Guía de ejercicios - Vue Router (I)

 ¡Hola! Te damos la bienvenida a esta nueva guía de estudio.

¿En qué consiste esta guía?

La siguiente guía de estudio tiene como objetivo practicar y ejercitar los contenidos que hemos visto en clase.

¡Vamos con todo!

Tabla de contenidos

¡Comencemos!

Actividad guiada: Integración de vue router

A continuación, realizaremos un repaso de integración del sistema de rutas vue router en una aplicación vue js. Sigamos los pasos:

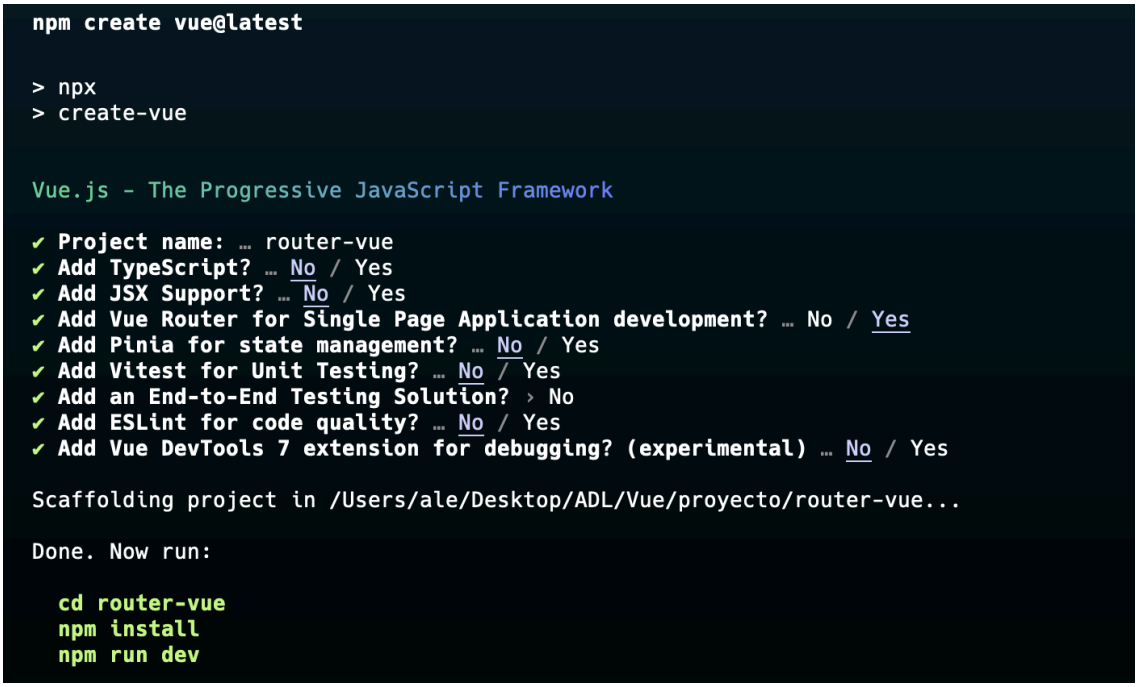
- **Paso 1:** Creamos una nueva aplicación con el nombre router-vue, para ello nos dirigimos a la terminal y ejecutamos el siguiente comando.

```
npm create vue@latest
```

- **Paso 2:** Agregamos el nombre del proyecto:

- **Paso 3:** Le decimos que no a configurar TypeScript y soporte de JSX

- **Paso 4:** Luego, agregamos vue router y a las demás configuraciones le decimos que no:



- **Paso 5:** Finalmente seguimos los pasos indicados en la consola.

Verificación de archivos integrados en la aplicación

Al ejecutar este conjunto de pasos, nuestra aplicación tendrá una estructura y un grupo de archivos que harán que el sistema de rutas funcione correctamente. Antes de utilizarlo es importante saber para qué son esos archivos y qué realizan a grandes rasgos.

