

**ACTIVIDAD DE VIDEO – VUE3
PROYECTO DE SOFTWARE III**

PRESENTADO POR:

JOAN SAMUEL ANACONA NARVAEZ

DOCENTE:

ADRIÁN IZQUIERDO

**INSTITUTO TECNOLÓGICO DEL PUTUMAYO-
UNIPUTUMAYO**

MOCOA

05 – 09 -2025

CONCEPTOS BÁSICOS

Configuración del entorno y estructura de un componente. En Vue 3, la unidad fundamental de trabajo son los **SFC (Single File Components)**, archivos con extensión .vue que integran en un solo lugar la parte visual, la lógica y los estilos. Cada componente se organiza en tres bloques principales:

1. **<template>**
Aquí se define la vista o estructura visual de la aplicación, escrita en HTML. En esta sección se colocan elementos como botones, formularios, listas, entre otros.
2. **<script>**
Contiene la lógica del componente en JavaScript o TypeScript. Es el espacio donde se declaran los datos reactivos (ref, reactive), los métodos, las propiedades, y también donde se pueden importar y usar *composables* para reutilizar lógica.
3. **<style>**
Permite aplicar los estilos propios del componente, ya sea en CSS o usando preprocesadores como SCSS.

Composition API – Fundamentos

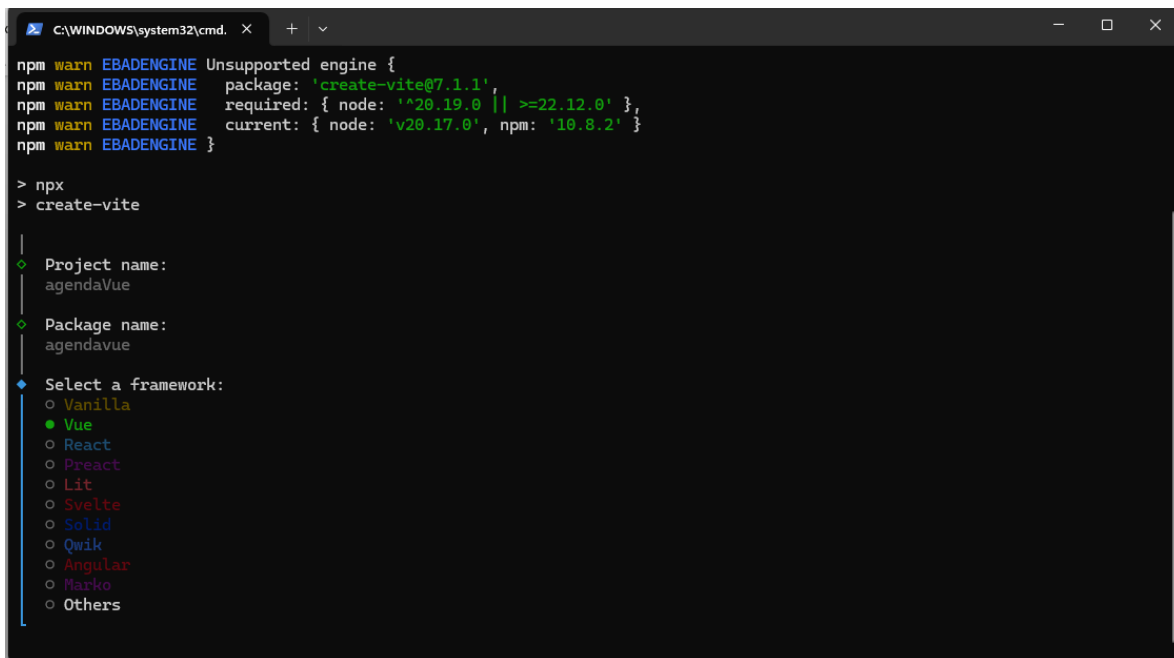
La **Composition API** es la forma moderna de trabajar con Vue 3 y se organiza principalmente en la función setup(). Todo lo que se declare dentro de setup (variables, funciones, props o estados) estará disponible para el componente.

- **Reactividad**
 - ref(): define variables reactivas de tipo simple (números, cadenas, booleanos).
 - reactive(): convierte un objeto o un arreglo completo en reactivo.
- **Propiedades derivadas**
 - computed(): crea valores calculados automáticamente en función de otros, actualizándose cuando cambian sus dependencias.
- **Observadores**
 - watch(): vigila variables específicas y ejecuta acciones cuando estas cambian.
 - watchEffect(): se ejecuta automáticamente, detectando las variables reactivas que se utilizan en su interior.
- **Ciclo de vida del componente**
Vue proporciona funciones que se ejecutan en momentos clave del ciclo de vida:
 - onMounted: cuando el componente se carga en la vista.
 - onUpdated: cuando ocurre una actualización en el estado o en las props.
 - onUnmounted: cuando el componente es eliminado de la vista.

Uso de <script setup>

Vue 3 introduce la sintaxis <script setup>, que simplifica la forma de trabajar. Todo lo que se defina dentro de este bloque queda disponible en el <template> sin necesidad de retornar manualmente. Esto hace que el código sea más limpio y fácil de mantener.

- Pasos para crear un Proyecto en Vue



```
C:\WINDOWS\system32\cmd. X + v
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: 'create-vite@7.1.1',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.2' }
npm warn EBADENGINE }

> npx
> create-vite

|
| Project name:
| agendaVue
|
| Package name:
| agendavue
|
| Select a framework:
|   o Vanilla
|   ● Vue
|   o React
|   o Preact
|   o Lit
|   o Svelte
|   o Solid
|   o Qwik
|   o Angular
|   o Marko
|   o Others
```

- Ya creado el Proyecto entro a la Carpeta Creada

```
Windows PowerShell
> npx
> create-vite

Project name:
agendaVue

Package name:
agendavue

Select a framework:
Vue

Select a variant:
JavaScript

Scaffolding project in D:\agendaVue...

Done. Now run:

cd agendaVue
npm install
npm run dev

npm notice
npm notice New major version of npm available! 10.8.2 -> 11.6.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.6.0
npm notice To update run: npm install -g npm@11.6.0
npm notice
PS D:\> cd .\agendaVue\
```

- Instalamos dependencias

```
Windows PowerShell
npm notice To update run: npm install -g npm@11.6.0
npm notice
PS D:\> cd .\agendaVue\
PS D:\agendaVue> npm intall
Unknown command: "intall"

Did you mean this?
  npm install # Install a package
To see a list of supported npm commands, run:
  npm help
PS D:\agendaVue> npm install
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: '@vitejs/plugin-vue@6.0.1',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.2' }
npm warn EBADENGINE }
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: 'vite@7.1.4',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.2' }
npm warn EBADENGINE }

added 34 packages, and audited 35 packages in 19s

6 packages are looking for funding
  run 'npm fund' for details

found 0 vulnerabilities
PS D:\agendaVue>
```

