



# Vue JS

# CONSUMIR API

---

Pablo Ortiz G.



[www.sena.edu.co](http://www.sena.edu.co)

# Instalar Axios


Axios: es un paquete que encontraremos en npm y que nos permite hacer peticiones o llamadas al contenido de un enlace HTTP

1. En la consola de comandos:

`npm install --save axios vue-axios`

2. Importar los paquetes instalados

3. Usar los paquetes importados

```
src >  main.js
 1  import { createApp } from 'vue'
 2  import App from './App.vue'
 3  import router from './router'
 4
 5  // Importar el paquete de Axios instalado
 6  import axios from 'axios'
 7  import VueAxios from 'vue-axios'
 8
 9  // Importar bootstrap
10  import "bootstrap/dist/css/bootstrap.min.css"
11  import "bootstrap"
12
13  // usar los paquetes importados use(VueAxios, axios)
14  createApp(App).use(router).use(VueAxios, axios).mount('#app')
15
```



# GET

# Crear Componente

Crear un componente o vista en donde se realizara la extracción de los datos de la API.

Estructura básica

```
> components > ✓ PostComponent.vue > Vetur > {} "PostCompo
1  <template>
2    <main>
3      <h1>Lista de Usuarios Registrados</h1>
4    </main>
5  </template>
6
7  <script>
8
9  export default {
10    name: 'GetComponente',
11    data() {
12      return {
13
14      }
15    },
16
17  }
18 </script>
19 <style scoped>
20
21 </style>
```

# Crear Método

En el script del componente realizar los pasos:

**PASO 1:** En data crear una variable para guardar los que serán extraídos de la API creada, para el ejemplo la variable usuarios guardara el arreglo o JSON que genera la API.

**PASO 2:** Crear el método para extraer los datos de la API mediante el paquete AXIOS.

Axios devuelve una promesa con la respuesta que se almacena en response **.then**

Permite establecer un estado de error si la respuesta de la API es negativa **.catch**

```
src > components > GetComponent.vue > Vue Language Features (Volar) > {} style scoped
28 </template>
29 <script>
30 export default {
31   name: 'GetComponent',
32   data(){
33     return{
34       // PASO 1. Crear variable para guardar los datos de la API
35       usuarios: []
36     }
37   },
38   // PASO 2. crear el metodo para extraer los datos de la API
39   async mounted(){
40     // hacer la peticion GET a la ruta de la API
41     await this.axios.get('http://localhost:3000/api/usuarios')
42     // devuelve la promesa con un resultado (response)
43     .then(response => {
44       // guardar en la variable usuarios la respuesta obtenida
45       this.usuarios = response.data;
46       // si se quiere ver el resultado en consola
47       console.log(response.data)
48     }) //Mostrar por consola el error
49     .catch((e) => {
50       console.log(e)
51     });
52   }
53 }
54 </script>
55 <style scoped>
```

# Mostrar los Datos

Crear la interfaz de usuario para mostrar los datos.

Se crea una tabla con su encabezado para los títulos y el cuerpo para colocar los datos.

La directiva v-for, permite iterar los datos del arreglo guardado en la variable usuarios para colocar uno por uno en cada celda.

La directiva v-bind permite asignar la secuencia de datos con un índice.

```
src > components > GetComponent.vue > Vetur > {} "GetComponent.vue" > template > div > table.table.table
1  <template>
2    <div>
3      // PASO 3. Crear la Interfaz para mostrar los datos
4      <h1> Listado de Usuarios</h1>
5      // crear la Tabla para organizar los Datos
6      <table class="table table-primary table-bordered border-primary">
7        // crear encabezados para los Títulos
8        <thead>
9          <tr class="table-warning">
10             <td>ID</td>
11             <td>NOMBRE</td>
12             <td>CIUDAD</td>
13             <td>ROL</td>
14             <td>ACCION</td>
15          </tr>
16        </thead>
17        // crear el cuerpo de la tabla para ubicar los datos
18        <tbody>
19          // Recorrer los datos de la Variable Usuarios mediante V-for
20          <tr v-for="usuario in usuarios" v-bind:key="usuario">
21            // interpolar los datos del arreglo en cada celda
22            <td>{{ usuario.id }}</td>
23            <td>{{ usuario.name }}</td>
24            <td>{{ usuario.city }}</td>
25            <td>{{ usuario.rol }}</td>
26            <td>
27              <input type="button" value="Editar" class="btn btn-primary">
28              <input type="button" value="Eliminar" class="btn btn-danger">
29            </td>
30          </tr>
31        </tbody>
32      </table>
33    </div>
34  </template>
35  <script>
```



# POST

# Crear Componente

Crear un componente o vista en donde se realizara la extracción de los datos de la API.

