

# DESENVOLVIMIENTO DE INTERFACES WEB 22\_23\_Ord

[Taboleiro](#) / [Os meus cursos](#) / [DIW 22 23 Ord](#) / [UD6. VUEJS](#) / [5. Vue Router.](#)

## 5. Vue Router.

### Vue-Router

**Vue-Router** es la **biblioteca de enrutamiento oficial del lado del cliente que proporciona las herramientas necesarias para asignar los componentes de una aplicación a diferentes rutas de URL del navegador.**

Permite crear Aplicaciones **de una sola página o Single Page Applications**. De esta manera tenemos una **navegación más fluida sin recargar la página**, ahorrando ancho de banda y agregando velocidad.

Creamos un proyecto, que llamaremos **vuerouter**.

Empezamos con algunas instalaciones.

Vue-Router

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\Carlos.PC-PEREZABALDE\WebstormProjects\vuerouter> npm install vue-router --save
```

Vue/Cli

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\Carlos.PC-PEREZABALDE\WebstormProjects\vuerouter> npm install --save-dev @vue/cli-service
```

Aunque no siempre es necesario instalamos *vue-router next*:

```
node.js v18.12.1
PS C:\Users\Carlos.PC-PEREZABALDE\WebstormProjects\vuerouter\src> npm install vue-router@next --save

added 23 packages, changed 1 package, and audited 678 packages in 8s

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

found 0 vulnerabilities
PS C:\Users\Carlos.PC-PEREZABALDE\WebstormProjects\vuerouter\src>
```

Nos preguntan si añadimos el modo history en el proyecto, le decimos que sí.

Añadimos router al proyecto.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\Carlos.PC-PEREZABALDE\WebstormProjects\vuerouter> vue add router
```

Para saber más sobre el **mode history y vue-router**, en general, [aquí](#)

## Configuración

Empezamos configurando las primeras **rutas estáticas**

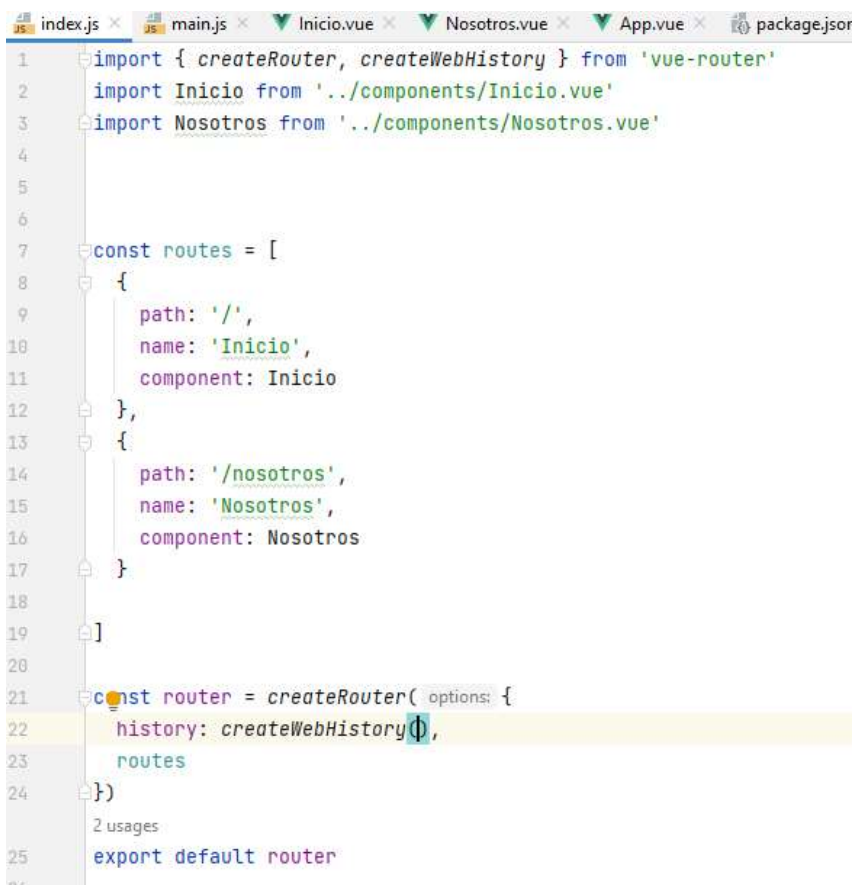
Para ello empezamos creando dos componentes **Nosotros.vue** e **Inicio.vue**

Fichero **main.js**



