Atelier 7 : Datatable et Modal

PrimeVue est un riche ensemble de composants d'interface utilisateur open source pour Vue. On va en utiliser son Datatable qui affiche les données sous forme de tableau avec plusieurs options. Pour l’ajout et la modification on va utiliser les Modals.

***Références*** :

<https://primevue.org/installation/>

<https://primevue.org/datatable/>

<https://primevue.org/icons/#import>

<https://primevue.org/dialog/>

# PrimeVue Datatble

## Installation

Installer le package de PrimeVue

npm install primevue

npm install primeicons

Le plugin PrimeVue doit être installé avec la fonction use pour configurer la configuration par défaut.



import axios from "./components/config/axios";

import { createApp } from 'vue';

import { createRouter, createWebHashHistory } from "vue-router";

import { routes } from "./routes";

import 'bootstrap/dist/css/bootstrap.min.css'

import PrimeVue from 'primevue/config';

import 'primevue/resources/themes/lara-light-teal/theme.css'

import App from './App.vue'

let app = createApp(App)

const router = createRouter({

    history: createWebHashHistory(),

    routes: routes,

})

app.use(router).use(PrimeVue);

app.mount("#app");

## Affichage

Sous /components créer le dossier scategories contenant le fichier viewScategories.vue



<template>

    <div class="card">

        <DataTable v-model:filters="filters" :value="scategories" paginator :rows="10" dataKey="id" filterDisplay="row" :loading="isLoading"

                :globalFilterFields="['nomscategorie']">

            <template #header>

                <div class="flex justify-content-end">

                    <span class="p-input-icon-left">

                        <i class="pi pi-search" />

                        <InputText v-model="filters['global'].value" placeholder="Keyword Search" />

                    </span>

                </div>

            </template>

            <template #empty> No scategories found. </template>

            <template #loading> Loading scategories data. Please wait. </template>

           <Column field="nomscategorie" header="Nom S/Catégorie" style="min-width: 12rem" :sortable="true">

                <template #body="{ data }">

                    {{ data.nomscategorie }}

                </template>

                <template #filter="{ filterModel, filterCallback }">

                    <InputText v-model="filterModel.value" type="text" @input="filterCallback()" class="p-column-filter" placeholder="Search by name" />

                </template>

            </Column>

            <Column header="Image" style="min-width: 12rem">

                <template #body="{ data }">

                    <div class="flex align-items-center gap-2">

                        <img :src="data.imagescategorie" :alt=data.nomscategorie  width="90" />

                    </div>

                </template>

            </Column>

            <Column field="id" header="Actions" style="min-width: 12rem">

               <template #body="slotProps">

        <button type="button" class="btn btn-warning rounded-circle btn-sm" @click="deleteSCategorie(slotProps.data.id)">

            <i class="pi pi-trash" style="color: black"></i>

        </button>

            </template>

            </Column>

        </DataTable>

    </div>

</template>

<script setup>

import { ref, onMounted } from 'vue';

import { FilterMatchMode } from 'primevue/api';

import axios from "../config/axios.js";

import Column from 'primevue/column';

import DataTable from 'primevue/datatable';

import InputText from 'primevue/inputtext';

import 'primeicons/primeicons.css'

const scategories = ref([]);

const isLoading=ref(true);

const filters = ref({

    global: { value: null, matchMode: FilterMatchMode.CONTAINS },

    'nomscategorie': { value: null, matchMode: FilterMatchMode.CONTAINS}

});

const fetchSCategories=async ()=> {

        await axios

        .get('/api/scategories/')

            .then((response)=>{

                isLoading.value=false;

                scategories.value = response.data

            })

        .catch ((error) =>{

            console.error('Error fetching categories:', error);

        });

}

onMounted(() => {

    fetchSCategories();

});

const deleteSCategorie=async (id)=> {

             await axios

                    .delete(`/api/scategories/${id}`)

                    .then(() => {

                        let i = scategories.value.map(data => data.id).indexOf(id);

                        scategories.value.splice(i, 1)

                    });

                }

</script>

Créer la route :



import ViewScategories from './components/scategories/ViewScategories.vue';

