

JavaScript Avanzado

Sesión 9



Universidad
Tecnológica
del Perú

¿Tienen alguna consulta o duda sobre la clase previa?

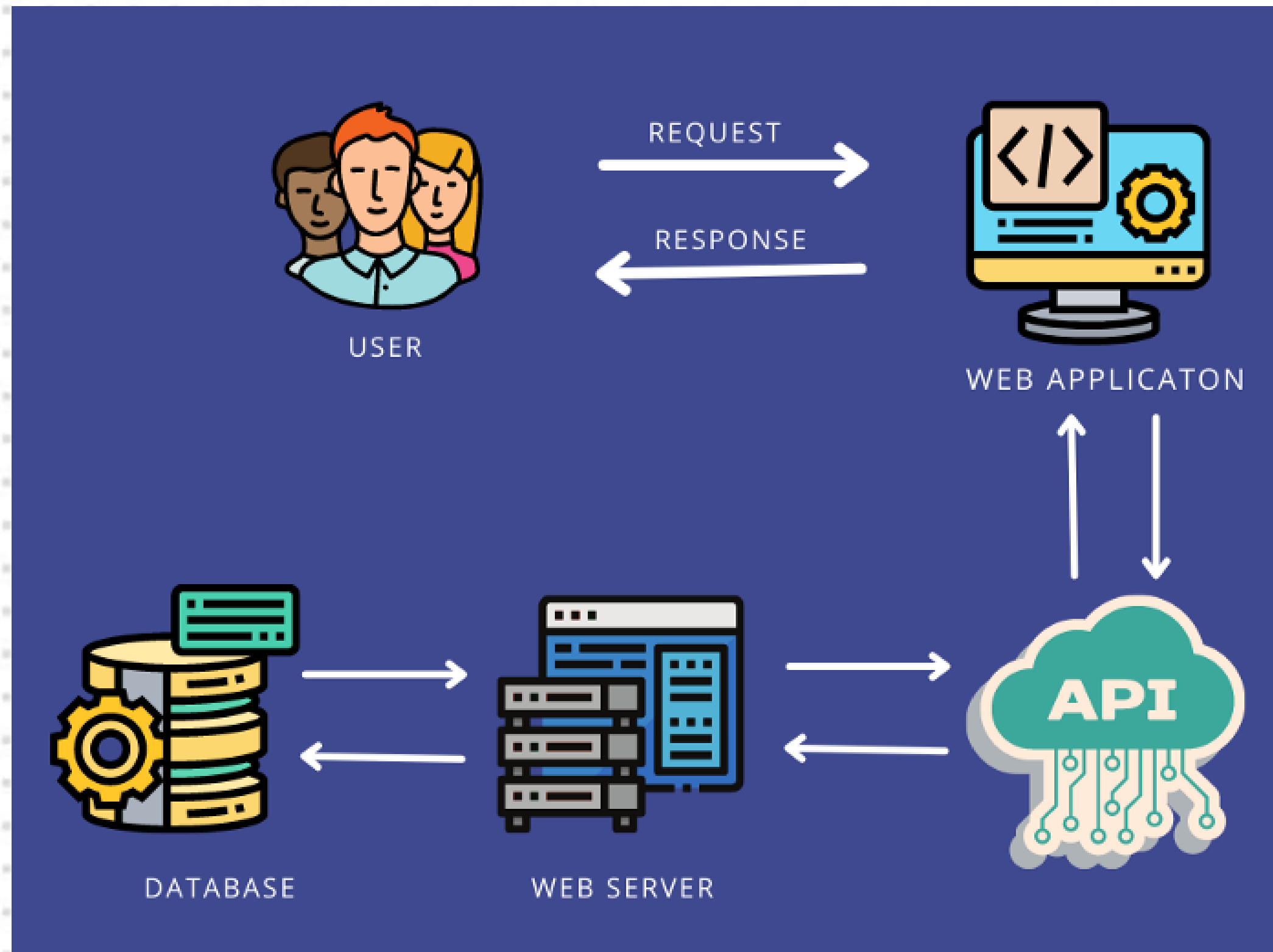


Logro de la sesión

Al finalizar la sesión, el estudiante crea aplicaciones web con el Framework Angular empleando servicios para consumir una API REST.

¿Qué es un método REST? ¿Cómo se consumen con Angular?





¿Cuál es la importancia de las API en el contexto cliente servidor?



