



# Hola Bienvenido!

Registra tu asistencia

<https://forms.gle/6hfdT5U5v5fF34RQ8>



# **Vue JS - Frontend**

# **Express JS - Backend**

# ¿Qué es una Vue js?



Vue te permite crear aplicaciones web dinámicas dividiendo tu interfaz en componentes reutilizables, con datos que reaccionan automáticamente cuando cambian.

- **Reactividad:**

Si cambias un dato en tu aplicación, la interfaz se actualiza automáticamente sin recargar la página.

- **Componentes:**

La aplicación se divide en piezas independientes (componentes), cada una con su **HTML (template)**, **CSS (estilos)** y **lógica (script)**.



- **Unidireccionalidad de datos:**

Los datos fluyen del **padre** → **hijo** (de componente en componente) de manera controlada.

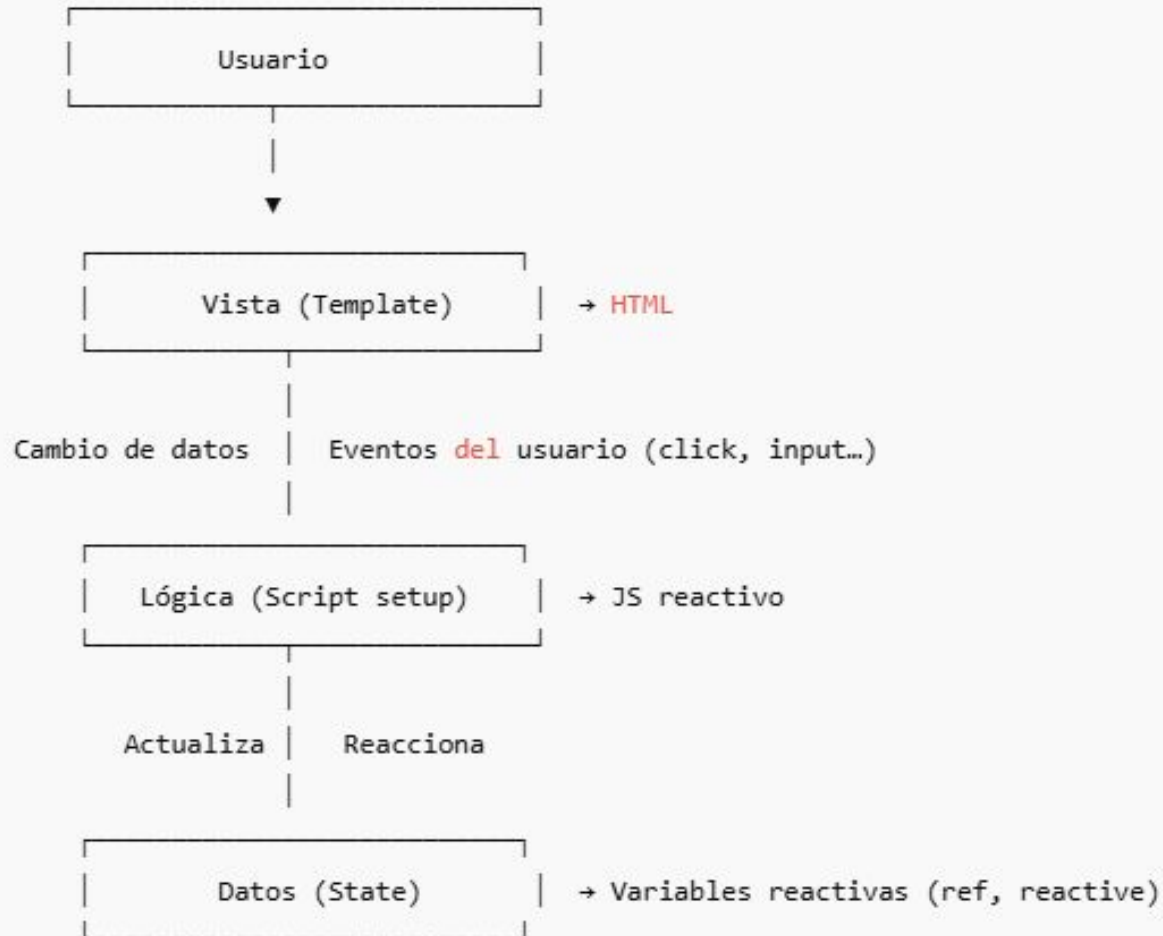
- **Virtual DOM:**

Vue actualiza solo las partes necesarias del DOM, sin volver a renderizar toda la página.

Esto mejora mucho el rendimiento.