```
1 import { createApp } from 'vue'
2 import App from './App.vue'
3
4 import './assets/main.css'
5 import router from './router'
6
7 const app = createApp(App)
8
9 app.use(router)
10 app.mount( rootContainer: '#app')
```

Abramos el fichero **router/index.js**



```
1 import { createRouter, createWebHistory } from 'vue-router'
2 import Inicio from '../components/Inicio.vue'
3 import Nosotros from '../components/Nosotros.vue'
4
5
6
7 const routes = [
8   {
9     path: '/',
10    name: 'Inicio',
11    component: Inicio
12   },
13   {
14     path: '/nosotros',
15     name: 'Nosotros',
16     component: Nosotros
17   }
18 ]
19
20
21 const router = createRouter( options: {
22   history: createWebHistory(),
23   routes
24 } )
25 export default router
```

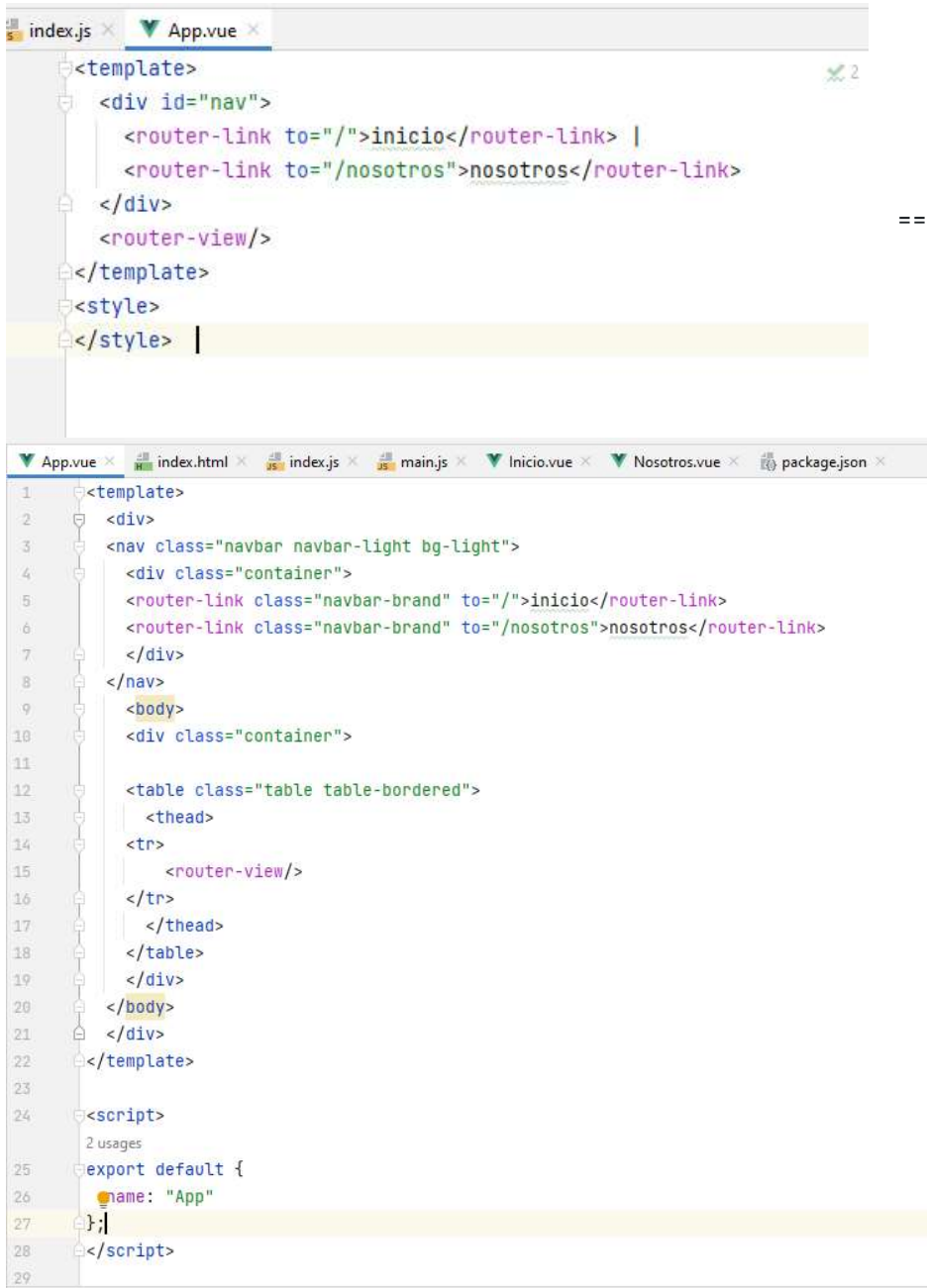
Vemos que estamos importando los módulos **createRouter** y **createWebHistory** de la librería **vue-router**.

En la línea 3 en adelante, hay un array que nos define dos rutas que son las de nuestra aplicación, de momento. La primera '/', significa que es la ruta inicial que **coincide que la URL base** de la página. La propiedad **name** representa el **nombre de la ruta**.

Finalmente instanciamos la creación de la rutas y exportamos para ser utilizada.

Fichero **el Vue.app**

### App.vue



