a la base de datos.

1. **Respuesta del Servidor:** Una vez procesada la solicitud, el servidor envía una respuesta al cliente, generalmente en formato JSON.
2. **Actualización de la Interfaz:** El cliente recibe la respuesta y actualiza la UI según los datos proporcionados (por ejemplo, mostrando una lista de tareas).

**Relación y Flujo de Información**

1. **El FrontEnd** se comunica con el BackEnd mediante API Rest (get, post, path, delete). Nunca accede directamente a al BD ya que eso sería inseguro.
2. **El BackEnd** recibe las solicitudes del FrontEnd (lógica de negocio) y se comunica con la BD.
3. **La BD** responde al BackEnd, enviándole los datos que necesita o confirmando las operaciones realizadas.
4. Finalmente el BackEnd devuelve una respuesta al FrontEnd, la cual es mostrada al usuario.