



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIOS WEB

REST, POST, PUT, DELETE Y ALL

**Arciniega Cristian &
García Oliver**

Responsables



Definición. -

Tecnología que permite que diferentes aplicaciones se comuniquen e intercambien información a través de internet, utilizando un conjunto de protocolos y estándares.

Forma de compartir información y funcionalidades a través de internet. Es como tener un montón de pequeñas aplicaciones trabajando juntas para hacer tu vida más fácil.



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIO WEB: REST

REPRESENTATIONAL STATE TRANSFER / TRANSFERENCIA DE ESTADO REPRESENTACIONAL



Definición. -

Servicio de arquitectura de software que proporciona estándares entre sistemas informáticos para establecer como se van a comunicar entre sí.

Es una forma sencilla y efectiva de crear servicios web que permiten a diferentes aplicaciones comunicarse entre sí. Es como un lenguaje común que facilita la comunicación e integración de sistemas.



Características principales. -

- 1. Protocolo de comunicación:** HTTP es el protocolo de comunicación utilizado en el servicio REST.
- 2. Arquitectura:** Hace uso de la arquitectura cliente - servidor.
- 3. Stateless (sin estado):** Cada petición del cliente contiene toda la información necesaria para que el servidor la procese y el servidor no necesita recordar información de solicitudes anteriores para atender una nueva.

4. Caché: En cada petición el cliente debe indicar si el resultado puede ser o no cacheado por medio de ('Cache-control', en la cabecera de la petición HTTP).

5. Sistema por capas: Permite agregar capas intermedias (proxies, firewalls) para mejorar la seguridad, la fiabilidad y la escalabilidad.



Ventajas y Desventajas. -

Ventajas	Desventajas
Sencillez: Fácil de entender y de implementar, lo que reduce la curva de aprendizaje.	Sobrecarga de datos: A veces se pueden solicitar más datos de los necesarios, lo que puede afectar el rendimiento.
Flexibilidad: Se adapta a diferentes tipos de aplicaciones y tecnologías.	Versionamiento: Gestionar cambios en la API puede ser complejo y afectar a los clientes existentes.
Escalabilidad: Puede manejar grandes volúmenes de tráfico y crecer a medida que aumenta la demanda.	Múltiples peticiones: Para obtener información completa, a veces se requieren varias peticiones, lo que puede aumentar la latencia.
Independencia de la plataforma: No está ligado a un lenguaje de programación o plataforma en particular.	Falta de estandarización estricta: Aunque hay estándares, la implementación puede variar entre diferentes servicios.
Cacheable: Mejora el rendimiento y la escalabilidad al permitir almacenar respuestas a solicitudes frecuentes.	Potencial para errores 404: Si las URLs no están bien diseñadas, pueden generar errores 404.
Ampliamente adoptado: Es el estándar de facto para servicios web, lo que facilita encontrar recursos y comunidades.	



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIO WEB: POST



Definición. -

Método HTTP que se usa para enviar datos a un servidor y crear una nueva entidad. Es uno de los métodos HTTP más comunes y se utiliza para enviar datos de forma más discreta que en la URL.



Características principales. -

1. Seguridad: Es discreto, ya que los datos no se muestran en el caché ni en el historial de navegación.

2. Flexibilidad: permite el envío de contenido multimedia (se pueden enviar textos cortos, fotos o videos).

3. Recursos: Si se realiza la misma petición POST varias veces, se crearán múltiples recursos.

4. Respuesta del servidor: El servidor procesa la solicitud y devuelve una respuesta, que puede ser un mensaje de confirmación, una redirección o un error.



Ventajas y Desventajas. -

Ventaja	Desventaja
Flexibilidad: Permite enviar datos de cualquier tipo y tamaño en el cuerpo de la solicitud, lo que lo hace muy versátil para crear nuevos recursos o realizar acciones complejas.	No idempotente: Si se realiza la misma petición POST varias veces, se crearán múltiples recursos idénticos, lo que puede llevar a duplicados en la base de datos.
Personalizable: Se pueden enviar datos estructurados (JSON, XML) o no estructurados (contenido multimedia) para crear recursos personalizados.	Menos seguro: Al crear nuevos recursos, se requiere una mayor atención a la validación y sanitización de los datos para prevenir ataques de inyección.
Común: Es el método HTTP más utilizado para enviar datos a un servidor, lo que facilita su comprensión y uso.	Puede ser menos eficiente: Para operaciones simples como actualizar un campo específico, puede ser más eficiente utilizar PUT.
Acciones: Puede utilizarse para desencadenar acciones en el servidor, como enviar un correo electrónico o procesar un pago, más allá de simplemente crear un recurso.	



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIO WEB: PUT



Definición. -

Método HTTP utilizado para actualizar o reemplazar un recurso existente en el servidor con los datos proporcionados.

Características principales. -

1. Idempotente: Se puede llamar varias veces sin que haya efectos secundarios, siendo útil para construir una API tolerante a fallos.