- Nos da la Url de el proyecto inicial e por defecto que seria esta

```
C:\WINDOWS\system32\cmd. X + v
npm warn EBADENGINE current: { node: 'v20.17.0', npm: '10.8.2' }
npm warn EBADENGINE }

added 34 packages, and audited 35 packages in 19s

6 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS D:\agendaVue> npm run
Scripts available in agendaVue@0.0.0 via `npm run-script`:
  dev
    vite
  build
    vite build
  preview
    vite preview
PS D:\agendaVue> npm run dev
> agendaVue@0.0.0 dev
> vite

You are using Node.js 20.17.0. Vite requires Node.js version 20.19+ or 22.12+. Please upgrade your Node.js version.

VITE v7.1.4 ready in 1313 ms
  → Local:   http://localhost:5173/
  → Network: use --host to expose
  → press h + enter to show help
```

- Ejecutamos la url en el navegador y nos muestra siguiente



Vite + Vue

count is 7

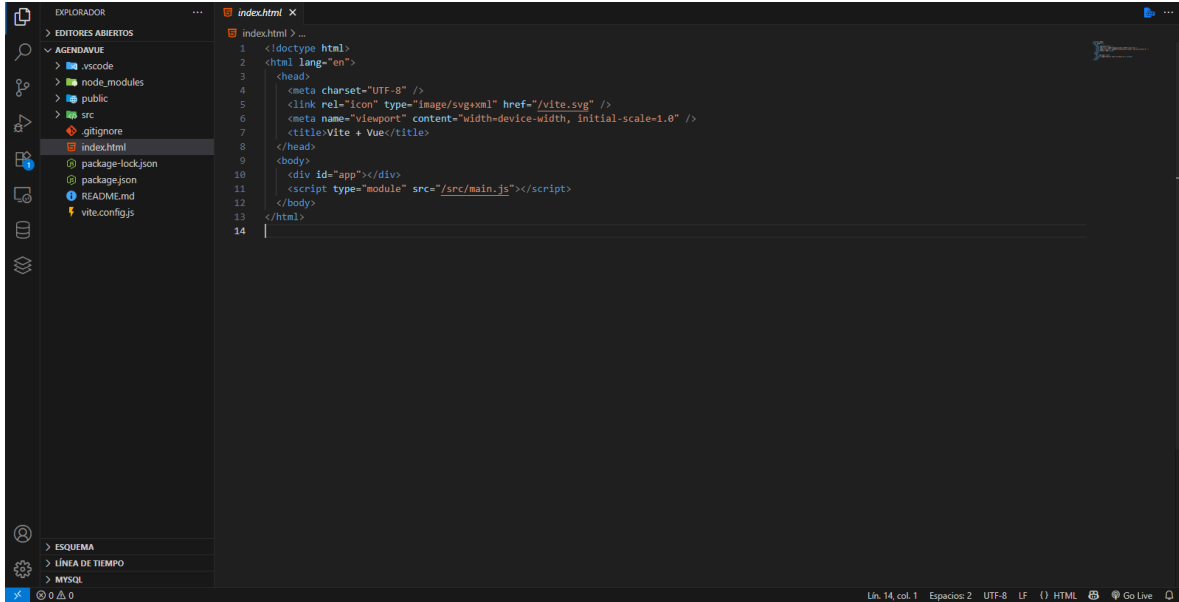
Edit components/HelloWorld.vue to test HMR

Check out [create-vue](#), the official Vue + Vite starter

Learn more about IDE Support for Vue in the [Vue Docs Scaling up Guide](#).

Click on the Vite and Vue logos to learn more

- Abrimos nuestro editor fav de código (en este abre el proyecto creado)



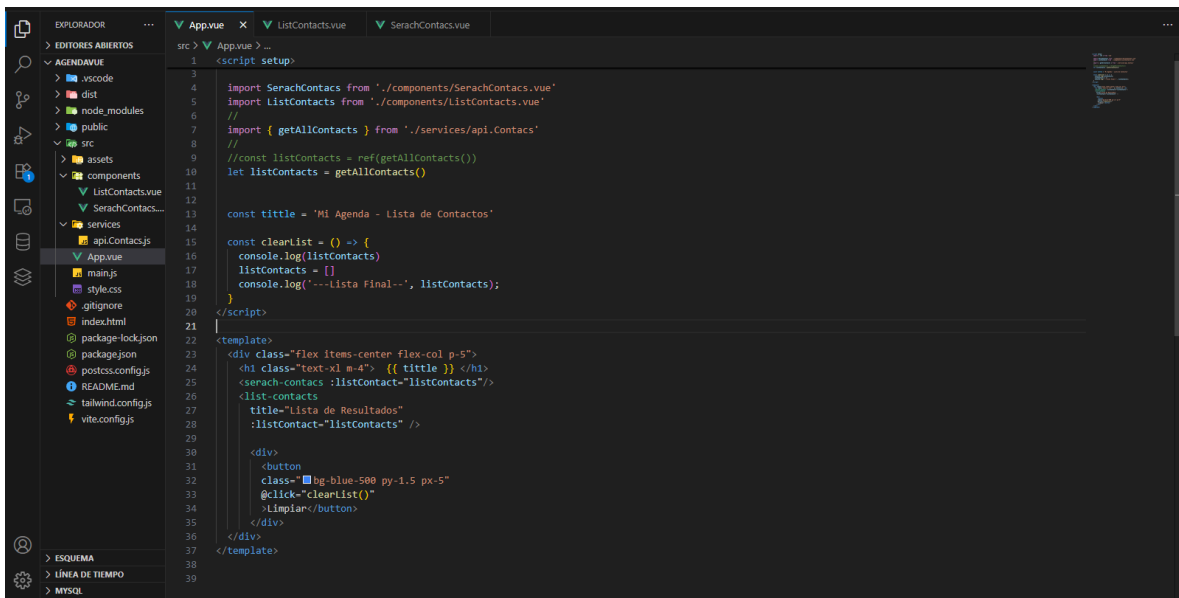
```

1 <!doctype html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8" />
5   <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7   <title>Vite + Vue</title>
8 </head>
9 <body>
10   <div id="app"></div>
11   <script type="module" src="/src/main.js"></script>
12 </body>
13 </html>
14
  