Vue router como dependencia y plugin

En el archivo package.json veremos el grupo de dependencias que harán que nuestra aplicación funcione correctamente. Una de ellas es vue-router.

```
{
  "name": "router-vue",
  "version": "0.0.0",
  "private": true,
  "type": "module",
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "preview": "vite preview"
  },
```

```
  "dependencies": {
    "vue": "^3.4.29",
    "vue-router": "^4.3.3"
  },
  "devDependencies": {
    "@vitejs/plugin-vue": "^5.0.5",
    "vite": "^5.3.1"
  }
}
```

La estructura de carpetas que integra vue router

Al integrar el sistema de rutas en una aplicación Vue js automáticamente se integra un conjunto de directorios por defecto.

Recordemos que Vue js es un framework o marco de trabajo que entrega una estructura por defecto en cada uno de sus proyectos.

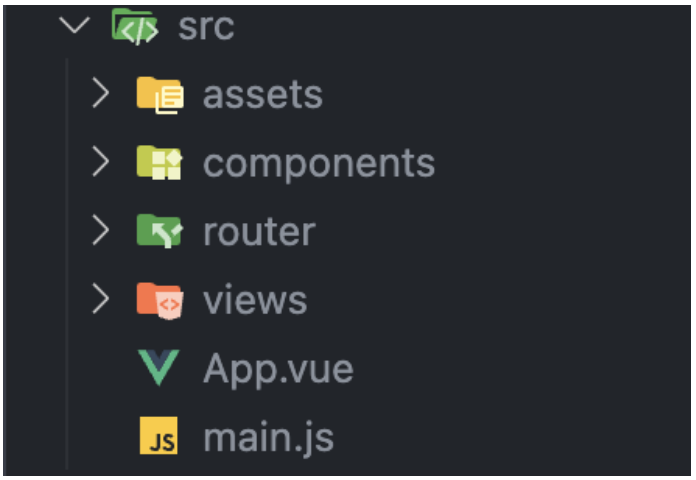


Imagen 1. Directorio router
Fuente: Desafío Latam

Como se observa en la imagen, en el directorio `src/` veremos entonces uno asociado al router. Esta en su interior contiene un archivo `index.js`.

Analicemos el archivo index.js

Para realizar este análisis iremos línea por línea contextualizando qué realiza dicho archivo.

1. Primero se realiza la importación de la dependencia de vue-router

```
import { createRouter, createWebHistory } from 'vue-router'
```

2. Posteriormente, importamos por defecto un componente llamado HomeView, el cual es integrado por defecto al momento de crear la aplicación con el sistema de rutas.

```
import HomeView from '../views/HomeView.vue'
```

3. Luego, tenemos una constante que representa una estructura de objeto, este en su interior recibe objetos asociados a cada ruta de la aplicación.

   Esta constante recibe el `createRouter` importado al inicio del documento y es un método. Este, recibe un objeto de configuración donde se le declara el history y las rutas.

```
const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'home',
      component: HomeView
    },
    {
      path: '/about',
      name: 'about',
      // route level code-splitting
      // this generates a separate chunk (About.[hash].js) for this
route
      // which is lazy-loaded when the route is visited.
      component: () => import('../views/AboutView.vue')
    }
  ]
})
```

En el momento que se genera la aplicación, vue-router integra por defecto dos rutas; `home` y `/about` y estas están representadas en un objeto que contiene ciertas propiedades.

El objeto de la ruta

La primera ruta `home` recibe:

- El `path` que es la ruta a la que apuntará dicho componente al ser clickeado.
- `name`, se refiere al nombre del componente que está enlazado a dicha ruta.
- `component`, establece el componente que estará asociado y en este caso recibe el mismo nombre de la importación.
- Esto mismo aplica para la ruta de `/about`.

¡Importante! Nótese la siguiente línea del objeto correspondiente a la ruta `/about`.

```
component: () => import('../views/AboutView.vue')
```

- Esta línea indica que la vista /about se mostrará pero a través de un callback.
- Este callback lo que genera es que el componente que muestra la información de esa vista solo se cargue al hacer clic en él.