```

<template>
  <div id="nav">
    <router-link to="/">inicio</router-link> |
    <router-link to="/nosotros">nosotros</router-link>
  </div>
  <router-view/>
</template>
<style>
</style>

```

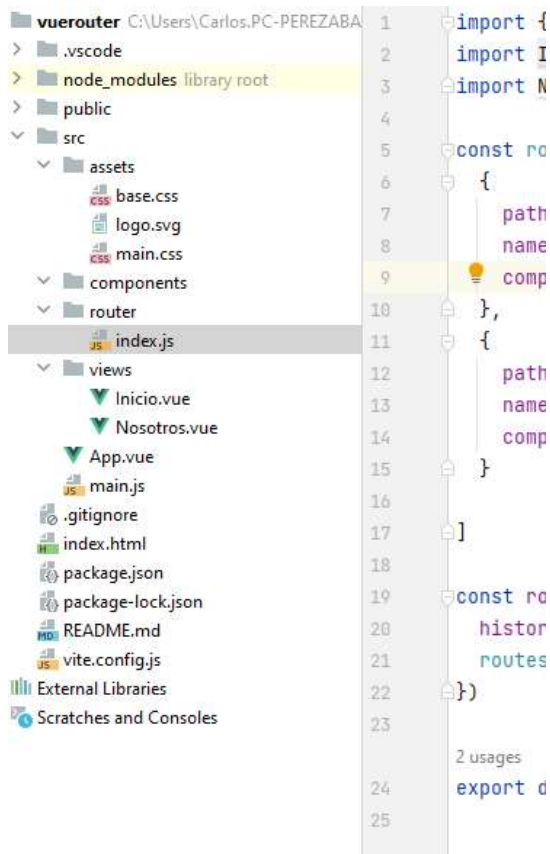
```

1 <template>
2 <div>
3 <nav class="navbar navbar-light bg-light">
4   <div class="container">
5     <router-link class="navbar-brand" to="/">inicio</router-link>
6     <router-link class="navbar-brand" to="/nosotros">nosotros</router-link>
7   </div>
8 </nav>
9 <body>
10  <div class="container">
11
12    <table class="table table-bordered">
13      <thead>
14        <tr>
15          <router-view/>
16        </tr>
17      </thead>
18    </table>
19  </div>
20 </body>
21 </div>
22 </template>
23
24 <script>
25   2 usages
26   export default {
27     name: "App"
28   };
29 </script>