```

Manejo de estados

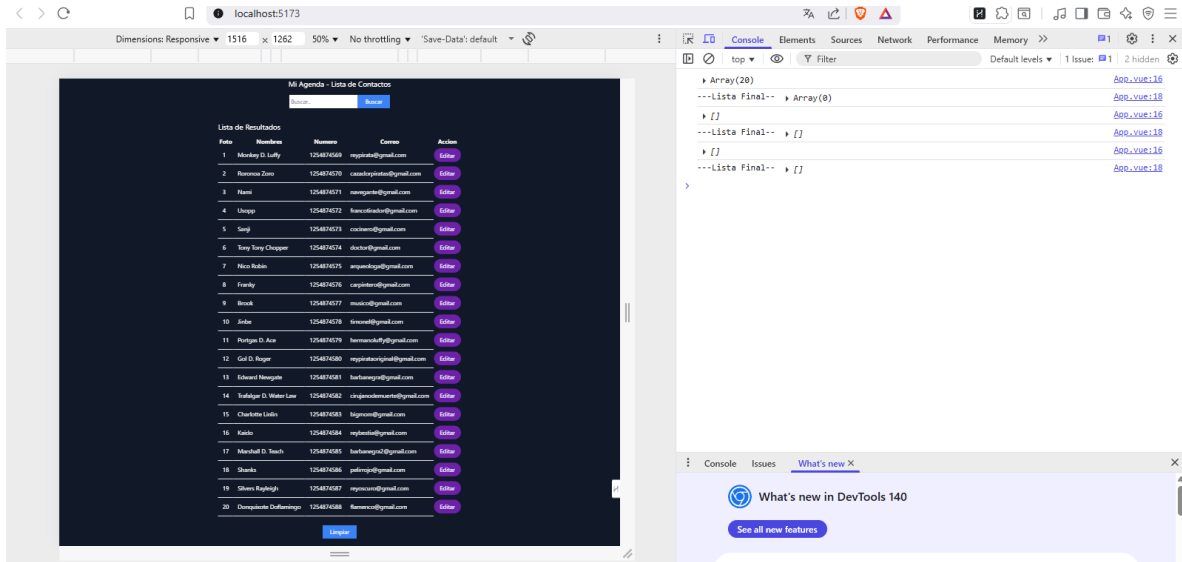
Los estados en Vue 3, se implementó una función para limpiar la lista de contactos. Inicialmente, al ejecutar la acción, los datos sí se eliminaban en la consola, pero no se reflejaban en la interfaz del frontend.

Este ejercicio permitió comprender que la reactividad no solo depende de cambiar valores en el código, sino también de asegurarse de que las variables reactivas estén correctamente vinculadas con la vista para que los cambios se visualicen al instante.



```

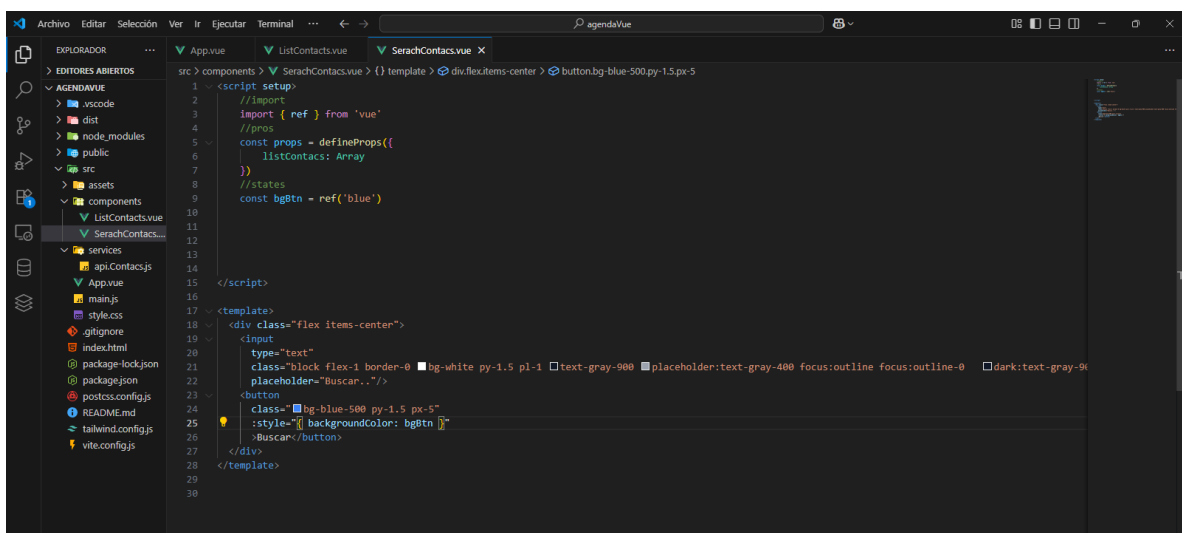
1 <script setup>
2
3
4 import SerachContacts from './components/SerachContacts.vue'
5 import ListContacts from './components/ListContacts.vue'
6 //
7 import { getAllContacts } from './services/api.Contacts'
8 //
9 //const listContacts = ref(getAllContacts())
10 let listContacts = getAllContacts()
11
12
13 const title = 'Mi Agenda - Lista de Contactos'
14
15
16 const clearList = () => {
17   console.log(listContacts)
18   listContacts = []
19   console.log('---Lista Final---', listContacts)
20 }
21
22 </script>
23
24 <template>
25   <div class="flex items-center flex-col p-5">
26     <h1 class="text-xl m-4"> {{ title }} </h1>
27     <serach-contacts :listContact="listContacts" />
28     <list-contacts
29       title="Lista de Resultados"
30       :listContact="listContacts" />
31
32     <div>
33       <button
34         class="bg-blue-500 py-1.5 px-5"
35         @click="clearList()"
36       >limpiar</button>
37     </div>
38   </div>
39 </template>
  
```

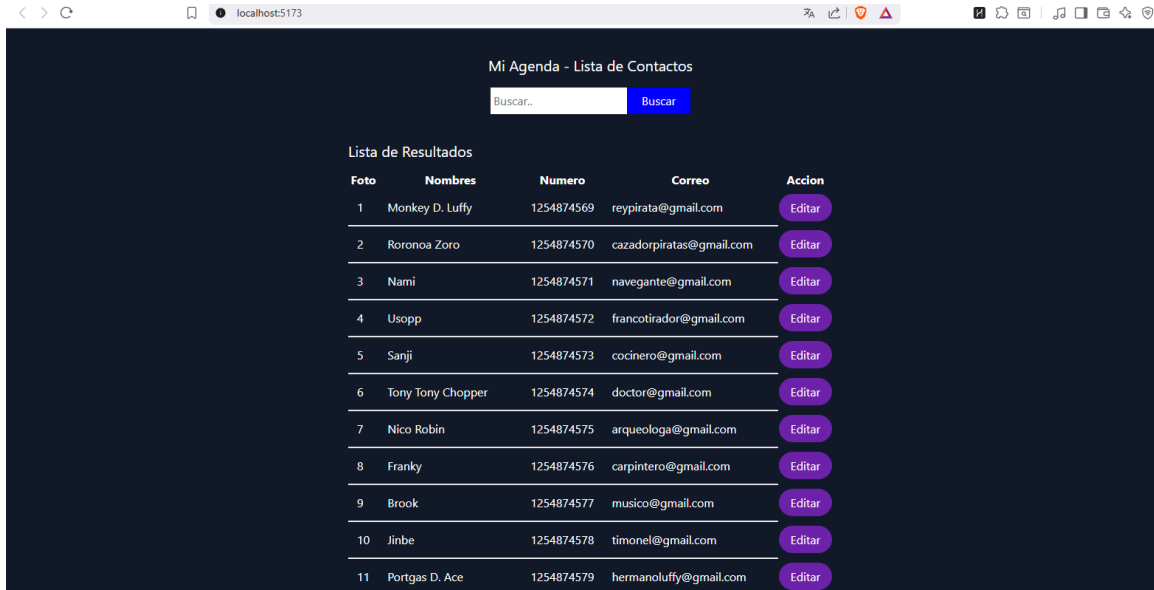


Enlace de datos (Binding)

El *binding* en Vue 3 se refiere al proceso de **conectar los datos definidos en el bloque `<script>` con los elementos del `<template>`**. Gracias a este enlace automático, cualquier cambio que ocurra en las variables de JavaScript se refleja de inmediato en la vista, sin necesidad de escribir código adicional.