…

{

        name: 'scategories',

        path: '/scategories',

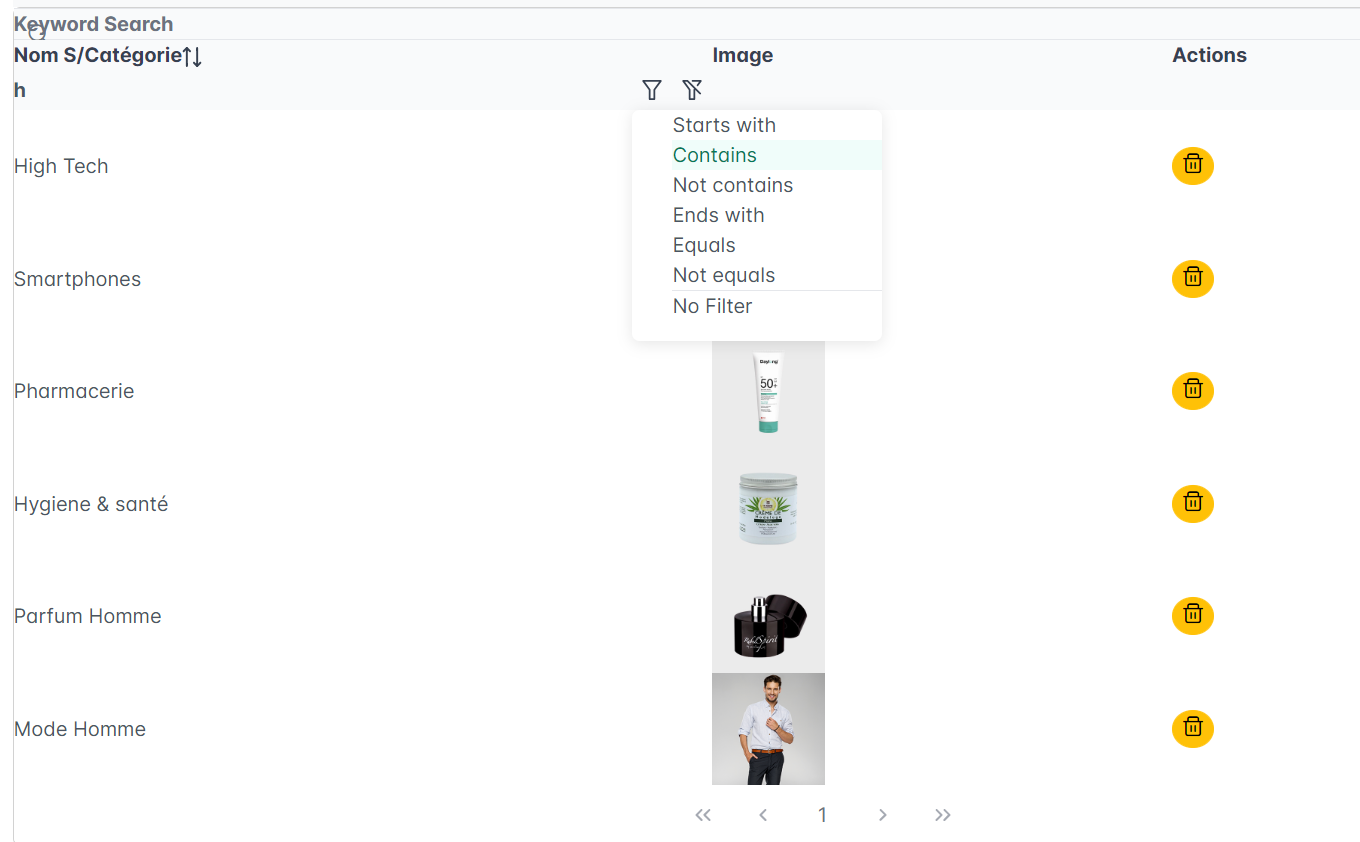
        component: ViewScategories

    },

Ajouter dans le menu



<router-link to="/scategories" class="nav-item nav-link">S/Categories</router-link>



# PrimeVue Dialog

Modifier la liste



<template>

**<addEditScaregories :scategories="scategories" :isEditing=false />**

    <div class="card">

        <DataTable v-model:filters="filters" :value="scategories" paginator :rows="10" dataKey="id" filterDisplay="row" :loading="isLoading"

                :globalFilterFields="['nomscategorie']">

            <template #header>

                <div class="flex justify-content-end">

                    <span class="p-input-icon-left">

                        <i class="pi pi-search" />

                        <InputText v-model="filters['global'].value" placeholder="Keyword Search" />

                    </span>

                </div>

            </template>

            <template #empty> No scategories found. </template>

            <template #loading> Loading scategories data. Please wait. </template>

           <Column field="nomscategorie" header="Nom S/Catégorie" style="min-width: 12rem" :sortable="true">

                <template #body="{ data }">

                    {{ data.nomscategorie }}

                </template>

                <template #filter="{ filterModel, filterCallback }">

                    <InputText v-model="filterModel.value" type="text" @input="filterCallback()" class="p-column-filter" placeholder="Search by name" />

                </template>

            </Column>

            <Column header="Image" style="min-width: 12rem">

                <template #body="{ data }">

                    <div class="flex align-items-center gap-2">

                        <img :src="data.imagescategorie" :alt=data.nomscategorie  width="90" />

                    </div>

                </template>

            </Column>

            <Column field="id" header="Actions" style="min-width: 12rem; display: inline;">

            <template #body="slotProps">

**<div class="d-flex">**

**<addEditScaregories :scategories="scategories" :isEditing=true :scategorie="slotProps.data" />**

                <button type="button" class="btn btn-warning rounded-circle btn-sm" @click="deleteSCategorie(slotProps.data.id)">