Estructura básica

```
1  <template>
2  |   <main>
3  |     <h1>Cconsultar Usuario por ID</h1>
4  |   </main>
5  </template>
6  <script>
7  export default {
8  |   name: 'PostComponent',
9  |   data(){
10 |     return{
11 |
12 |     }
13 |   }
14 }
15 </script>
16 <style scoped>
17
18 </style>
```



# Crear Método

En el script del componente realizar los pasos:

**PASO 1:** En data crear una variable para guardar los que serán enviados a la API, se crea un arreglo tipo JSON, con los campos vacíos (campos que se llenaran desde un formulario)

**PASO 2:** Crear el método para enviar los datos a la API (postUser)

Axios envía los datos a la API desde una arreglo que esta en la variable (usuarioNuevo)

Se puede ver el resultado del envío en la consola

```
<script>
import axios from 'axios';
export default {
  name: 'PostComponent',
  data() {
    return { // PASO 1
      // Crear variable para almacenar los datos que se enviaran a la API
      // los campos estaran vacios por que se insertan desde un formulario
      usuarioNuevo: {
        name: '',
        city: '',
        rol: ''
      }
    },
    methods: { // PASO 2: se crea el metodo con el nombre postUser
      postUser() {
        axios // con el metodo POST se enviaran los datos a la API
        // los datos se envian a la API desde la variable UsuarioNuevo
        .post('http://localhost:3000/api/usuarios', this.usuarioNuevo)
        // se podran ver por consola una vez se envien
        .then((response) => console.log(response))
      }
    }
  }
}</script>
```

# Insertar los Datos

Crear la interfaz de usuario para ingresar los datos a la variable creada (usuarioNuevo) en data.

La directiva v-model conecta el campo (input) del formulario con el campo de la variable creada en data (usuarioNuevo.name).

Mediante el formulario se capturan los datos y se envían a la variable para que axios los envíe a la API

```
<template>
  <main>
    <h1>Registrar Nuevo Usuario</h1>
    PASO 3: Crear la Intefaz
    <form @submit =postUser()>
      <div>
        <label for="nomUser">Nombre</label>
        <input type="text" id="nomUser" v-model="usuarioNuevo.name">
      </div>
      <br>
      <div>
        <label for="cityUser">Ciudad</label>
        <input type="text" id="cityUser" v-model="usuarioNuevo.city">
      </div>
      <br>
      <div>
        <label for="rolUser">Rol</label>
        <input type="text" id="rolUser" v-model="usuarioNuevo.rol">
      </div>
      <button>Crear</button>
    </form>
  </main>
</template>
```



# GET por ID

# Crear Componente

Crear un componente o vista en donde se realizara la extracción de los datos de la API.

Estructura básica

```
1  <template>
2    <main>
3      <h1>Consultar Usuario por ID</h1>
4    </main>
5  </template>
6  <script>
7    export default {
8      name: 'GetIdComponent',
9      data(){
10        return{
11
12        }
13      }
14    }
15  </script>
16  <style scoped>
17
18  </style>
```

# Crear Método

En el script del componente realizar los pasos:

**PASO 1:** En data crear una variable para guardar los que serán enviados a la API, se crea un arreglo tipo JSON, con los campos vacíos (campos que se llenaran desde un formulario)

**PASO 2:** Crear el método para enviar los datos a la API (postUser)

Axios envía los datos a la API desde una arreglo que esta en la variable (usuarioNuevo)

Se puede ver el resultado del envío en la consola

```
<script>
import axios from 'axios';
export default {
  name: 'PostComponent',
  data() {
    return { // PASO 1
      // Crear variable para almacenar los datos que se enviaran a la API
      // los campos estaran vacios por que se insertan desde un formulario
      usuarioNuevo: {
        name: '',
        city: '',
        rol: ''
      }
    },
    methods: { // PASO 2: se crea el metodo con el nombre postUser
      postUser() {
        axios // con el metodo POST se enviaran los datos a la API
        // los datos se envian a la API desde la variable UsuarioNuevo
        .post('http://localhost:3000/api/usuarios', this.usuarioNuevo)
        // se podran ver por consola una vez se envien
        .then((response) => console.log(response))
      }
    }
  }
}</script>
```

# Insertar los Datos

Crear la interfaz de usuario para ingresar los datos a la variable creada (usuarioNuevo) en data.

La directiva v-model conecta el campo (input) del formulario con el campo de la variable creada en data (usuarioNuevo.name).

Mediante el formulario se capturan los datos y se envían a la variable para que axios los envíe a la API

```
<template>
  <main>
    <h1>Registrar Nuevo Usuario</h1>
    PASO 3: Crear la Intefaz
    <form @submit =postUser()>
      <div>
        <label for="nomUser">Nombre</label>
        <input type="text" id="nomUser" v-model="usuarioNuevo.name">
      </div>
      <br>
      <div>
        <label for="cityUser">Ciudad</label>
        <input type="text" id="cityUser" v-model="usuarioNuevo.city">
      </div>
      <br>
      <div>
        <label for="rolUser">Rol</label>
        <input type="text" id="rolUser" v-model="usuarioNuevo.rol">
      </div>
      <button>Crear</button>
    </form>
  </main>
</template>
```





**G R A C I A S**

Línea de atención al ciudadano: 01 8000 910270  
Línea de atención al empresario: 01 8000 910682



[www.sena.edu.co](http://www.sena.edu.co)