De esta manera, los valores que gestionamos en el estado del componente pueden mostrarse directamente en la interfaz HTML, manteniendo sincronizados los datos y la presentación.

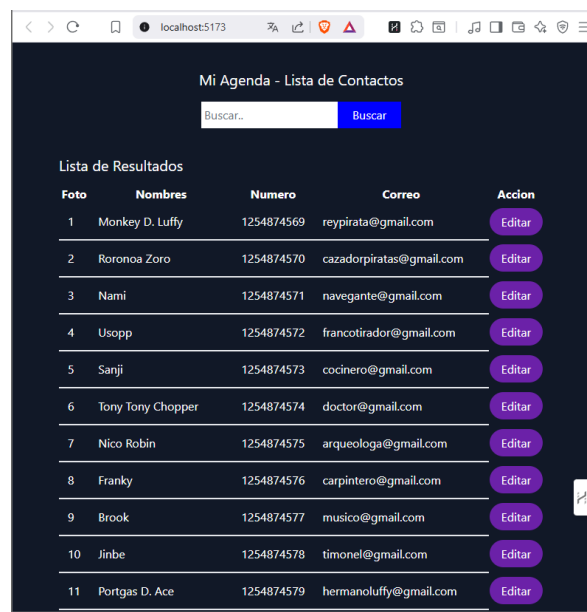


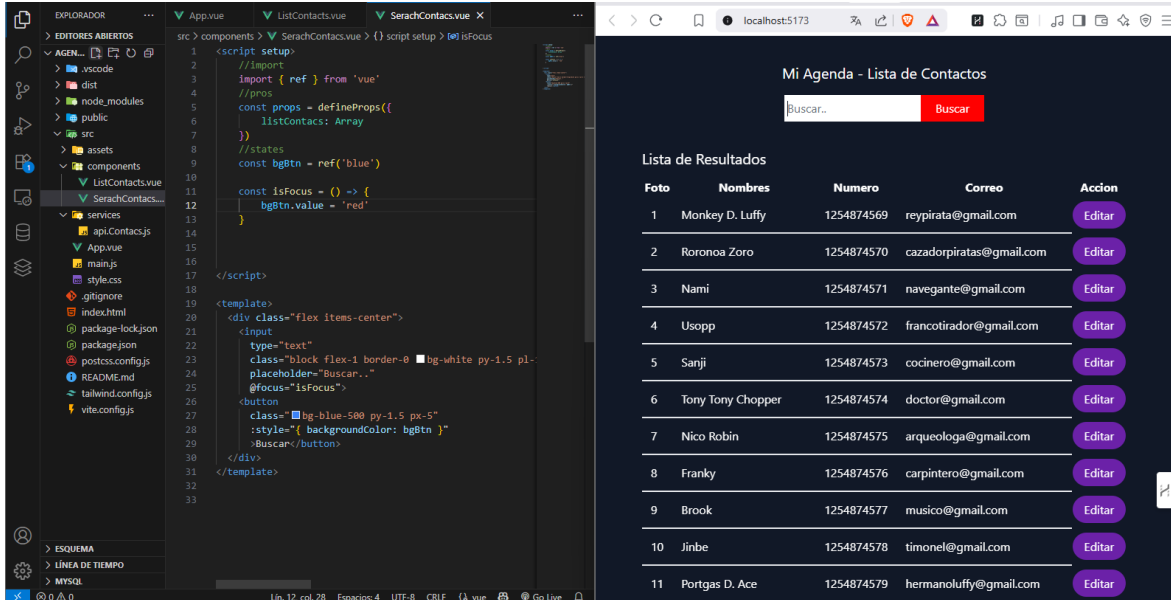


Eventos en Vue – Ejemplo con “focus”

Se probó el manejo de eventos dentro de Vue 3. Para el caso del botón de búsqueda, se configuró que, al escribir en el campo correspondiente y activar el evento focus, el botón cambiara de color automáticamente.

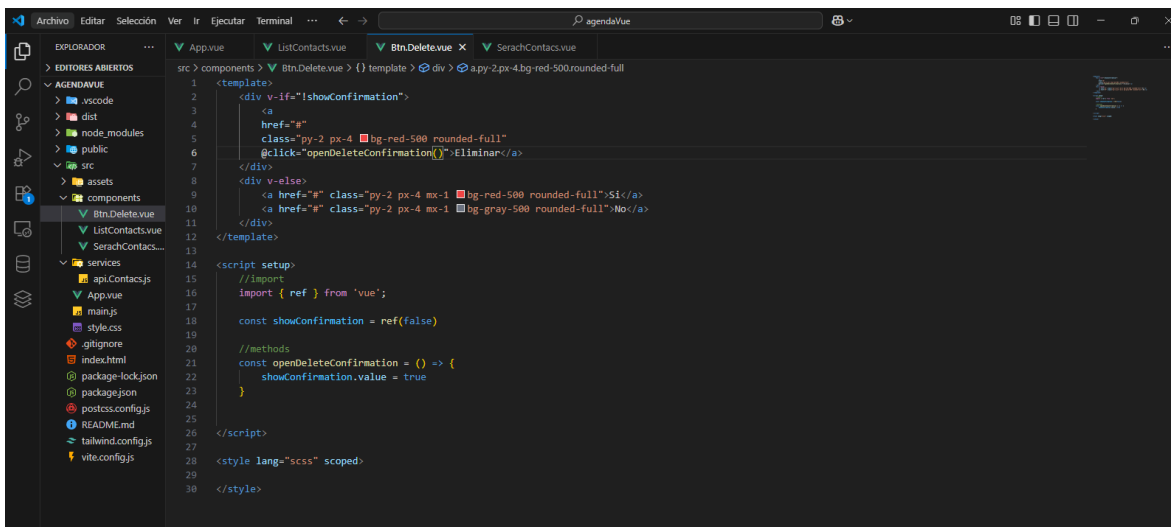
Esta parte del ejercicio permitió comprobar cómo los eventos pueden: modificar dinámicamente la apariencia o el comportamiento de un elemento, reforzando la interacción entre el usuario y la interfaz.

