Como crear un crud en esta aplicación  
  
1) **crear el modelo:** adonis make:model Marca -m  
2) **modificar la migración** creada y realizar las relaciones en el modelo de ser necesario.  
3) **crear el controlador:** adonis make:controller Marca (http)   
4) **copiar de un controlar existente** las sentencias para el crud

'use strict'

const Marca = use('App/Models/Marca')

const AuthorizationService = use('App/Services/AuthorizationService')

class MarcaController {

  async index ({ auth }) {

    const user = await auth.getUser()

    return await Marca.all()

  }

  async create ({ auth, request }) {

    const user = await auth.getUser()

    const { nombre, activo } = request.all()

    const marca = new Marca()

    AuthorizationService.verifyPermission(marca, user)

    marca.fill({nombre,activo})

    await marca.save(marca)

    return marca

  }

  async destroy ( { auth, request, params }) {

    const user = await auth.getUser()

    const { id } = params

    const marca = await Marca.find(id)

    AuthorizationService.verifyPermission(marca, user)

    await marca.delete()

    return marca

  }

  async update ( { auth, request, params }) {

    const user = await auth.getUser()

    const { id } = params

    const marca = await Marca.find(id)

    AuthorizationService.verifyPermission(marca, user)

    marca.merge(request.only(['nombre','activo']))

    await marca.save()

    return marca

  }

}

module.exports = MarcaController

5) agregar las rutas en server\routes.js

Route.get('marcas','MarcaController.index').middleware('auth')

  Route.post('marcas','MarcaController.create').middleware('auth')

  Route.delete('marcas/:id','MarcaController.destroy').middleware('auth')

  Route.patch('marcas/:id','MarcaController.update').middleware('auth')

6) agregar en App.vue la opción correspondiente

<v-list-item link :to="{name: 'marcas'}">

<v-list-item-title>

       <v-icon>verified\_user</v-icon>

      Marcas

      </v-list-item-title>

</v-list-item>

7) crear la vista correspondiente en en \view\ y realizar las modificaciones necesarias

<template>

  <v-layout align-start>

    <v-flex>

      <v-data-table

        :headers="headers"

        :items="marcas"

        :search="search"

        sort-by="id"

        dense

        class="elevation-3"

        :footer-props="footerProps">

        <template v-slot:top>

          <v-system-bar color="indigo darken-2" dark></v-system-bar>

          <v-toolbar flat color="indigo">

            <template v-slot:extension>

              <v-btn

                fab

                color="cyan accent-2"

                bottom

                left

                absolute

                @click="dialog = !dialog">

                <v-icon>mdi-plus</v-icon>

              </v-btn>

            </template>

            <v-toolbar-title class="white--text">Marcas de Artículos</v-toolbar-title>

              <v-divider class="mx-4" inset vertical></v-divider>

            <v-spacer></v-spacer>

            <!--  Modal del diálogo para Alta y Edicion    -->

            <v-dialog v-model="dialog" max-width="500px">

              <template v-slot:activator="{ on }"></template>

              <v-card>

              <!-- para el EDICION-->

              <v-card-title  class="cyan white--text">

                  <span class="headline">{{ formTitle }}</span>

                </v-card-title>

                <v-card-text>

                  <v-container>

                    <v-row>

                      <!--EL ID NO SE MODIFICA YA QUE ES AUTOINCREMENTAL EN LA BASE DE DATOS-->

                      <!--<v-col cols="12" sm="6" md="4">

                        <v-text-field v-model="editado.id" label="ID"></v-text-field>

                      </v-col>-->

                      <v-col cols="6" sm="6" md="4">

                        <v-text-field

                          autofocus

                          v-model="editado.nombre"

                          label="Marca">

                        </v-text-field>

                      </v-col>

                      <v-col cols="6" sx="6" mx="4">

                        <v-switch

                          label="Activo"

                          v-model="editado.activo">

                        </v-switch>

                      </v-col>

                  </v-row>

                  </v-container>

                </v-card-text>

                <v-card-actions>

                  <v-spacer></v-spacer>

                  <v-btn

                    color="blue-grey"

                    class="ma-2 white--text"

                    @click="cancelar">

                    Cancelar

                  </v-btn>

                  <v-btn

                    color="teal accent-4"

                    class="ma-2 white--text"

                    @click="guardar">

                    Guardar

                  </v-btn>

                </v-card-actions>

              </v-card>

            </v-dialog>

          </v-toolbar>

            <!-- Barra de búsqueda  -->

            <v-col cols="12" sm="12">

              <v-text-field

                v-model="search"

                append-icon="search"

                label="Buscar"

                single-line hide-details>

              </v-text-field>

            </v-col>

        </template>

        <template v-slot:item.activo="{ item }">

          <v-chip :color="getColor(item.activo)" dark>{{ item.activo }}</v-chip>

        </template>

        <template v-slot:item.accion="{ item }">

          <v-btn class="mr-2" fab dark x-small color="cyan" @click="editar(item)">

          <v-icon dark>mdi-pencil</v-icon>

          </v-btn>

          <v-btn class="mr-2" size="36" fab dark x-small color="error" @click="borrar(item)">