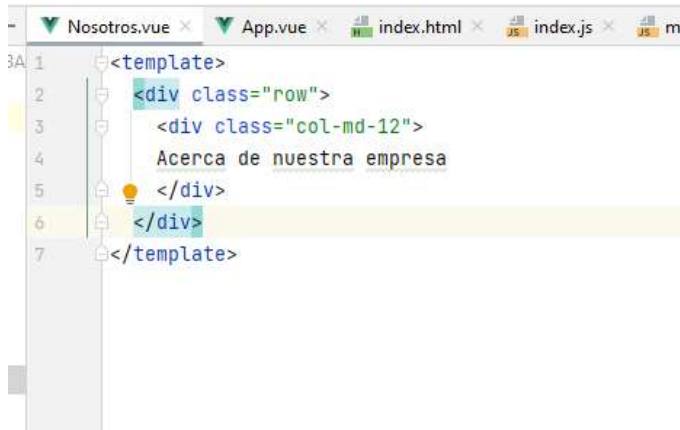
```

Fijémonos en el **atributo router-link, son enlaces de anclaje**. Sin embargo, a diferencia de un *enlace ancla* (etiqueta <a href="">), el **<enlace-enrutador> no recargará toda la página**. Recuerda que Vue es una aplicación de una sola página. **Los datos de la aplicación ya se han descargado del servidor**. Cuando enrutamos a otra vista, la aplicación simplemente **oculta cierta información y muestra la información solicitada**. Las etiquetas de enlace de enrutador tienen un nombre, que se refiere a qué página visitar. La etiqueta **<router-view/>** es lo que representa el componente correspondiente **cuando se activan los enlaces de navegación**.

Ahora necesitamos **crear las vistas (ver la segunda línea de routes/index.js)**. Vamos a crear un directorio **views** con su correspondientes ficheros **.vue**



Ejemplo de **vista**



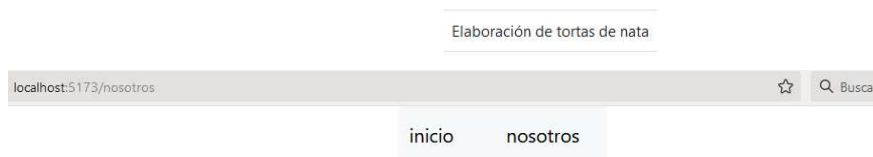
Fichero **index.html**

```

index.html x index.js x main.js x Inicio.vue x Nosotros.vue x App.vue x package.json x
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <link rel="icon" href="/favicon.ico">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css"
8     rel="stylesheet" integrity="sha384-GLhLTQ8iRABdZLL603oVMWSktQ0p6b7In1ZL3/Jr59b6EGGoI1aFkw7cmDA6j4
9     crossorigin="anonymous">
10  <title>Routes con Vue</title>
11 </head>
12 <body>
13   <div id="app"></div>
14   <script type="module" src="/src/main.js"></script>
15 </body>
16 </html>
17

```

## Resultados



## Rutas con parámetros

Vamos a crear una nueva vista o componente, por ejemplo, **articulos.vue**

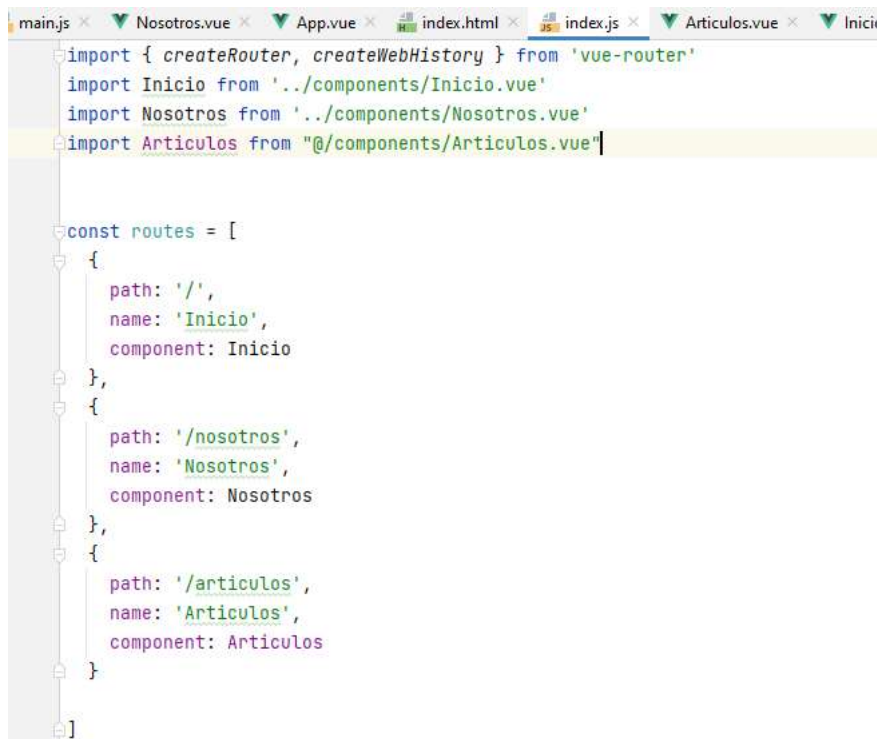