Contenido

Servicios en Angular

- Creación de un Back-end en JSON
- Métodos REST
- Herramientas de prueba
- Creación de servicios e implementación una aplicación web consumiendo servicios REST



Creación de un Back-end en JSON

json-server

- Es una herramienta muy útil para desarrolladores que permite crear una API REST completa simulada usando un archivo JSON como base de datos.
- Es ideal para el desarrollo rápido y las pruebas de aplicaciones front-end, ya que proporciona endpoints **HTTP** sin necesidad de configurar un servidor backend complejo.



Ventajas del simulador json-server

- Permite crear una API REST simulada rápidamente a partir de un archivo JSON.
- Cada propiedad en el objeto JSON se convierte en un recurso que se puede consultar, actualizar, eliminar, etc., mediante las peticiones HTTP estándar (GET, POST, PUT, DELETE).
- Es ideal para pruebas y desarrollo de frontend, ya que se puede crear un backend simulado sin necesidad de escribir código en el servidor.

Instalación de "json-server"

1. Instalar Node.js y npm (Node Package Manager) en el sistema.
2. Instalar "json-server" con el terminal o consola y ejecutando el comando:
 - `npm install -g json-server`





Nota: "-g" instalará "json-server" de forma global para que puedas usarlo desde cualquier ubicación en tu sistema.



EXPLORER

OPEN EDITORS

✕ {} movies.json api

UNT...    

myapp

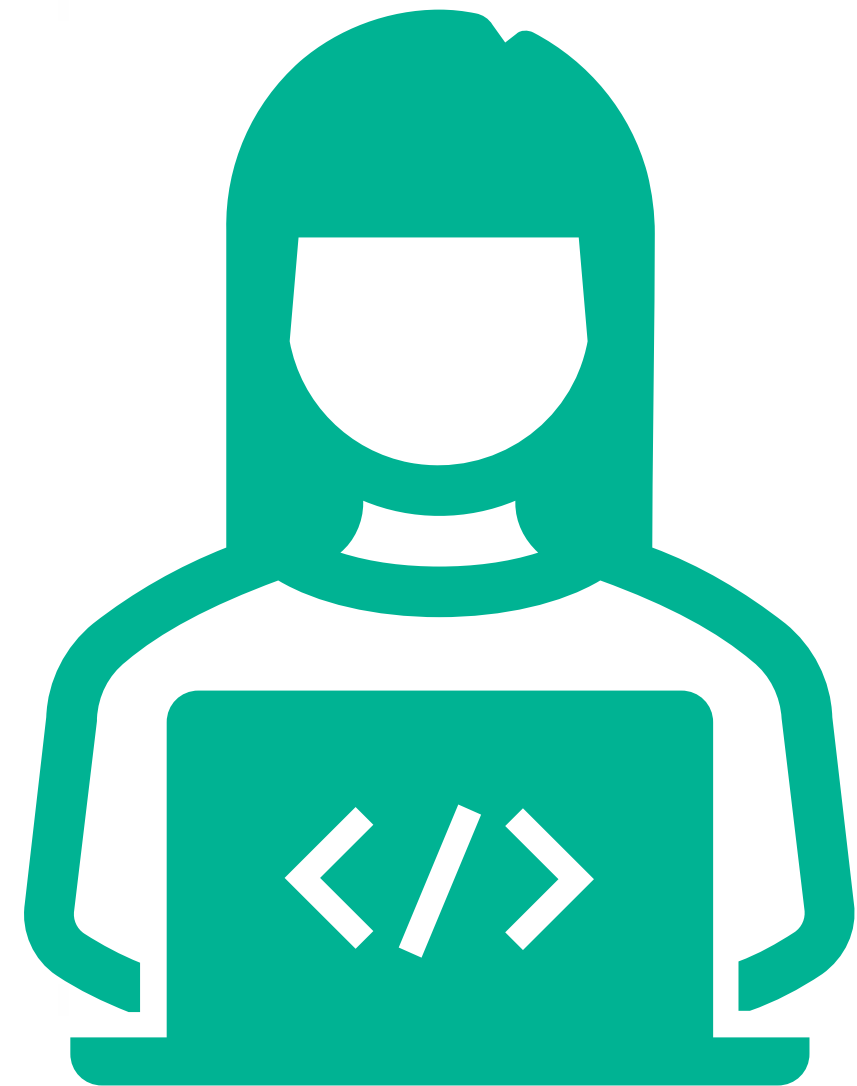
- api
 - {} movies.json
- node_modules
- JS index.js
- {} package.json

{} movies.json ✕

```
1  {
2    "movies": [
3      {
4        "id": 1,
5        "title" : "Meru",
6        "year" : 2015,
7        "Director" : "Chin / Vasarhelyi"
8      },
9      {
10       "id": 2,
11       "title" : "Star Wars",
12       "year" : 1977,
13       "Director" : "George Lucas"
14     }
15   ]
16 }
```

Configurando

- Instalar, configurar y ejecutar un **servidor JSON** mediante **Node.js**.