Por último tenemos la exportación del router para ser usado en el resto de la aplicación:

```
export default router
```

Seguidamente, en el archivo main.js tendremos el llamado a esta exportación por defecto.

Archivo main.js.

```
import './assets/main.css'

import { createApp } from 'vue'
import App from './App.vue'
import router from './router'

const app = createApp(App)

app.use(router)

app.mount('#app')
```

Hasta este punto del aprendizaje, esta es la configuración que vue router integra en una aplicación para habilitar el sistema de enrutamiento.
Si levantamos el servidor con el comando `npm run dev` veremos el siguiente resultado.

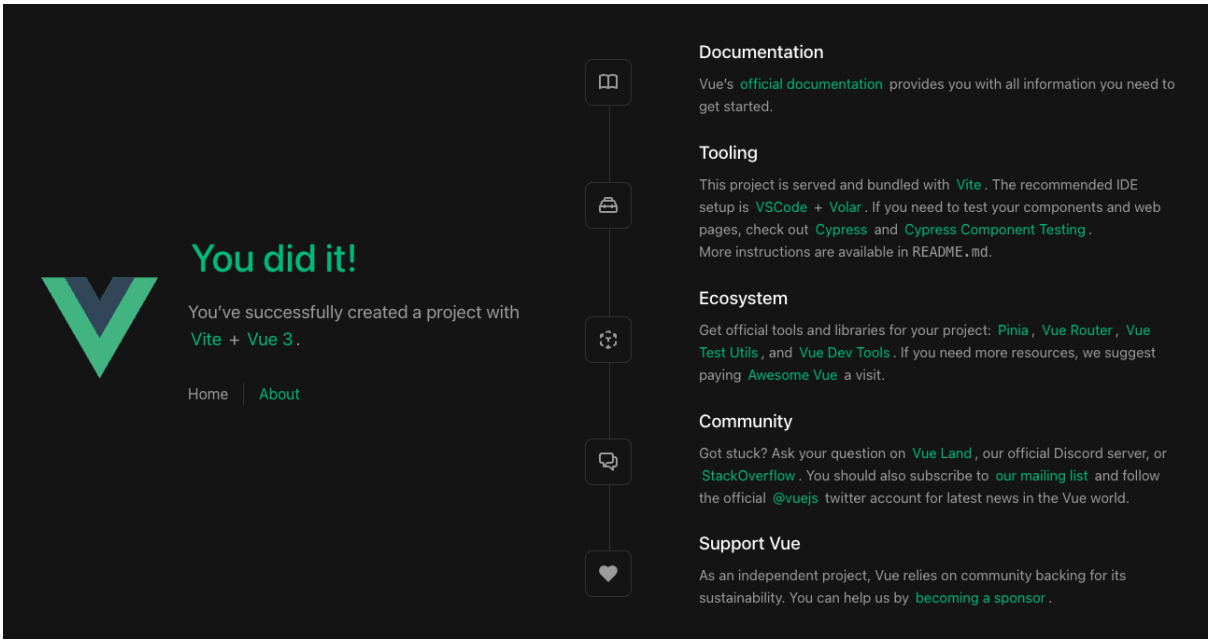


Imagen 2. Levantando el servidor
Fuente: Desafío Latam

¿Cómo se muestra esta información al levantar el servidor?

Cuando hacemos la integración de vue router mediante CLI se incorpora a la estructura de directorios una llamada /views. Dicho directorio es el que contiene los componentes que cargan la información de cada vista.