2. Técnica: PUT envía una entidad al servidor para actualizar un recurso completo identificado por la URI proporcionada.

3. Recursos: Reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

URN

URL

<https://odiseageek.es/post/2187#comments>

Protocolo

Indica cómo acceder al recurso

Recurso

No es obligatorio que aparezca siempre

URI

Fuente: (Odyssey, 2021)

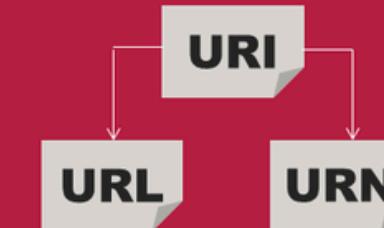
URI, URN, URL

¿qué son?



URI ("Uniform Resource Identifier")
Secuencia de caracteres que permite identificar a un recurso.

Puede usarse como localizador del recurso, como nombre o como ambos.



URL ("Uniform Resource Locator")
Es un URI que permite localizar a un recurso. Incluye un protocolo.

<http://www.misitioweb.com>
<mailto:direccion@dominio.com>

ejemplos



URN ("Uniform Resource Name")
Es un URI que provee un nombre para un recurso.

<urn:uuid:6e8bc430-9c3a-11d9-9669-0800200c9a66>
<urn:ISSN:0167-6423>



Ventajas y Desventajas. -

Ventaja	Desventaja
Idempotencia: Realizar múltiples solicitudes PUT con los mismos datos tendrá el mismo resultado que realizarla una sola vez. Esto es ideal para reintentos en caso de fallos de red.	Puede ser menos eficiente: Si solo deseas actualizar una pequeña parte de un recurso, enviar todo el recurso nuevamente puede ser ineficiente.
Claro y conciso: La intención de reemplazar completamente un recurso es evidente.	Requiere enviar todos los datos: Incluso si solo deseas modificar un campo, debes enviar todos los datos del recurso en la solicitud.
Seguro para actualizaciones repetidas: Al ser idempotente, puedes reintentar una solicitud PUT sin preocuparte de crear duplicados o datos inconsistentes.	Puede ser más complejo: En algunos casos, puede ser más complejo de implementar que POST, especialmente cuando se trata de actualizaciones parciales (campos específicos).
Ideal para actualizaciones completas: Es perfecto para reemplazar completamente un recurso existente con nuevos datos.	



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIO WEB: DELETE



Definición. -

Método HTTP utilizado para eliminar un recurso específico identificado por una URI.

DELETE solicita al servidor que elimine el recurso identificado. Si el recurso no existe, el servidor puede devolver un estado de éxito, indicando que ya no está disponible.



Características principales. -

- 1. Idempotente:** Realizar la misma petición DELETE múltiples veces tiene el mismo efecto que realizarla una sola vez.
- 2. Body:** La información necesaria para identificar el recurso a eliminar se incluye en la URL.
- 3. Base de datos:** Usar solo en caso necesarios ya que puede llegar a eliminar datos importantes que se encuentren registrados en nuestro servidor de base de datos.

Códigos de respuesta. -

- 202 (Accepted):** Si la acción ha sido exitosa pero aún no se ha ejecutado.
- 204 (No Content):** Si la acción se ha ejecutado y no se debe proporcionar más información.
- 200 (OK):** Si la acción se ha ejecutado y el mensaje de respuesta incluye una representación que describe el estado.



Ventajas y Desventajas. -

Ventaja	Desventaja
Simple y directo: Su función es clara y concisa: eliminar un recurso.	Irreversible: Una vez eliminado un recurso, generalmente no se puede recuperar a menos que se tenga un mecanismo de respaldo implementado.
Idempotente: Realizar múltiples solicitudes DELETE para el mismo recurso tiene el mismo efecto que realizarla una sola vez. Esto es útil para reintentos en caso de fallos.	Puede tener consecuencias inesperadas: Si un recurso eliminado está relacionado con otros recursos, su eliminación puede causar efectos colaterales no deseados si no se gestionan las dependencias correctamente.
Seguro para eliminaciones repetidas: Al ser idempotente, puedes enviar múltiples solicitudes DELETE sin preocuparte de eliminar el recurso accidentalmente varias veces.	Requiere una cuidadosa consideración de las dependencias: Es importante evaluar las relaciones entre los recursos antes de eliminar uno para evitar problemas de integridad de los datos.
Fácil de entender: Su semántica es intuitiva y fácil de comprender para los desarrolladores.	



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

SERVICIO WEB: ALL



Definición. -

Método utilizado en algunos frameworks para manejar todas las solicitudes HTTP (GET, POST, PUT, DELETE, etc.) a una URI específica.



Características principales. -

- 1. Flexibilidad:** Evita la duplicidad de lógica en una misma identidad de ruta.

- 2. Lógica:** Si no se gestiona correctamente, puede dificultar la separación de responsabilidades entre métodos HTTP, llevando a confusiones o errores en el manejo de solicitudes.