Métodos REST (transferencia de estado representacional)

- Son un conjunto de principios de diseño para **crear servicios web** que enfatizan la escalabilidad, el rendimiento y la independencia del cliente y el servidor.
- Una **API REST** utiliza métodos **HTTP** estándar como **GET**, **POST**, **PUT** y **DELETE** para operaciones de lectura, creación, actualización y eliminación de recursos.
- Cada operación en una **API REST** es **stateless**, lo que significa que no guarda ningún estado entre peticiones. Esto permite que las aplicaciones sean más escalables y fáciles de mantener.

Métodos REST



Herramientas de prueba

Postman

- Ampliamente utilizada para pruebas de API RESTful, permite realizar pruebas sin escribir código y admite pruebas automatizadas.



POSTMAN

SoapUI

- Soporta servicios REST y SOAP, ofreciendo pruebas completas en funcionalidad, carga y seguridad.



Universidad
Tecnológica
del Perú

Herramientas de prueba

JMeter

- Herramienta de código abierto para pruebas de carga y funcionales, permite simular múltiples usuarios y verificar el comportamiento bajo carga pesada.

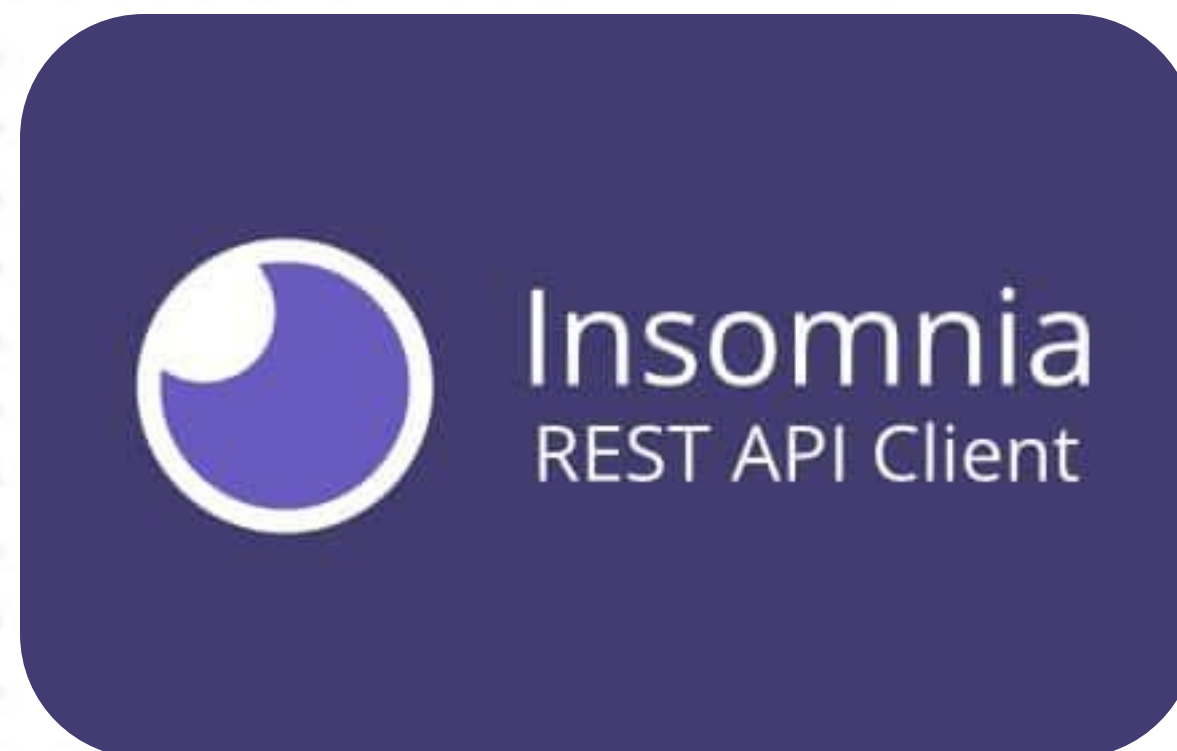
ReadyAPI

- Proporciona pruebas funcionales, de rendimiento, de seguridad y de virtualización para RESTful, SOAP, GraphQL y otros servicios web.

