Laravel

¿Qué es Laravel?

Laravel es un marco de desarrollo de PHP que facilita la creación de aplicaciones web robustas y elegantes. Ofrece características como enrutamiento, manejo de bases de datos, autenticación y más.

Diferencia entre Eloquent ORM y Query Builder

Eloquent ORM es como una forma elegante de interactuar con tu base de datos utilizando objetos y relaciones, mientras que *Query Builder* te permite construir consultas SQL directamente. *Eloquent* es genial para trabajar con bases de datos de manera más orientada a objetos.

Inyección de Dependencias en Laravel

La inyección de dependencias es un concepto de diseño que ayuda a mantener tu código limpio y flexible. En Laravel, puedes inyectar dependencias automáticamente en tus clases y controladores para facilitar la prueba y la gestión de componentes externos.

Middleware en Laravel

Los Middleware son filtros que puedes aplicar a tus rutas para realizar tareas como autenticación, registro de actividad o manejo de sesiones antes de que lleguen a tus controladores. Son útiles para agregar capas de seguridad y funcionalidad a tu aplicación.

Autenticación de Passport en Laravel

Passport es un paquete de Laravel que facilita la autenticación a través de API. Es útil cuando estás construyendo aplicaciones móviles o de un solo página que necesitan autenticación de usuarios.

Flutter

¿Qué es Flutter?

Flutter es un marco de código abierto de Google para crear aplicaciones móviles nativas en iOS y Android. Lo mejor de Flutter es que puedes escribir una vez y ejecutar en ambas plataformas, lo que ahorra tiempo y esfuerzo.

StatefulWidget vs StatelessWidget

StatelessWidget es inmutable, mientras que *StatefulWidget* puede cambiar su estado durante su ciclo de vida. Usas *StatelessWidget* cuando los elementos de tu interfaz no cambian, y *StatefulWidget* cuando necesitas gestionar cambios dinámicos.

"Hot Reload" en Flutter

Hot Reload es una característica de Flutter que te permite ver los cambios en tu aplicación en tiempo real sin tener que reiniciarla. Facilita la experimentación y la corrección de errores de manera rápida.

Manejo de Estado en Flutter

Para aplicaciones grandes, puedes usar bibliotecas de manejo de estado como *Provider* o *Bloc* para mantener tus datos organizados y compartidos entre widgets.

Arquitectura de Flutter y Patrones de Diseño

Flutter sigue una arquitectura de "Widgets", donde la interfaz de usuario se construye utilizando widgets. Los patrones de diseño comunes incluyen *Provider*, *Bloc* y *MobX* para la gestión de estado.

Angular

¿Qué es Angular?

Angular es un marco de desarrollo de aplicaciones web creado por Google. Proporciona herramientas para construir aplicaciones web dinámicas y robustas. Angular se basa en *TypeScript*.

ngOnChanges vs ngOnInit

ngOnInit se llama una vez después de que se inicializa un componente, mientras que *ngOnChanges* se llama cuando los datos de entrada cambian. Puedes usar *ngOnChanges* para reaccionar a cambios en los datos de entrada.

Concepto de "binding" en Angular

El "binding" es una forma de conectar datos en tu aplicación con la interfaz de usuario. Los tipos de *binding* incluyen el *interpolation* ({{}}, [] y (), y [*ngModel*]).

Servicios en Angular

Los servicios en Angular son clases que proporcionan funcionalidades compartidas entre componentes. Se comunican con los componentes a través de inyección de dependencias.

Ventajas de Lazy Loading en Angular

Lazy Loading es útil para cargar módulos de la aplicación solo cuando se necesitan, lo que mejora la velocidad de carga inicial. Es especialmente útil en aplicaciones grandes con muchas rutas.

SQL

¿Qué es SQL?

SQL significa "Structured Query Language" y se utiliza para gestionar y consultar bases de datos relacionales. Es un lenguaje poderoso para interactuar con datos almacenados.

Diferencia entre INNER JOIN y LEFT JOIN

INNER JOIN devuelve solo filas que tienen coincidencias en ambas tablas, mientras que *LEFT JOIN* devuelve todas las filas de la tabla izquierda y las coincidencias de la tabla derecha, o valores NULL si no hay coincidencias.

Normalización de Bases de Datos

La normalización es un proceso para diseñar bases de datos de manera eficiente, reduciendo la redundancia y mejorando la integridad de los datos. Ayuda a evitar problemas de actualización y reduce el espacio de almacenamiento.

Transacciones en SQL

Las transacciones son operaciones de base de datos que deben ser ejecutadas en su totalidad o no en absoluto. Garantizan la consistencia de los datos y se utilizan para agrupar múltiples acciones en una sola unidad atómica.

Diferencia entre GROUP BY y HAVING en SQL

`GROUP BY` se utiliza para agrupar filas por un valor específico en una columna, mientras que `HAVING` se utiliza para filtrar grupos de resultados después de aplicar `GROUP BY`. Son útiles para resumir datos y aplicar condiciones a los grupos.