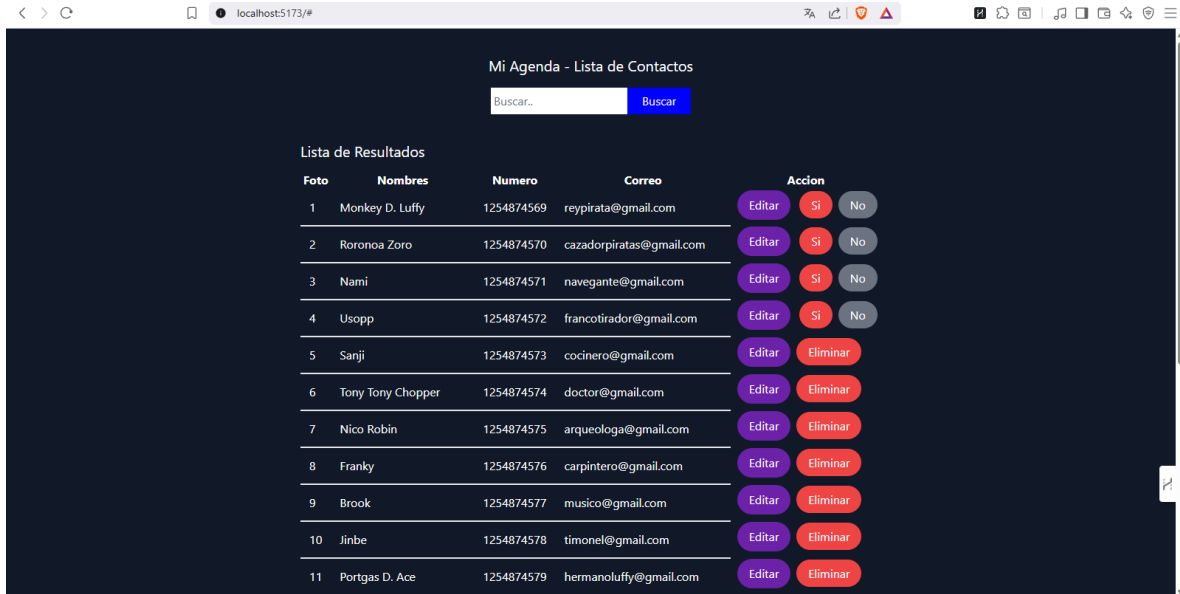




Implementación del botón de eliminar

Se añadió un botón de eliminación que, al ser presionado por el usuario, muestra automáticamente una confirmación con las opciones **“SI”** o **“No”**. De esta manera, antes de ejecutar la acción definitiva, el sistema da la oportunidad de confirmar o cancelar, evitando eliminaciones accidentales y mejorando la experiencia de uso.

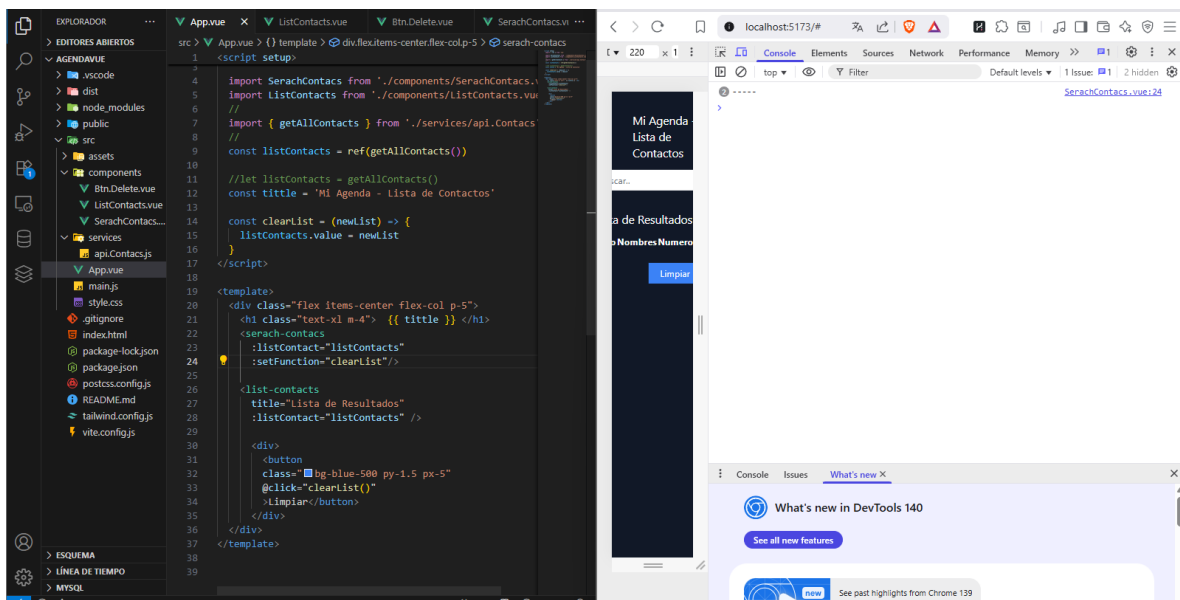


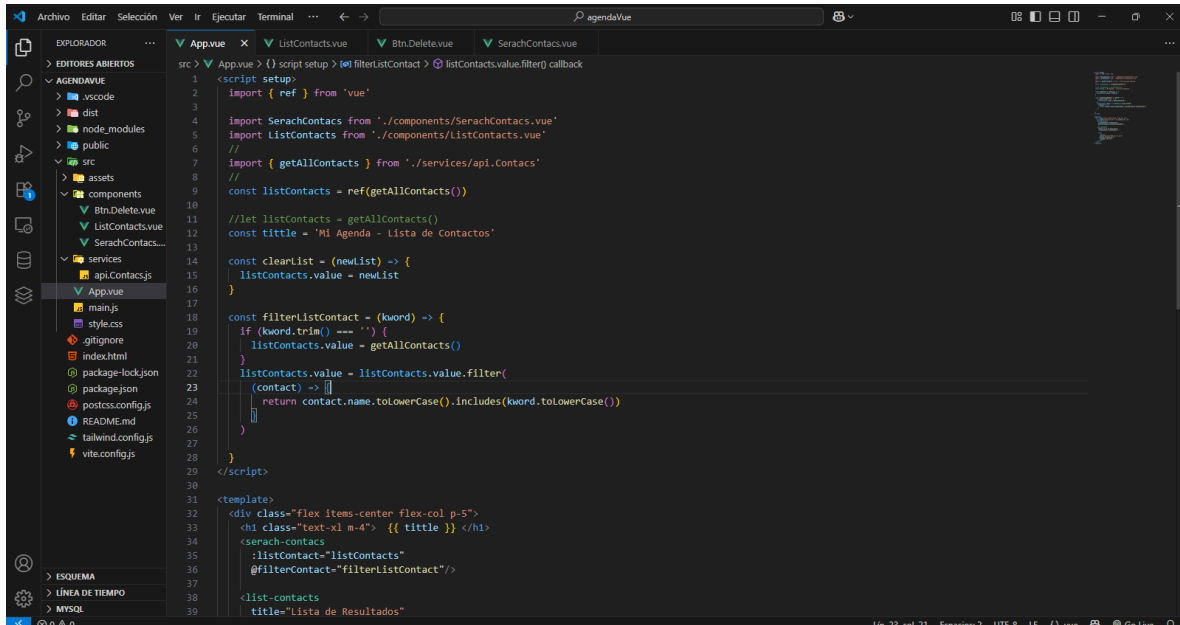


Búsqueda dinámica en la lista

Se implementó una funcionalidad que permite filtrar la lista en tiempo real. A medida que el usuario escribe en el campo de búsqueda, la lista se va actualizando automáticamente y muestra únicamente los elementos que coinciden con el texto ingresado.