          <v-icon dark>mdi-delete</v-icon>

          </v-btn>

        </template>

      </v-data-table>

      <!-- template para el snackbar-->

      <template>

          <div class="text-center ma-2">

          <v-snackbar

            v-model="snackbar"

            timeout:1000>

            {{ textSnack }}

            <v-btn color="info" text @click="snackbar = false">Cerrar</v-btn>

          </v-snackbar>

          </div>

      </template>

    </v-flex>

  </v-layout>

</template>

<script>

/\* eslint-disable \*/

import HTTP from '../http';

import { mapGetters } from 'vuex';

import router from '../router';

export default {

  data: () => ({

      footerProps: {'items-per-page-options': [6, 9, 12, 15, 100]},

      search: '', // para el cuadro de búsqueda de datatables

      snackbar: false, // para el mensaje del snackbar

      textSnack: 'texto snackbar', // texto que se ve en el snackbar

      dialog: false, // para que la ventana de dialogo o modal no aparezca automáticamente

      // definimos los headers de la datatables

      marcas: [],

      headers: [

        {

          text: 'ID',

          align: 'left',

          sortable: false,

          value: 'id',

        },

        { text: 'NOMBRE', value:'nombre'},

        { text: 'ACTIVO', value:'activo'},

        { text: 'ACCIONES', value: 'accion', sortable: false },

      ],

      // marcas: [], //definimos el array marcas

      editedIndex: -1,

      editado: {

        id: '',

        nombre: '',

        activo: false,

      },

      defaultItem: {

        id: '',

        nombre: '',

        activo: false,

      },

    }),

    computed: {

      // Dependiendo si es Alta o Edición cambia el título del modal

      ...mapGetters('authentication', ['isLoggedIn']),

      formTitle () {

        // operadores condicionales "condición ? expr1 : expr2"

        // si <condicion> es true, devuelve <expr1>, de lo contrario devuelve <expr2>

        return this.editedIndex === -1 ? 'Nuevo Registro' : 'Editar Registro';

      },

    },

    watch: {

      dialog (val) {

        val || this.cancelar();

      },

    },

    mounted () {

      if (!this.isLoggedIn) {

        return router.push('/login');

      } else {

        return HTTP().get('/marcas')

          .then(({ data }) => {

            this.marcas = data;

          });

      }

    },

    created () {

      this.listarMarcas();

    },

    methods: {

      // PROCEDIMIENTOS para el CRUD

      // Procedimiento Listar moviles

      getColor (activo) {

        if (activo === 1) return 'red'

        else if (activo === 0) return 'orange'

        else return 'green'

      },

      listarMarcas:function() {

        return HTTP().get('/marcas')

          .then(({ data }) => {

            this.marcas = data;

        });

      },

      //Procedimiento Alta de moviles.

      altaMovil:function() {

        return HTTP().post('/marcas', {

          nombre: this.nombre,

          activo: true,

          }).then(({ data }) => {

            this.listarMarcas();

          });

      },

      // Procedimiento EDITAR.

      editarMovil:function(data) {

        return HTTP().patch(`marcas/${data.id}`, data)

          .then(() => {

            this.listarMarcas();

          });

      },

      // Procedimiento BORRAR.

      borrarMovil:function(id) {

        return HTTP().delete(`/marcas/${id}`)

          .then(() => {

            this.listarMarcas();

          });

      },

      editar (item) {

        this.editedIndex = this.marcas.indexOf(item);

        debugger;

        this.editado = Object.assign({}, item);

        this.dialog = true;

      },

      borrar (item) {

        const index = this.marcas.indexOf(item);

        var r = confirm('¿Está seguro de borrar el registro?');

        if (r === true) {

          this.borrarMovil(this.marcas[index].id);

          this.snackbar = true;

          this.textSnack = 'Se eliminó el registro.';

        } else {

          this.snackbar = true;

          this.textSnack = 'Operación cancelada.';

        }

      },

      cancelar() {

        this.dialog = false;

        this.editado = Object.assign({}, this.defaultItem);

        this.editedIndex = -1;

      },

      guardar() {

        this.snackbar = true;

        if (this.editedIndex > -1) {

          // Guarda en caso de Edición

          this.id = this.editado.id;

          this.nombre = this.editado.nombre;

          this.activo = this.editado.activo;

          this.textSnack = '¡Actualización Exitosa!';

          this.editarMovil(this.editado);

        } else if (this.editado.nombre === '' || this.editado.activo === null) {

          // Guarda el registro en caso de Alta

          this.textSnack = 'Datos incompletos.';

        } else {

          this.nombre = this.editado.nombre;

          this.activo = this.editado.activo;

          this.textSnack = '¡Alta exitosa!';

          this.altaMovil();

        }

        this.cancelar();

      },

    },

};

</script>

8) Generar las rutas para el cliente en client\router.js

import AfipOperaciones from './views/AfipOperaciones.vue';

y luego el mapeo:

{

path: '/marcas',

      name: 'marcas',

      component: Marcas,

},