```

1 <template>
2   <div class="row">
3     <div class="col-md-12">
4       Articulos a la venta
5     </div>
6   </div>
7 </template>
8
9 <script>
10   no usages
11   export default {
12     name: "Articulos"
13   }
14 </script>
15 <style scoped>
16
17 </style>

```

Luego añadimos su ruta en... **routes/index.js**



```

import { createRouter, createWebHistory } from 'vue-router'
import Inicio from '../components/Inicio.vue'
import Nosotros from '../components/Nosotros.vue'
import Articulos from "@components/Articulos.vue"

const routes = [
  {
    path: '/',
    name: 'Inicio',
    component: Inicio
  },
  {
    path: '/nosotros',
    name: 'Nosotros',
    component: Nosotros
  },
  {
    path: '/articulos',
    name: 'Articulos',
    component: Articulos
  }
]

```

Finalmente, en **App.vue**

```

main.js x Nosotros.vue x App.vue x index.html x index.js x Articulos.vue x Inicio.vue x package.json x
<template>
  <div>
    <nav class="navbar navbar-light bg-light">
      <div class="container">
        <router-link class="navbar-brand" to="/">inicio</router-link>
        <router-link class="navbar-brand" to="/articulos">articulos</router-link>
        <router-link class="navbar-brand" to="/nosotros">nosotros</router-link>
      </div>
    </nav>
    <body>
      <div class="container">
        <table class="table table-bordered">
          <thead>
            <tr>
              <td><router-view/></td>
            </tr>
          </thead>
        </table>
      </div>
    </body>
  </div>
</template>

```

## Resultado



Una aplicación necesita a menudo **pasar parámetros** a través de la ruta a una vista. Vamos a ampliar la ruta **Articulos.vue** con el siguiente código.



```

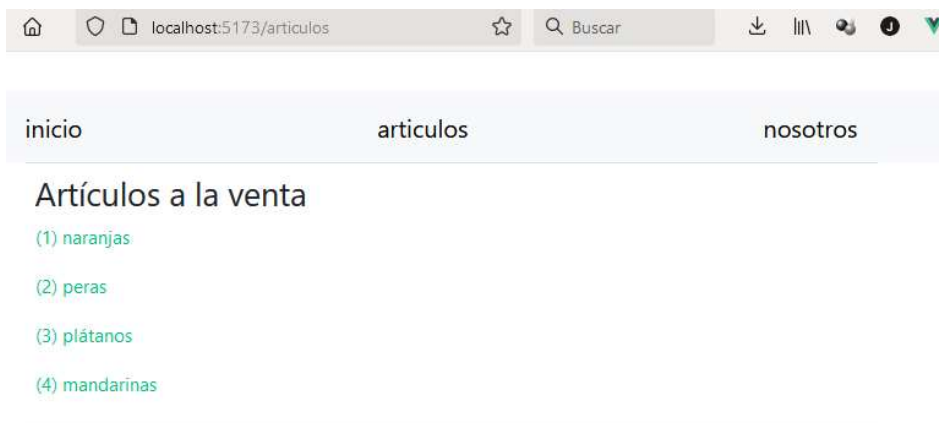
1 <template>
2   <div class="row">
3     <div class="col-md-12">
4       <h3>Artículos a la venta</h3>
5       <p v-for="articulo in articulos" :key="articulo.id">
6         <router-link :to=" {
7           name: 'Fruta',
8           params: { id: articulo.id, producto: articulo.producto, precio: articulo.precio }
9         }">
10          ({{articulo.id}}) {{ articulo.producto }}
11        </router-link></p>
12      </div>
13    </div>
14  </template>
15
16  <script>
17    2 usages
18    export default {
19      data() {
20        return {
21          articulos: [
22            {id: 1, producto: 'naranjas', precio: '1.25'},
23            {id: 2, producto: 'peras', precio: '1.19'},
24            {id: 3, producto: 'plátanos', precio: '2.25'},
25            {id: 4, producto: 'mandarinas', precio: '0.99'}
26          ]
27        }
28      }
29    }
30  </script>

```

Hemos añadido un router-link que **muestra enlaces** a una vista llamada **Fruta.vue**. Además tenemos un array de artículos cuyo id se lo pasamos al router-link para que muestre a su vez los datos de una fruta concreta.

Fijémonos detenidamente que la **directiva :to** lleva **puntos al inicio**, y necesita la vista Fruta y le pasa los parámetros de la fruta en cuestión.

Resultado inicial:



La vista **Fruta.vue** sería algo así:



```

1 <template>
2   <div class="row">
3     <div class="col-md-12">
4       <h4>Fruta del día:</h4>
5     </div>
6     <div class="col-md-6">
7       <p>Fruta id: {{ $route.params.id }}</p>
8       <p>Producto: {{ $route.params.producto }}</p>
9       <p>Precio: {{ $route.params.precio }}</p>
10    </div>
11  </div>
12 </template>

```

Si en Artículos pulsamos una fruta cualquiera el resultado sería:



Sin embargo la parametrización de componentes en Vue se realiza mediante el **atributo props**, que permite pasar **todo tipo de parámetros a la vista y de diferente tipo**.

Hagamos los cambios siguientes.

En **index.js**

```

{
  path: '/articulos/:id',
  name: 'Fruta',
  component: Fruta,
  props: true
}

```

### Como manejar el error 404 o Ruta No encontrada

A veces los usuarios introducen una página dentro de una web que no existe o bien que existió pero que se eliminó. El típico *Página no encontrada*

En las versiones de Vuex3 en adelante la gestión de dicha situación se hace de la forma siguiente:

En **routes/index.js** añadimos la siguiente ruta:

```

},
{
  path: '/*',
  name: 'NotFound',
  component: NotFound
}

```

Como vemos hay que **crear un componente** que llamaremos **NotFound.vue**. Algo así:

```

1 <template>
2   <div class="row">
3     <div class = 'col-md-12'>
4       Perdon, algo hemos hecho mal, pero esta página no existe
5       <hr>
6       <router-link class="btn btn-light-green" :to="{name: 'Inicio'}">Volver a Inicio</router-link>
7     </div>
8   </div>
9 </template>
10
11 <script>
12   export default {};
13 </script>

```

Básicamente, hemos decidido que si no se encuentra la página nos muestre un mensaje y un botón que nos devuelva a la ruta Inicio. También pudimos haber decidido otra.

localhost:5173/abotu

Ejercicios de CSS 3 | C... HTML/CSS Ejercicios, ... Europass Mobility Examen CSS HTML 4 ejemplos prácticos p... Studielink

inicio

articulos

nosotros

Perdon, algo hemos hecho mal, pero esta página no existe

Volver a Inicio

Y pulsando el botón nos devuelva a Inicio o a otra página.



[inicio](#) [articulos](#) [nosotros](#)

## Elaboración de Pasteles

Última modificación: Luns, 27 de Febreiro de 2023, 20:58

◀ 4. MEVN. MongoDB, Express, Vue y Node. Configuración del Frontend. ( II )

Ir a...

6. PINIA (Vuex5). Tienda Básica ▶

Vostede accedeu como Raúl Arias Pérez (Saír)  
DIW\_22\_23\_Ord

Resumen da retención de datos  
Obter a apli móbil