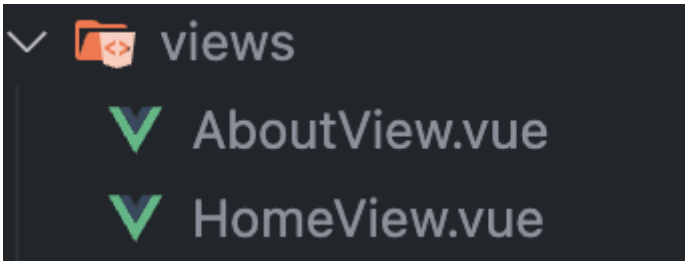


Imagen 3. Directorio /views
Fuente: Desafío Latam

En la parte inferior del costado izquierdo veremos que Home y About nos pueden redirigir dado que son enlaces, si damos clic en About veremos la siguiente información.

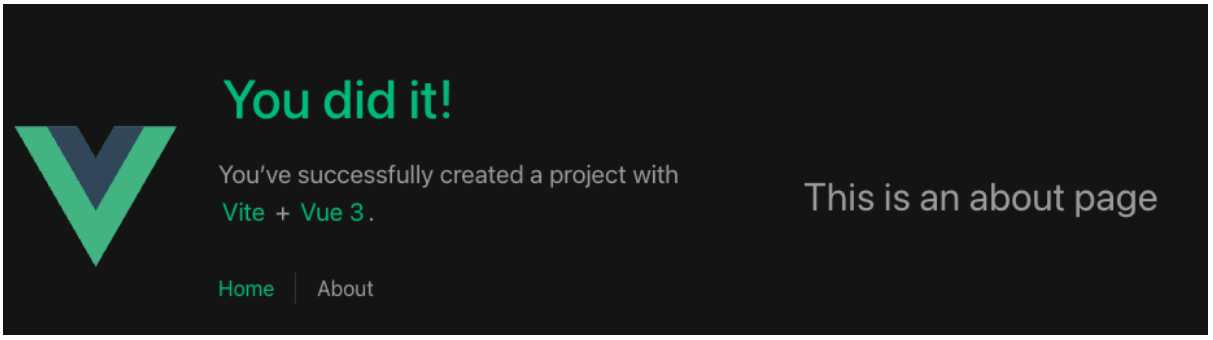


Imagen 4. Vista /about
Fuente: Desafío Latam

Si accedemos al archivo AboutView.vue en el editor de texto y modificamos el h1, podremos ver cambios en esta vista.

```
<template>
  <div class="about">
    <h1>Acerca de nosotros</h1>
  </div>
</template>
```

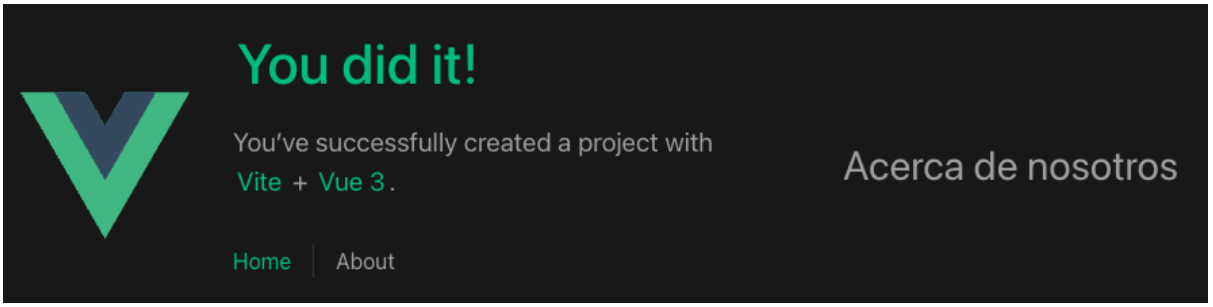


Imagen 5. Vista /about modificada
Fuente: Desafío Latam

¡Excelente! Hasta este punto ya conocemos algunas funcionalidades que integra vue router y cuáles son esos componentes de vue que muestran la información.
Recuerda que estos componentes son tratados como vistas.

Actividad guiada: Agregando una nueva vista a la aplicación

A continuación, veremos cómo agregar una nueva ruta a la aplicación que venimos trabajando desde la actividad anterior. Para ello realizaremos los siguientes pasos.