Ventajas y Desventajas. -

Ventaja	Desventaja
Flexibilidad: Permite manejar múltiples métodos HTTP con una sola función, simplificando el manejo de rutas.	Ambigüedad: Puede generar falta de claridad si no se separan correctamente las lógicas específicas de cada método.
Duplicidad: Evita duplicar lógica común a varios métodos HTTP.	Depuración: Difículta la depuración y el mantenimiento si la lógica de métodos es muy compleja.
	Compatibilidad: Es dependiente de frameworks que soporten esta funcionalidad, no es un método HTTP estándar.
	Desarrollo: No es recomendable para producción, ya que puede afectar la claridad y organización del código.



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

¿Cuándo usar cada
servicio?



Servicio	Cuando usarlo	Casos de uso
REST	<ul style="list-style-type: none">- Cuando necesitas construir una API que siga principios arquitectónicos estándar y sea escalable, sencilla y eficiente.	<ul style="list-style-type: none">- API de terceros como una pasarela de pagos (ejemplo: PayPal REST API) o integración con servicios como Google Drive o AWS.
POST	<ul style="list-style-type: none">- Cuando necesitas crear un recurso nuevo en el servidor.- Para enviar datos al servidor que no tienen una representación existente o única.	<ul style="list-style-type: none">- Crear un nuevo usuario en una base de datos: /users con los datos en el cuerpo de la solicitud.- Subir un archivo o registrar un nuevo producto.-
PUT	<ul style="list-style-type: none">- Cuando necesitas actualizar o reemplazar completamente un recurso existente.- En algunos casos, para crear un recurso si no existe (cuando se permite esta funcionalidad).	<ul style="list-style-type: none">- Actualizar los datos de un usuario con ID 123: /users/123 con los datos nuevos en el cuerpo de la solicitud.- Subir una configuración inicial a un servidor, garantizando que sea única.
DELETE	<ul style="list-style-type: none">- Cuando necesitas eliminar un recurso específico identificado por una URI.- Usar cuando quieras liberar espacio o limpiar datos antiguos en el sistema.	<ul style="list-style-type: none">- Eliminar un producto con ID 456: /products/456.- Eliminar comentarios ofensivos o caducados en una aplicación.
ALL	<ul style="list-style-type: none">- Durante el desarrollo o pruebas, para manejar múltiples métodos HTTP con una lógica común en un punto de entrada. <p>Cuando necesitas un manejador temporal antes de definir la lógica específica de cada método.</p>	<ul style="list-style-type: none">- Crear una función genérica para manejar solicitudes a /recurso sin importar si es GET, POST, PUT o DELETE. <p>Implementar una API base para pruebas iniciales.</p>



Referencias Bibliográficas

Amazon.com. (s. f.). Amazon.com. Recuperado 5 de enero de 2025, de <https://aws.amazon.com/es/what-is/restful-api/#:~:text=Interfaz%20uniforme&text=Ella%20indica%20que%20el%20servidor,en%20la%20aplicaci%C3%B3n%20del%20servidor>.

Arquitectura REST. (s. f.). Scribd. Recuperado 5 de enero de 2025, de <https://es.scribd.com/presentation/460447292/Arquitectura-REST-pptx>

¿Cuál es la diferencia entre los métodos de http como PUT, POST, DELETE, HEAD, GET....? (s. f.). Stack Overflow en español. Recuperado 5 de enero de 2025, de <https://es.stackoverflow.com/questions/370130/cu%C3%A1l-es-la-diferencia-entre-los-m%C3%A9todos-de-http-como-put-post-delete-head-g>

DELETE. (s. f.). MDN Web Docs. Recuperado 5 de enero de 2025, de <https://developer.mozilla.org/es/docs/Web/HTTP/Methods/DELETE>

En qué se diferencia la URL, URI y URN. (2021, junio 18). Odisea Geek. <https://odiseageek.es/posts/en-que-se-diferencia-la-url-uri-y-urn/>

Julio Pari (IT Architect IBM). (s. f.). Diferencia entre PUT y POST en un Web Service REST en Java. Arquitecturaibm.com. Recuperado 5 de enero de 2025, de <https://arquitecturaibm.com/diferencia-entre-put-y-post-en-un-web-service-rest-en-java/>

Netapp.com. (s. f.). Netapp.com. Recuperado 5 de enero de 2025, de <https://www.netapp.com/es/data-storage/unstructured-data/what-is-unstructured-data/>

(PPT) Rest. (s. f.). Scribd. Recuperado 5 de enero de 2025, de <https://es.scribd.com/document/637141270/PPT-REST>

PUT - HTTP. (s. f.). MDN Web Docs. Recuperado 5 de enero de 2025, de <https://developer.mozilla.org/es/docs/Web/HTTP/Methods/PUT>

Surra, B. (2023, julio 11). Qué es REST: definición, utilidades y ventajas. MyTaskPanel Consulting. <https://www.mytaskpanel.com/que-es-rest/>



INSTITUTO SUPERIOR TECNOLÓGICO
17 DE JULIO

TECNOLOGÍA SUPERIOR EN
DESARROLLO DE SOFTWARE

AGRADECemos
SU ATENCIÓN

Urcuquí - Yachay

06-01-2025