# Flujo de VUE JS



`main.js`



Crea la app y monta el componente raíz (`App.vue`)



`App.vue`



Contiene varios componentes hijos (`Header.vue`, `Translator.vue`, `Footer.vue`)



Cada componente maneja sus datos y comunica eventos al padre



El estado reactivo actualiza automáticamente la interfaz



# Arquitectura a gran escala

src/

- assets/ → Imágenes, íconos, estilos globales
- components/ → Componentes reutilizables (botones, tarjetas)
- views/ → Páginas completas (Home, Login, Dashboard)
- router/ → Configuración de rutas (vue-router)
- store/ → Estado global (Pinia o Vuex)
- App.vue → Componente raíz
- main.js → Punto de entrada

# Instalar Vue



## 1. Haber instalado previamente Node js

En esta ocasión se utilizará vite, pero existen alternativas, tales como:

- VUE CLI ( Webpack)
- NUXT JS
- VUE CDN



Instanciar VITE como herramienta de creación de proyectos frontend



```
npm create vite@latest nombre-proyecto --  
--template vue
```

```
cd nombre-proyecto
```

```
npm install
```

```
npm run dev
```



# Flujo de trabajo en VUE JS

## Flujo básico de trabajo

1. Crear el proyecto (con Vite o CLI).
2. Instalar dependencias (`npm install`).
3. Ejecutar servidor (`npm run dev`).
4. Editar componentes en `src/`.
5. Compilar para producción (`npm run build`).




# Proyecto VUE JS + EXPRESS JS Traductor web




# API - GOOGLE TRANSLATE

<https://translate.googleapis.com>



# Definimos estructura base cliente- servidor:

```
translate-app/  
|  
├─ backend/                # Servidor con Express.js  
|   ├─ package.json  
|   └─ server.js  
|  
└─ frontend/               # Aplicación Vue.js  
    ├─ package.json  
    ├─ vite.config.js  
    └─ src/  
        ├─ App.vue  
        ├─ main.js  
        └─ components/  
            └─ Translator.vue
```



# Qué es Package JSON?

El archivo **package.json** es el “cerebro” del proyecto Node.js.

Sección	Descripción	
<code>"name"</code>	Nombre del proyecto (puede ser cualquiera).	
<code>"version"</code>	Versión del proyecto (convención semántica).	
<code>"description"</code>	Breve descripción del propósito del proyecto.	
<code>"main"</code>	Archivo principal que arranca el servidor (normalmente <code>server.js</code> ).	
<code>"type": "module"</code>	Indica que usas sintaxis moderna <code>import/export</code> (en lugar de <code>require</code> ).	
<code>"scripts"</code>	Comandos personalizados para ejecutar el proyecto ( <code>npm start</code> correrá <code>node server.js</code> ).	
<code>"dependencies"</code>	Librerías externas que el proyecto necesita para funcionar (Express, Axios, CORS, etc.).	

En el **package.json** no se ejecuta código, solo se declaran configuraciones y dependencias.  
Es el archivo que le dice a Node.js qué instalar y cómo ejecutar el proyecto.

# Qué se instancia en Server.js ?

El archivo `server.js` es el **punto de entrada del backend**, donde realmente se **instancian los módulos** (Express, Axios, CORS) y se **configura el servidor web**.

Elemento	Qué hace	Tipo
<code>express()</code>	Crea la instancia del servidor web.	 Clase principal
<code>app.use(cors())</code>	Instancia el middleware de CORS para permitir peticiones externas.	 Configuración
<code>app.use(express.json())</code>	Instancia el middleware para parsear JSON del cuerpo de las peticiones.	 Configuración
<code>axios.get(...)</code>	Instancia una petición HTTP a la API de Google.	 Cliente externo
<code>app.listen(3000, ...)</code>	Instancia el servidor HTTP que queda escuchando en un puerto.	 Ejecución



# Publicar proyecto en la nube

## Separar frontend y backend (más común y profesional)

- **Frontend (Vue)** → Se sube como sitio estático (HTML, JS, CSS).
- **Backend (Express)** → Se despliega como API en un servidor Node.js.



# Publicar proyecto en la nube Gratuitamente

Parte	Servicio	Qué hace	Ventajas
Frontend	<a href="#">Vercel</a> o <a href="#">Netlify</a>	Alojan sitios estáticos (como tu app Vue compilada)	Fácil despliegue desde GitHub. Gratis y rápido.
Backend	<a href="#">Render</a> , <a href="#">Railway</a> , o <a href="#">Fly.io</a>	Ejecutan tu servidor Express.js en Node.js	Gratis (con límite), soporta APIs y bases de datos.



**Siguiente semana:  
Despliegue/Deploy en la nube**