Esto facilita encontrar rápidamente la información deseada y demuestra cómo Vue gestiona la reactividad para actualizar la interfaz de manera inmediata según los cambios en los datos.





```

1 <script setup>
2 import { ref } from 'vue'
3
4 import SerachContacts from '../components/SerachContacts.vue'
5 import ListContacts from '../components/ListContacts.vue'
6 //
7 import { getAllContacts } from '../services/api.Contacts'
8 //
9 const listContacts = ref(getAllContacts())
10
11 //let listContacts = getAllContacts()
12 const title = 'Mi Agenda - Lista de Contactos'
13
14 const clearList = (newList) => {
15   listContacts.value = newList
16 }
17
18 const filterListContact = (keyword) => {
19   if (keyword.trim() === '') {
20     listContacts.value = getAllContacts()
21   }
22   listContacts.value = listContacts.value.filter(
23     (contact) => {
24       return contact.name.toLowerCase().includes(keyword.toLowerCase())
25     }
26   )
27 }
28
29 </script>
30
31 <template>
32 <div class="flex items-center flex-col p-5">
33   <h1 class="text-xl m-4"> {{ title }} </h1>
34   <serach-contact>
35     <list-contact="listContacts"
36       @filterContact="filterListContact"/>
37   </list-contacts>
38   <title="Lista de Resultados"
  
```

Resultado de la búsqueda y eliminación automática

Se comprobó que al escribir un término la lista se filtra correctamente. Posteriormente, al borrar el texto del buscador, los resultados también se eliminan de manera automática, restaurando la lista completa.

Mi Agenda - Lista de Contactos

Buscar

Lista de Resultados

Foto	Nombres	Numero	Correo	Accion	
4	Usopp	1254874572	francotirador@gmail.com	Editar	Eliminar
5	Sanji	1254874573	cocinero@gmail.com	Editar	Eliminar
11	Portgas D. Ace	1254874579	hermanoluffy@gmail.com	Editar	Eliminar
17	Marshall D. Teach	1254874585	barbanegra2@gmail.com	Editar	Eliminar
18	Shanks	1254874586	pelirrojo@gmail.com	Editar	Eliminar
19	Silvers Rayleigh	1254874587	reyoscuro@gmail.com	Editar	Eliminar

Limpiar

Mi Agenda - Lista de Contactos

Buscar..

Buscar

Lista de Resultados

Foto	Nombres	Numero	Correo	Accion	
1	Monkey D. Luffy	1254874569	reypirata@gmail.com	Editar	Eliminar
2	Roronoa Zoro	1254874570	cazadorpiratas@gmail.com	Editar	Eliminar
3	Nami	1254874571	navegante@gmail.com	Editar	Eliminar
4	Usopp	1254874572	francotirador@gmail.com	Editar	Eliminar
5	Sanji	1254874573	cocinero@gmail.com	Editar	Eliminar
6	Tony Tony Chopper	1254874574	doctor@gmail.com	Editar	Eliminar
7	Nico Robin	1254874575	arqueologa@gmail.com	Editar	Eliminar
8	Franky	1254874576	carpintero@gmail.com	Editar	Eliminar
9	Brook	1254874577	musico@gmail.com	Editar	Eliminar
10	Jinbe	1254874578	timonel@gmail.com	Editar	Eliminar

Incorporación de la propiedad “recompensa”

En la etapa final del proyecto se agregó una nueva propiedad llamada “**recompensa**” dentro del archivo `apiContact.js`. Esta propiedad fue calculada a partir de ciertos parámetros definidos en el código, lo que permitió asignar dinámicamente un valor específico a cada contacto.

La implementación de este método demuestra cómo se pueden extender los datos provenientes de una API, adaptándolos a las necesidades de la aplicación y enriqueciendo la información mostrada en la interfaz.

```

13     }
14   })
15   //methods
16   const totalRecompesas = () => {
17     return props.listContact.reduce(
18       (accumulator, contact) => accumulator + contact.recompensa, 0
19     )
20   }
21 </script>
22
23 <template>
24   <div class="m-6">
25     <h3 class="text-xl my-3"> {{ title }}- {{ totalRecompesas }}</h3>
26     <table class="table-fixed">

```

Mi Agenda - Lista de Contactos

Buscar..