Herramientas de prueba

Insomnia

- Herramienta que permite realizar pruebas para facilitar la creación, el envío y la depuración de solicitudes HTTP y API



Servicios en Angular

- Son clases que se pueden inyectar en otros componentes y servicios, y que permiten separar los datos y funciones de la aplicación.
- Los servicios llevan el decorador **@Injectable** de Angular, que indica que pueden inyectar otras dependencias.

Servicios en Angular

```
import {Injectable} from '@angular/core';

@Injectable({
  providedIn: 'root',
})
export class CalculatorService {
  add(x: number, y: number) {
    return x + y;
  }
}
```

```
import { Injectable } from '@angular/core';

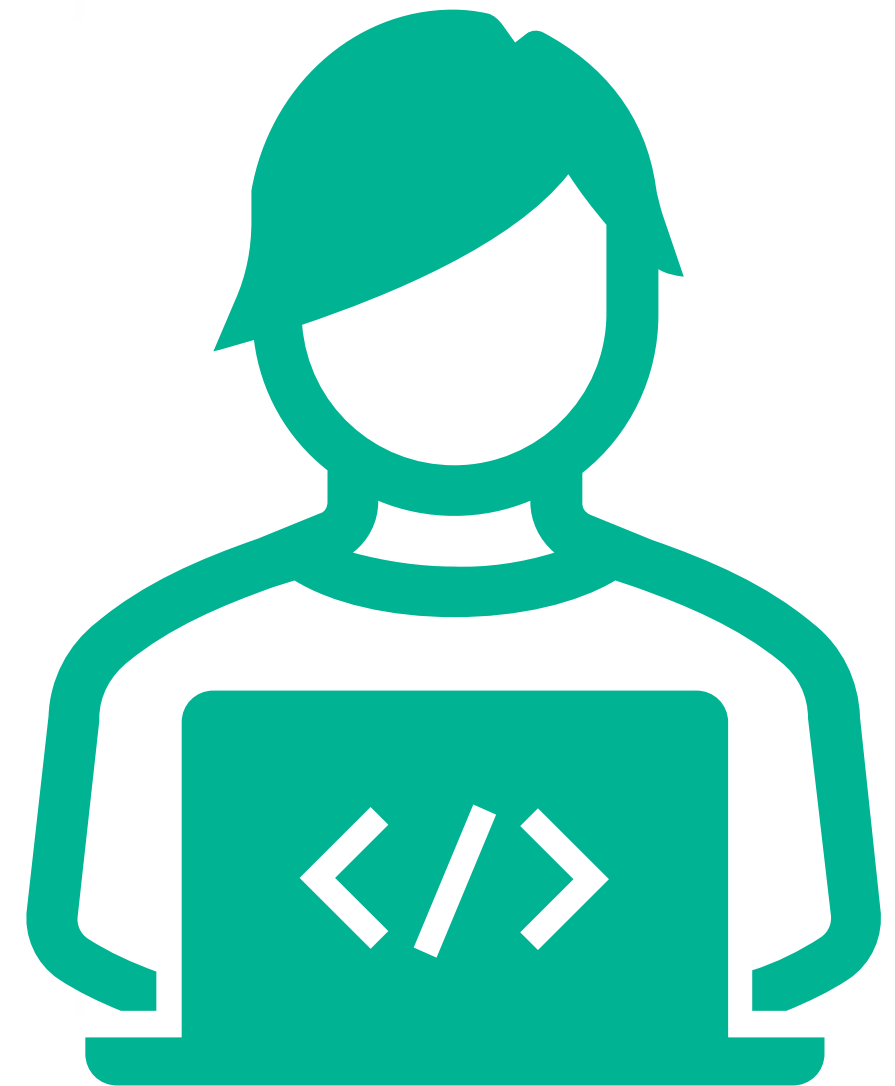
@Injectable({
  providedIn: 'root',
})
export class HeroService {

  constructor() { }

}
```

Programando

- Construir un proyecto en Angular que emplee **servicios** para consumir una **API REST** de un **servidor JSON**.



¿Tienen alguna consulta o duda?



Actividad



Resolver la actividad planteada en la plataforma.

Cierre

¿Qué hemos aprendido hoy?



Elaboramos nuestras conclusiones sobre el tema tratado



**Universidad
Tecnológica
del Perú**