1.5 contexto de desarrollo MERN

1.5 contexto de desarrollo MERN

Investigación de MERN

**¿Qué es MERN?**

MERN es un stack de desarrollo web que combina MongoDB, Express.js, React.js y Node.js para crear aplicaciones web full-stack con JavaScript en todas sus capas.

* **M** - MongoDB → Base de datos NoSQL
* **E** - Express.js → Framework backend para Node.js
* **R** - React.js → Biblioteca frontend para la UI
* **N** - Node.js → Entorno de ejecución en el servidor

Este stack es muy popular porque permite a los desarrolladores trabajar con un solo lenguaje (JavaScript) en el frontend, backend y base de datos.

1. Explicación de cada componente de MERN

* MongoDB (Base de Datos NoSQL): MongoDB es una base de datos NoSQL basada en documentos, lo que significa que almacena la información en JSON estructurado en lugar de tablas relacionales.
* Ventajas de MongoDB en MERN:
* Almacena datos en formato JSON (ideal para aplicaciones web).
* Alta escalabilidad y rendimiento.
* Fácil integración con Node.js mediante la librería Mongoose.
* Express.js (Framework Backend para Node.js); Express.js es un framework minimalista para Node.js que permite manejar rutas, peticiones y middleware de manera eficiente.
* Ventajas de Express.js:
* Facilita la creación de APIs RESTful.
* Manejo rápido y eficiente de rutas y middleware.
* Integración sencilla con MongoDB y React.js.
* React.js (Frontend - Interfaz de Usuario). React.js es una biblioteca de JavaScript creada por Facebook para construir interfaces de usuario dinámicas y reactivas.
* Ventajas de React.js en MERN:
* Componentes reutilizables y eficientes.
* Virtual DOM para mejorar el rendimiento.
* Manejo fácil del estado con React Hooks o Redux.
* Node.js (Entorno de Ejecución en el Servidor). Node.js es un entorno de ejecución que permite ejecutar JavaScript en el backend, utilizando un modelo asíncrono y no bloqueante.
* Ventajas de Node.js en MERN:
* Permite ejecutar JavaScript en el servidor.
* Alta escalabilidad y rendimiento con event-driven architecture.
* Compatible con Express.js y MongoDB.

1. Flujo de Trabajo en una Aplicación MERN

* El usuario interactúa con la interfaz creada en React.js.
* React envía una petición HTTP al backend Express.js (API).
* Express maneja la petición y accede a MongoDB si es necesario.
* MongoDB devuelve los datos y Express los envía a React.
* React muestra los datos en la interfaz.

1. Diferencias clave:

* MERN vs. MEAN: React usa componentes, mientras que Angular usa TypeScript y una estructura más rígida.
* MERN vs. LAMP: LAMP usa MySQL y PHP, mientras que MERN es completamente basado en JavaScript.

**Diferencias entre Backend y Frontend**

El **Frontend** y el **Backend** son dos partes esenciales en el desarrollo web, cada una con funciones y tecnologías específicas.

|  |  |  |
| --- | --- | --- |
| Aspecto | Frontend | Backend |
| Definición | Se encarga de la interfaz de usuario y la experiencia visual. | Se encarga de la lógica, el procesamiento de datos y la base de datos. |
| Rol | Lo que el usuario ve e interactúa. | Lo que sucede detrás de escena en el servidor. |
| Lenguajes | HTML, CSS, JavaScript | JavaScript (Node.js), Python, PHP, Ruby, Java, C# |
| Frameworks | React.js, Angular, Vue.js | Express.js, Django, Laravel, Spring Boot |
| Funcionalidad | Diseña la apariencia, botones, formularios y animaciones. | Gestiona usuarios, bases de datos, seguridad y lógica de negocio. |
| Ejemplo | Una página web con botones y formularios. | Procesar el inicio de sesión y validar credenciales en la base de datos. |
| Comunicación | Hace peticiones HTTP al backend para obtener o enviar datos. | Responde a las peticiones del frontend y maneja la base de datos. |