Buscar

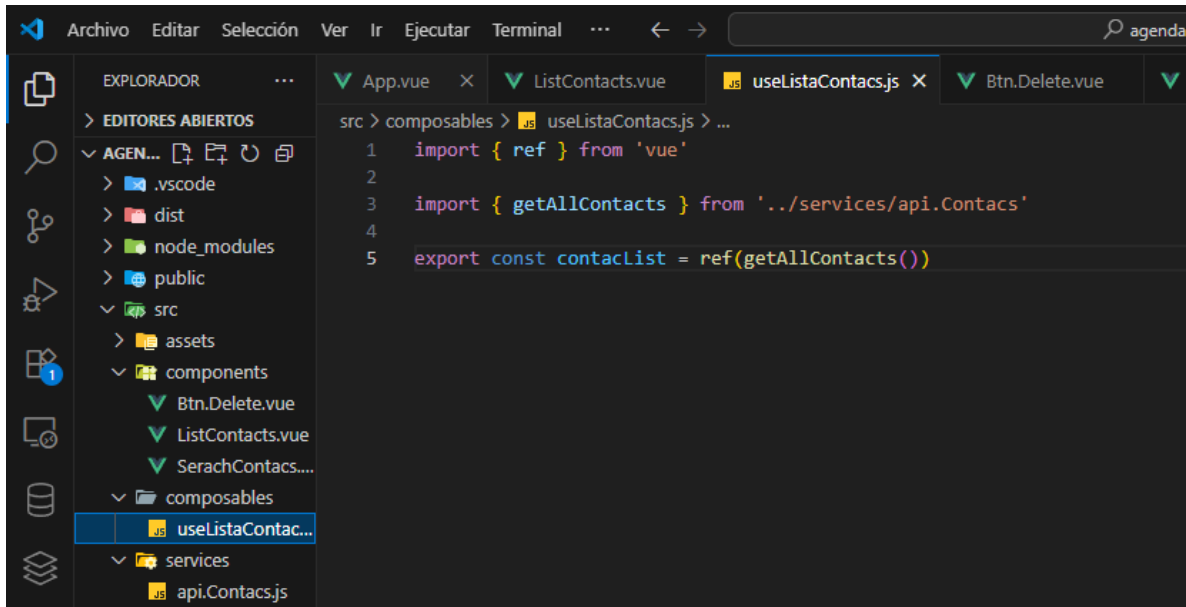
Lista de Resultados- 315.401

Foto	Nombres	Numero	Correo	Accion	
1	Monkey D. Luffy	1254874569	reypirata@gmail.com	Editar	Eliminar
2	Roronoa Zoro	1254874570	cazadorpiratas@gmail.com	Editar	Eliminar
3	Nami	1254874571	navegante@gmail.com	Editar	Eliminar
4	Usopp	1254874572	francotirador@gmail.com	Editar	Eliminar
5	Sanji	1254874573	cocinero@gmail.com	Editar	Eliminar
6	Tony Tony Chopper	1254874574	doctor@gmail.com	Editar	Eliminar
7	Nico Robin	1254874575	arqueologa@gmail.com	Editar	Eliminar
8	Franky	1254874576	carpintero@gmail.com	Editar	Eliminar
9	Brook	1254874577	musico@gmail.com	Editar	Eliminar
10	Jinbe	1254874578	timonel@gmail.com	Editar	Eliminar

Archivos globales en el proyecto

Los archivos globales cumplen la función de centralizar configuraciones o funcionalidades que se necesitan en diferentes partes de la aplicación. Su uso permite reducir la cantidad de **props** que se deben pasar entre componentes y evita escribir código repetitivo o demasiado extenso.

De esta manera, se logra un proyecto más organizado, fácil de mantener y con una estructura más clara.



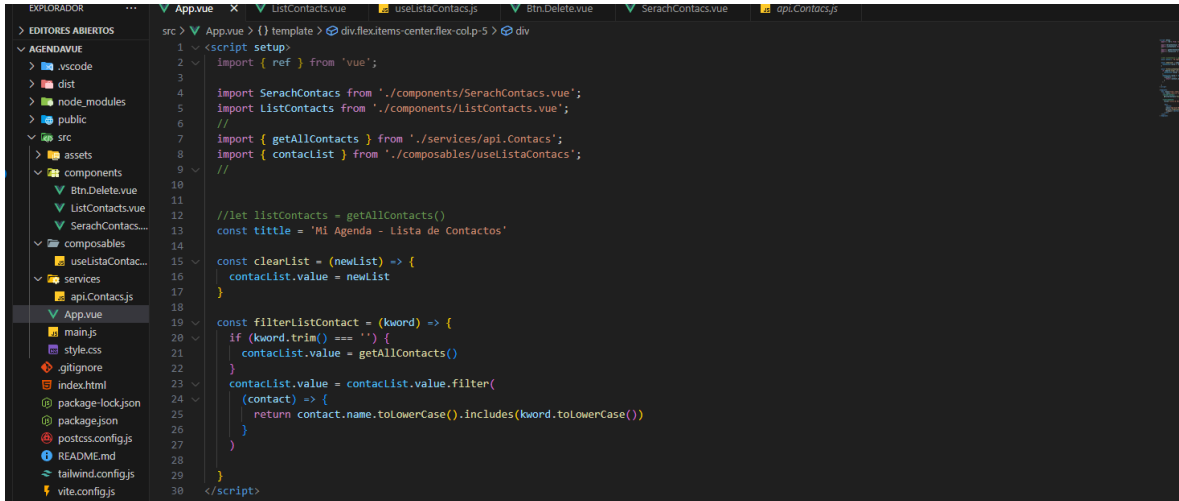
The screenshot shows the VS Code interface. The Explorer sidebar on the left displays the project structure, with the 'composables' folder expanded, showing 'useListaContacs.js'. The main editor area shows the content of 'useListaContacs.js' with the following code:

```
src > composables > useListaContacs.js > ...
1 import { ref } from 'vue'
2
3 import { getAllContacts } from '../services/api.Contacs'
4
5 export const contacList = ref(getAllContacts())
```



The screenshot shows the VS Code interface with the 'ListContacts.vue' file open in the main editor. The breadcrumb navigation at the top indicates the path: 'src > components > ListContacts.vue > {} script setup > totalRecompesa > computed() callback'. The code in the editor is as follows:

```
1 <script setup>
2   import { computed } from 'vue';
3   //import
4   import { contacList } from '../composables/useListaContacs';
5   import BtnDelete from './Btn.Delete.vue';
6   //props
7   const props = defineProps({
8     title: {
9       type: String,
10      default: 'Lista'
11    },
12  })
13  //methods
14  const totalRecompesa = computed(() => {
15    return contacList.value.reduce(
16      (accumulator, contact) => accumulator + contact.recompensa, 0
17    )
18  })
19
20  const totalRecompesas = () => {
21    return contacList.value.reduce(
22      (accumulator, contact) => accumulator + contact.recompensa, 0
23    )
24  }
```



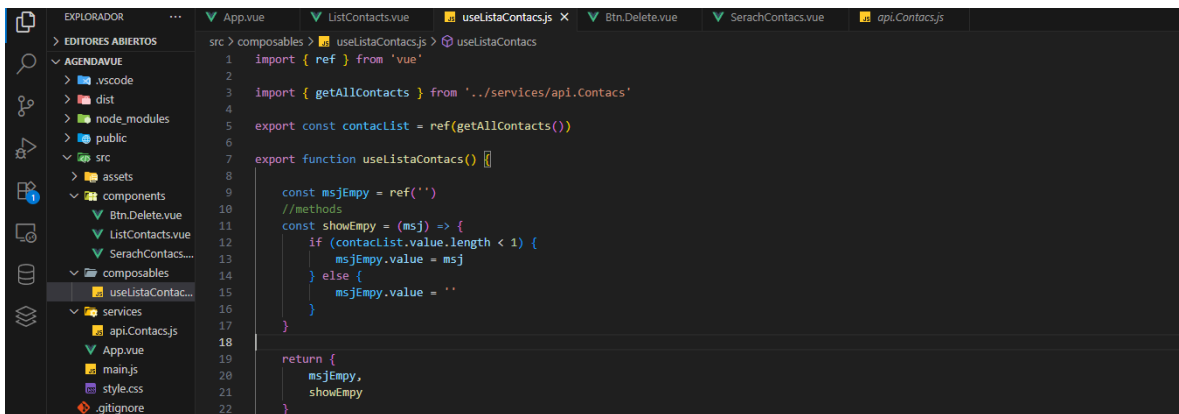
```

1 <script setup>
2 import { ref } from 'vue';
3
4 import SerachContacts from '../components/SerachContacs.vue';
5 import ListContacts from '../components/ListContacts.vue';
6 //
7 import { getAllContacts } from '../services/api.Contacs';
8 import { contactList } from '../composables/useListaContacs';
9 //
10
11 //let listContacts = getAllContacts()
12 const title = 'MI Agenda - Lista de Contactos'
13
14 const clearList = (newList) => {
15   contactList.value = newList
16 }
17
18 const filterListContact = (keyword) => {
19   if (keyword.trim() === '') {
20     contactList.value = getAllContacts()
21   }
22   contactList.value = contactList.value.filter(
23     (contact) => {
24       return contact.name.toLowerCase().includes(keyword.toLowerCase())
25     }
26   )
27 }
28
29 </script>
  