- **Paso 1:** Accedemos al archivo index.js del directorio /router.
- **Paso 2:** Hacemos una copia del objeto que controla la vista /about y lo cambiamos a contact.

Objeto para la vista contact:

```
{
    path: '/contact',
    name: 'contact',
    component: () => import(/* webpackChunkName: "about" */
'../views/Contact.vue')
},
```

Una vez que tenemos nuestro nuevo objeto que controlará la vista contact, debemos crear la vista que mostrará su información.

- **Paso 3:** Nos dirigimos al directorio /views y creamos un documento con el nombre Contact.vue y definimos la estructura para mostrar un título.

```
<template lang="">
    <h1>Información de contacto</h1>
</template>
```

Si guardamos, y revisamos en el navegador con el servidor levantado, veremos el siguiente resultado.

- **Paso 3:** Nos dirigimos al archivo **App.vue** y agregamos un item RouterLink dentro del Nav, para poder acceder a nuestra vista recién creada:

```
<nav>
     <RouterLink to="/">Home</RouterLink>
     <RouterLink to="/about">About</RouterLink>
     <RouterLink to="/contact">Contacto</RouterLink>
</nav>
```

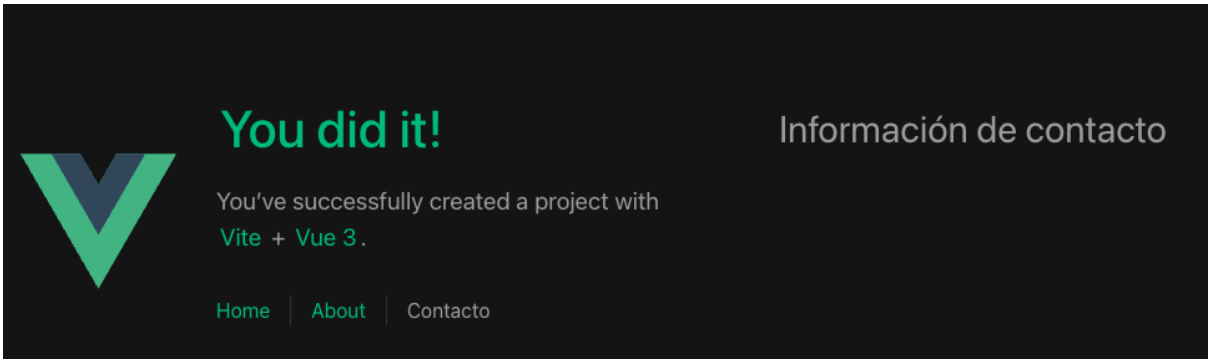


Imagen 6. Vista contact
Fuente: Desafío Latam

De este modo, hemos incorporado una nueva vista a la aplicación, recordemos los pasos:

- Modificamos el archivo index.js del router y agregamos la ruta que necesitemos, recordemos que es un objeto con diferentes propiedades.
- En el directorio `/views` agregamos el nuevo componente que se enlazará dicha vista.
- Modificamos el archivo App.vue para agregar el enlace a nuestra vista.

¡Manos a la obra! - Genera una nueva vista en la app

Una página web en Vue js puede tener tantas vistas como se consideren necesarias, es por ello que a partir de la creatividad te invitamos a crear una nueva vista a la aplicación trabajada en esta guía de ejercicios.

- Recuerda modificar el archivo `index.js` del router.
- Integra el nuevo objeto de la ruta con un componente en el directorio `/views`.
- En el componente de la vista nueva muestra al menos un título.

Solución manos a la obra

- Objeto para la nueva vista en el archivo index.js.

```
{
    path: '/products',
    name: 'products',
    component: () => import(/* webpackChunkName: "about" */
'../views/Products.vue')
},
```

- Componente de la vista Products.

```
<template lang="">
    <h1>Información de los productos</h1>
</template>
```