                    <i class="pi pi-trash" style="color: black"></i>

                </button>

**</div>**

            </template>

            </Column>

</DataTable>

    </div>

</template>

<script setup>

import addEditScaregories from './addEditScaregories.vue';

import { ref, onMounted } from 'vue';

import { FilterMatchMode } from 'primevue/api';

import axios from "../config/axios.js";

import Column from 'primevue/column';

import DataTable from 'primevue/datatable';

import InputText from 'primevue/inputtext';

import 'primeicons/primeicons.css'

const scategories = ref([]);

const isLoading=ref(true);

const filters = ref({

    global: { value: null, matchMode: FilterMatchMode.CONTAINS },

    'nomscategorie': { value: null, matchMode: FilterMatchMode.CONTAINS}

});

const fetchSCategories=async ()=> {

        await axios

        .get('/api/scategories/')

            .then((response)=>{

                isLoading.value=false;

                scategories.value = response.data

            })

        .catch ((error) =>{

            console.error('Error fetching categories:', error);

        });

}

onMounted(() => {

    fetchSCategories();

});

const deleteSCategorie=async (id)=> {

             await axios

                    .delete(`/api/scategories/${id}`)

                    .then(() => {

                        let i = scategories.value.map(data => data.id).indexOf(id);

                        scategories.value.splice(i, 1)

                    });

                }

</script>

Ensuite créer le composant



L’élément Dialog de primevue est un modal qui permet d’afficher le formulaire selon la valeur de isEditing envoyée en props. Pour savoir s’il s’agit d’un cas d’ajout ou d’edit.

<template>

<button type="button" :class="isEditing ? 'btn btn-success rounded-circle btn-sm' : 'btn btn-success btn-sm'" @click="visible = true">

  <span style="color: white">

    <i v-if="isEditing" class="pi pi-pencil" style="color: white"></i>

    <i v-else class="pi pi-plus" style="color: white"></i>

    {{ isEditing ? null : "Nouveau" }}

  </span>

</button>

    <div class="card flex justify-content-center">

        <Dialog

            v-model:visible="visible"

            modal

            :pt="{

                mask: {

                    style: 'backdrop-filter: blur(2px)'

                }

            }"

        >

            <template #container="{ closeCallback }">

                <div class="flex flex-column px-8 py-5 gap-4" style="border-radius: 12px; background-image: radial-gradient(circle at left top, var(--primary-200), var(--primary-300))">

                    <div class="inline-flex flex-column gap-2">

                        <label for="nomscategorie" class="text-primary-50 font-semibold">Nom</label>

                        <InputText v-model="scategorie.nomscategorie" id="nomscategorie" class="bg-white-alpha-20 border-none p-3 text-primary-50"></InputText>

                    </div>

                    <div class="inline-flex flex-column gap-2">

                        <label for="imagescategorie" class="text-primary-50 font-semibold">Image</label>

                        <InputText v-model="scategorie.imagescategorie" id="imagescategorie" class="bg-white-alpha-20 border-none p-3 text-primary-50" type="imagescategorie"></InputText>

                    </div>

                    <div class="inline-flex flex-column gap-2">

                       <label for="categorieID" class="text-primary-50 font-semibold">Categorie</label>

                        <select v-model="scategorie.categorieID" class="form-control">

                        <option v-for="c in categories" :key="c.id" :value=c.id style="color: black">{{c.nomcategorie}}</option>

                        </select>

                    </div>

                    <div class="flex align-items-center gap-2">

                        <Button label="Enregistrer" @click="saveSCategory" text class="p-3 w-full text-primary-50 border-1 border-white-alpha-30 hover:bg-white-alpha-10"></Button>

                        <Button label="Annuler" @click="closeCallback" text class="p-3 w-full text-primary-50 border-1 border-white-alpha-30 hover:bg-white-alpha-10"></Button>

                    </div>

                </div>

            </template>

        </Dialog>

    </div>

</template>

<script setup>

import { ref,onMounted } from "vue";

import axios from "../config/axios.js";

import InputText from 'primevue/inputtext';

import Button from 'primevue/button';

import Dialog from 'primevue/dialog';

import 'primeicons/primeicons.css'

const visible = ref(false);

let scategorie=ref({id:'',nomscategorie:'',imagescategorie:'',categorieID:''});

const props = defineProps(['scategories','isEditing','scategorie'])

const saveSCategory = async () => {

  if (props.isEditing) {

    // Mettre à jour la scatégorie existante

    await axios

                    .put('/api/scategories/'+scategorie.value.id, scategorie.value)

                    .then(() => {

                        const scategoryIndex = props.scategories.findIndex((scategory) => scategory.id === scategorie.value.id);

                        if (scategoryIndex !== -1) {

                            props.scategories[scategoryIndex] = scategorie.value;

                        }

                     })

                    .catch(err => console.log(err))

    } else {

    // Ajouter une nouvelle scatégorie

    await axios

                    .post('/api/scategories', scategorie.value)

                    .then(() => (

                        props.scategories.push(scategorie.value)

                    ))

                    .catch(err => console.log(err))

                }

   visible.value=false;

};

const categories = ref([]);

const getCategories = () => {

  axios.get('/api/categories')

    .then(res => {

      categories.value = res.data;

    })

    .catch(error => {

      console.log(error);

    });

}

onMounted(() => {

  getCategories();

  if (props.isEditing) {

    scategorie.value=props.scategorie;

    }

});

</script>