```

Creación de un composable para búsquedas

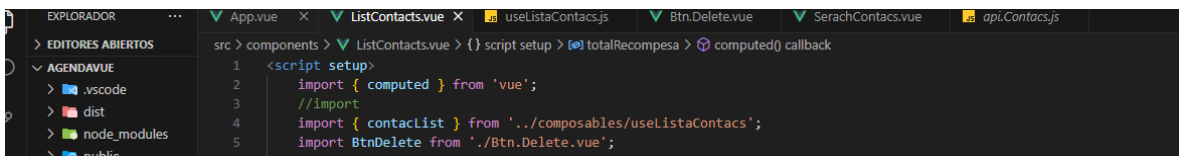
Se desarrolló un **composable** que permite gestionar el caso en el que una búsqueda no arroje resultados. Su función es detectar cuando la lista está vacía y mostrar automáticamente un mensaje al usuario, informando que no se encontraron coincidencias.

Este enfoque demuestra cómo los composables ayudan a reutilizar lógica y mantener el código más ordenado, ya que la misma funcionalidad puede aplicarse en distintos componentes sin necesidad de repetirla.



```

1 import { ref } from 'vue'
2
3 import { getAllContacts } from '../services/api.Contacs'
4
5 export const contactList = ref(getAllContacts())
6
7 export function useListaContacs() {
8
9   const msjEmpty = ref('')
10   //methods
11   const showEmpty = (msj) => {
12     if (contactList.value.length < 1) {
13       msjEmpty.value = msj
14     } else {
15       msjEmpty.value = ''
16     }
17   }
18
19   return {
20     msjEmpty,
21     showEmpty
22   }
23 }
  
```



```

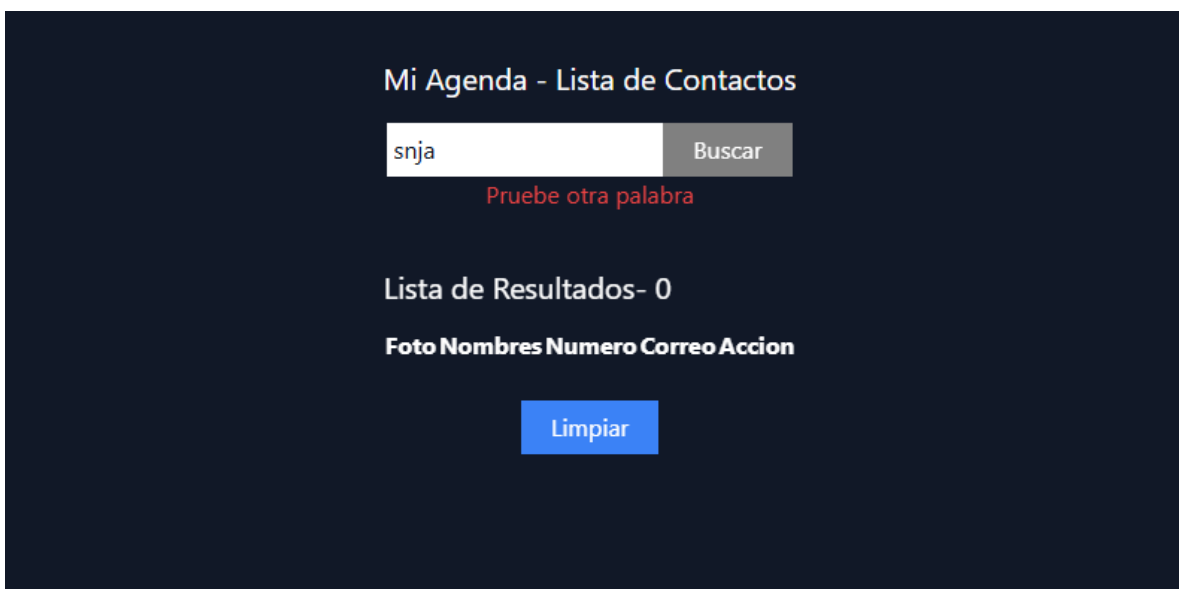
1 <script setup>
2 import { computed } from 'vue';
3 //import
4 import { contactList } from '../composables/useListaContacs';
5 import BtnDelete from '../Btn.Delete.vue';
  
```

```

EXPLORADOR  ...  App.vue  ListContacts.vue  useListaContacs.js  Btn.Delete.vue  SerachContacs.vue  api.Contacs.js
> EDITORES ABIERTOS
  AGENDAVUE
  > .vscode
  > dist
  > node_modules
  > public
  > src
  > assets
  > components
    > Btn.Delete.vue
  > App.vue  <script setup>
    1  <script setup>
    2    import { ref } from 'vue';
    3
    4    import SerachContacs from './components/SerachContacs.vue';
    5    import ListContacts from './components/ListContacts.vue';
    6    //
    7    import { getAllContacts } from './services/api.Contacs';
    8    import { contacList } from './composables/useListaContacs';
    9    //
   10
   11
  
```

```

27  const filtrarDatos =(value) => {
28    showEmpy('Pruebe otra palabra')
29    emit('filterContact', kword.value)
30
31  }
  
```



- Se hace lo mismo para listaContact

```

Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  ...  agendaVue
EXPLORADOR  ...  App.vue  ListContacts.vue  useListaContacs.js  Btn.Delete.vue  SerachContacs.vue
> EDITORES ABIERTOS
  AGENDAVUE
  > .vscode
  > dist
  > node_modules
  > public
  > src
  > assets
  > components
    > Btn.Delete.vue
    > ListContacts.vue
    > SerachContacs...
  > composables
    > useListaContac...
  > services
    > api.Contacs.js
    > App.vue
    > main.js
    > style.css
    > .gitignore
    > index.html
    > package-lock.json
    > package.json
    > portcss.config.js
  > components > ListContacts.vue <script setup> <watchEffect> callback
    1  <script setup>
    2    import { computed, watch, watchEffect } from 'vue';
    3    //import
    4    import { contacList } from '../composables/useListaContacs';
    5    import { useListaContacs } from '../composables/useListaContacs';
    6    import BtnDelete from './Btn.Delete.vue';
    7    //props
    8    const props = defineProps({
    9      title: {
   10        type: String,
   11        default: 'Lista'
   12      },
   13    });
   14    //states composables
   15    const { msjEmpy, showEmpy } = useListaContacs()
   16
   17    //watchers
   18    //watch(contacList, (newValue, oldValue) => {
   19      //console.log(newValue, oldValue)
   20      //showEmpy('No se encontro resultados')
   21    //})
   22
   23    watchEffect(() => {
   24      console.log(contacList.value)
   25      showEmpy('No se encontro resultados')
   26    })
  
```


Mi Agenda - Lista de Contactos

SAM

Buscar

Lista de Resultados- 0

Foto Nombres Numero Correo Accion

No se encontro resultados

Limpiar