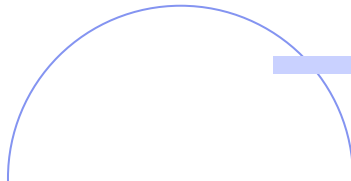


Java Backend

Glosario de términos



- **API:** Conjunto de definiciones y protocolos que permite la comunicación entre diferentes sistemas o componentes de *software*.
- **Arquitectura de Software:** Conjunto de decisiones estructurales y de diseño que definen la organización de un sistema, su comportamiento, y las interacciones entre sus componentes, para cumplir con los requisitos funcionales y no funcionales.
- **Backend:** Parte del desarrollo web que maneja la lógica del servidor, bases de datos y la integración de sistemas. Se encarga de procesar solicitudes, gestionar datos y ejecutar la lógica de la aplicación. Proporciona los datos necesarios al *frontend*.
- **Controller:** Componente en el patrón MVC (Modelo-Vista-Controlador) que maneja las solicitudes entrantes y coordina la lógica de la aplicación.



- **Dependency Injection:** Patrón de diseño utilizado para gestionar las dependencias de una aplicación. Facilita la creación de componentes desacoplados.
- **Frontend:** Parte del desarrollo web que se encarga de la interfaz de usuario y la experiencia del usuario. Incluye el diseño y la implementación de la apariencia y el comportamiento de la aplicación web en el navegador. Utiliza tecnologías como HTML, CSS y JavaScript.
- **gRPC (Google Remote Procedure Calls):** Framework de comunicación de alto rendimiento que permite la ejecución de procedimientos remotos en servicios distribuidos. Utiliza HTTP/2 para el transporte y *Protocol Buffers* como lenguaje de serialización. Ofrece eficiencia y soporte para múltiples lenguajes y características como el streaming bidireccional.
- **Identificador Uniforme de Recurso (URI):** Cadena de caracteres que identifica de manera única un recurso en la web o en una red. Proporciona un medio para acceder y referenciar dicho recurso, como una página web o un archivo.

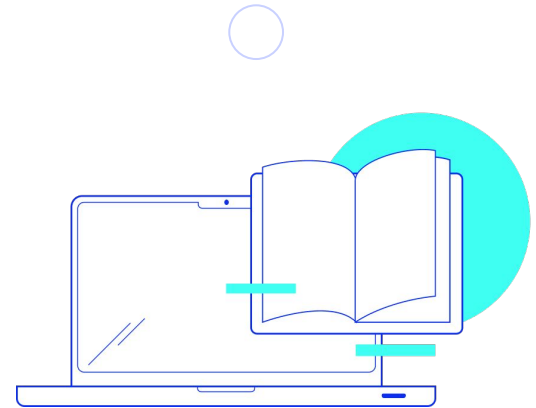


- **Java API for RESTful Web Services (JAX-RS):** Especificación para crear servicios web *RESTful* en Java, con anotaciones para simplificar el desarrollo. Desde la versión 1.1, es parte de Java EE 6 y su implementación de referencia es Jersey.
- **Jersey:** Implementación de referencia de la especificación JAX-RS para crear servicios web *RESTful* en Java. Proporciona una biblioteca y herramientas para simplificar el desarrollo y despliegue de servicios *RESTful*.
- **JSON (JavaScript Object Notation):** Formato de intercambio de datos ligero y fácil de leer, ampliamente utilizado para la transmisión de datos entre un servidor y una aplicación web.
- **JSON Web Token:** Es un estándar abierto, basado en JSON, para la creación de tokens de acceso que permiten la propagación de identidad y privilegios.
- **Microservicios:** Arquitectura que descompone una aplicación en servicios pequeños e independientes, cada uno ejecuta un proceso específico.

- **OpenAPI:** Especificación estándar para describir, producir, consumir y visualizar *APIs RESTful* de manera independiente de la plataforma. Permite a los desarrolladores definir la estructura de las *APIs*, facilita la documentación y la generación de código automáticamente para múltiples lenguajes y frameworks.
- **Patrones de APIs:** Estrategias para diseñar *APIs* efectivas, como *Versioning* (gestionar cambios), *Rate Limiting* (controlar el tráfico), *HATEOAS* (guiar con enlaces), y *Circuit Breaker* (manejar fallas).
- **Representational State Transfer (REST):** Estilo de arquitectura para sistemas distribuidos que utiliza protocolos web, como HTTP, para manejar recursos mediante una serie de operaciones estándar (GET, POST, PUT, DELETE). *REST* promueve la escalabilidad, simplicidad y la separación entre cliente y servidor. Permite la interacción a través de representaciones de recursos, como JSON o XML.
- **RESTful:** Es un término utilizado para describir servicios web que implementan el estilo arquitectónico *REST*.

- **Serializadores:** En el contexto de una API RESTful en Java, es un componente que convierte objetos Java (modelos de datos) en un formato que puede ser enviado a través de la red, típicamente JSON o XML, y viceversa.
- **Simple Object Access Protocol (SOAP):** Protocolo estándar para intercambiar información estructurada en servicios web, utilizando XML para definir el formato de los mensajes y HTTP o SMTP para la transmisión. SOAP permite la comunicación entre aplicaciones en diferentes plataformas y lenguajes.
- **Spring Boot:** Es una herramienta que simplifica aún más el desarrollo de aplicaciones basadas en el ya popular *framework* Spring Core. Spring Boot busca que el desarrollador solo se centre en el desarrollo de la solución, olvidándose por completo de la compleja configuración de un proyecto Java.
- **Swagger:** Herramienta que facilita el diseño, documentación y consumo de *APIs RESTful*. Permite a los desarrolladores generar documentación interactiva y probar *endpoints*.

- **Web Service:** Es un conjunto de protocolos y estándares que permiten la comunicación entre aplicaciones a través de la web (Internet).
- **XML:** Lenguaje de marcado extensible diseñado para almacenar y transportar datos de manera estructurada. XML permite definir etiquetas personalizadas y organizar datos en un formato legible por humanos y máquinas.



**Ahora sí,
¡comencemos!**