Manual del Programador: Desarrollo de una Página Web

Introducción

Este documento detalla la configuración técnica y los procedimientos necesarios para desarrollar, mantener y extender la aplicación web. Se describen la estructura del proyecto, las rutas configuradas en Vue.js, la instalación de dependencias y los pasos para compilar y ejecutar el proyecto.

Estructura de Directorios

El repositorio del proyecto se organiza de la siguiente forma:

/:
  - Archivos de configuración base, incluyendo `package.json`, `README.md` y configuración de Webpack.
/src/:
  - Código fuente principal de la aplicación.
/src/router/:
  - Configuración de las rutas de Vue Router.
/src/modules/:
  - Módulos organizados por funcionalidad (usuarios, carrito, productos).
/src/modules/user/:
  - Código relacionado con la gestión de usuarios (rutas, vistas y lógica).
/src/modules/cart/:
  - Código relacionado con el carrito de compras.
/src/modules/product/:
  - Código relacionado con los productos (categorías, detalles de productos, etc.).
/src/pages/:
  - Componentes de páginas individuales, cargados dinámicamente por las rutas.
/public/:
  - Recursos estáticos como imágenes y el archivo `index.html`.

Entorno de Desarrollo

Para trabajar con el proyecto es necesario tener instalado:

1. Node.js (v16 o superior):
Administrado con NVM para garantizar compatibilidad.
Instalar NVM: Guía oficial de NVM.

2. Vue CLI:
Herramienta para visualizar y gestionar proyectos Vue.js.
Instalar con:
npm install -g @vue/cli

3. Git:
Para gestionar el repositorio.
Instalar desde: Git.

Obtención del Código Fuente

Clonar el repositorio desde GitHub:

git clone git clone https://github.com/Bussy888/ecommerce-fendermed
cd ecommerce-fendermed

Estructura de Rutas

La aplicación utiliza Vue Router con un enfoque modular. Las rutas están organizadas en módulos específicos:

1. Archivo principal de rutas (index.js): Combina las rutas definidas en cada módulo:
javascript
Copy code

```
import { createRouter, createWebHashHistory } from 'vue-router';
import userRoutes from '@/modules/user/routes';
import cartRoutes from '@/modules/cart/routes';
import productRoutes from '@/modules/product/routes';

const routes = [
  ...cartRoutes,
  ...productRoutes,
  ...userRoutes
];

const router = createRouter({
```

```
  history: createWebHashHistory(),
  routes
});

export default router;
```

2. Módulo de Producto (src/modules/product/routes/index.js): Define las rutas para la funcionalidad de productos:

```
export default [
  { path: '/', name: 'Home', component: () => import('../pages/Home.vue') },
  { path: '/about', name: 'About', component: () => import('../pages/About.vue') },
  { path: '/category', name: 'Category', component: () => import('../pages/Category.vue') },
  { path: '/product/:id', name: 'Product', component: () => import('../pages/Product.vue') },
];
```

3. Módulo de Usuario (src/modules/user/routes/index.js): Define las rutas relacionadas con la gestión de usuarios:

```
export default [
  { path: '/login', name: 'Login', component: () => import('../pages/Login.vue') },
  { path: '/register', name: 'Register', component: () => import('../pages/Register.vue') },
  { path: '/reset-pass', name: 'ResetPass', component: () => import('../pages/ResetPass.vue') },
];
```

4. Módulo del Carrito (src/modules/cart/routes/index.js): Contiene una única ruta para el carrito:

```
export default [
  { path: '/cart', name: 'Cart', component: () => import('../pages/Cart.vue') },
];
```

Instalación de Dependencias

Ejecutar los siguientes comandos para instalar las dependencias necesarias:

```
npm install
```

Compilación y Ejecución

1. **Modo Desarrollo:**
   npm run serve
   Esto iniciará un servidor local accesible desde http://localhost:8080.

2. **Compilación para Producción:**

   npm run build

   Genera archivos optimizados para producción en la carpeta /dist.

Testing

Las pruebas unitarias se configuran utilizando Jest o Cypress. Para ejecutar los tests:

npm run test:unit

Actualización de Dependencias

Actualizar las dependencias utilizando:

npm outdated
npm update

Añadir Nuevas Funcionalidades

Para añadir una nueva funcionalidad:

1. Crear una nueva rama desde main:
   git checkout -b feature/nueva-funcionalidad
2. Crear o modificar las rutas en su módulo correspondiente.
3. Añadir los componentes necesarios en src/modules/[modulo]/pages.
4. Realizar commits siguiendo buenas prácticas:
   git commit -m "Añadir nueva funcionalidad: [descripción]"
5. Fusionar los cambios en main:
   bash
   Copy code
   git checkout main

```
git merge feature/nueva-funcionalidad
git push origin main
```