

INSTALLATION GUIDE

Information	Tools	Version
<i>Text Editor</i>	<i>Visual Studio Code</i>	1.86.2
<i>Framework</i>	- <i>Laravel (Back End)</i> - <i>Vue Js (Front End)</i>	10.45.0 3.2.37
<i>DBMS</i>	<i>MySQL</i>	
<i>Web Server</i>	<i>Apache</i>	
<i>Browser</i>	<i>Microsoft Edge</i>	
<i>Server</i>	<i>localhost</i>	
<i>Additional Software</i>	- <i>Node Js</i> - <i>Composer</i> - <i>Git</i> - <i>Heidi SQL</i>	20.11.0 2.6.6 2.42.0 11.1
<i>Dependencies (NPM Package)</i>	- <i>@kyvg/vue3-notification</i> - <i>primeicons</i> - <i>vue-axios</i> - <i>vue-router</i>	3.2.0 6.0.1 3.5.2 4.2.5

Install Aplikasi Laravel

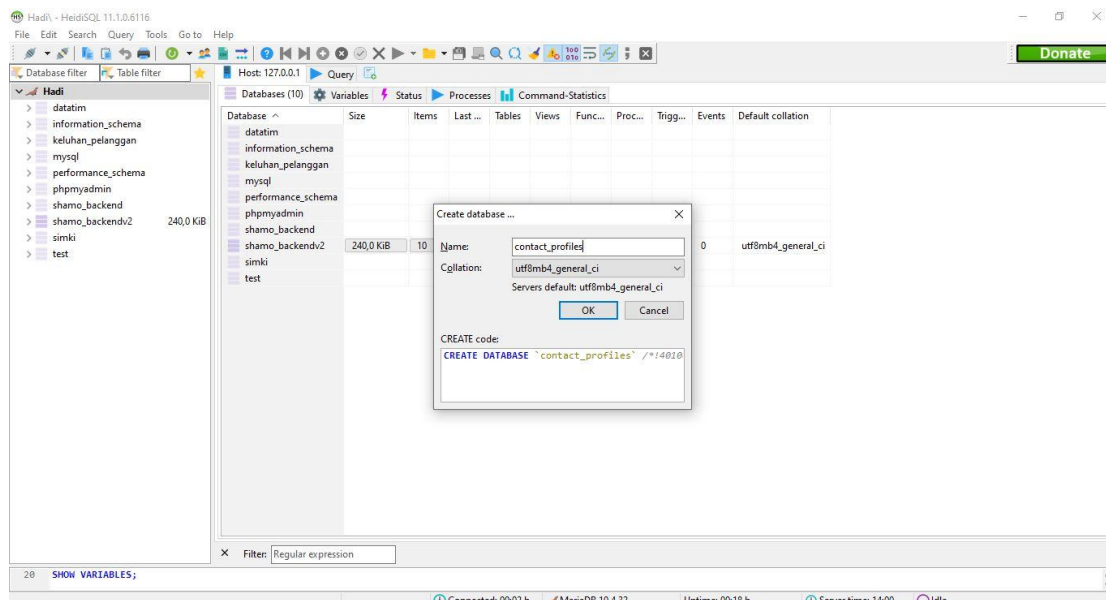
Buka terminal, ketik perintah dibawah ini untuk membuat projek laravel baru



```
composer create-project laravel/laravel contact-profiles
```

Buat Database

Sebelum membuka projek laravel yang telah dibuat, selanjutnya membuat *database* yang digunakan nantinya untuk menampung data-data yang akan disimpan. Disini kebetulan saya memakai tools *HeidiSQL*, bisa juga menggunakan *phpmyadmin*.



Koneksi Database

Proses masuknya data memerlukan koneksi antara sebuah projek aplikasi dan *database*. Buka projek laravel yang telah dibuat sebelumnya, pada file `.env` tambahkan *database name, username, password*.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=contact_profiles
DB_USERNAME=root
DB_PASSWORD=
```

Model dan Migration

Membuat *Model* dan *Migration* pada sebuah aplikasi laravel, bisa ketik dan jalankan perintah dibawah ini

```
php artisan make:model Contact -m
```

Tambahkan *code* pada *database/migrations/create_contacts_table.php*

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('contacts', function (Blueprint $table) {
            $table->id();
            $table->string('name',50);
            $table->string('email_address')->unique();
            $table->string('phone_number',20);
            $table->string('address',100);
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('contacts');
    }
};
```

Model digunakan sebagai representasi dari tabel *database*. Tambahkan *code* pada *app/Models/Contact.php*. Perintah *protected \$fillable* berfungsi sebagai proteksi *atribut* yang boleh melakukan *insert/update* ke *database*.

```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Contact extends Model
{
    use HasFactory;
    protected $fillable = [
        'name',
        'email_address',
        'phone_number',
        'address'
    ];
}
```

Selanjutnya, menjalankan migrasi yang telah dibuat dengan perintah dibawah ini.

```
php artisan migrate
```

Controller

Controller bertanggung jawab menyimpan, memproses algoritma sebuah aplikasi dan mengembalikan hasil yang sesuai. Untuk membuat *controller* buka terminal, jalankan perintah seperti dibawah ini.

```
php artisan make:controller ContactProfilesController
```

Buka `app/Http/Controllers/ContactProfilesController`. Tambahkan `code` untuk proses tambah, edit, hapus dan lihat data.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Contact;

class ContactProfilesController extends Controller
{
    //
}
```

Get Data (All)

Function ini menjelaskan bagaimana cara memanggil data pada *database* dengan *Eloquent ORM* yaitu `Contact::all()` dan mengembalikan hasil dalam bentuk JSON.

Keterangan :

Contact = Model

all() = Ambil Data Keseluruhan

```
/* Get Data (All)
public function index(){
    $contact = Contact::all();
    return response([
        'message' => 'Data
Contact', 'success' => true,
        'data' => $contact,
    ]);
}
```

Get Data By Id (Ambil Data Berdasarkan Id)

Function ini menjalankan perintah untuk memanggil data berdasarkan *id*

```
/* Get Data (By Id)
public function show($id){
    $person = Contact::find($id);
    return response([
        'message' => 'Show Person Successfull',
        'success' => true,
        'data' => $person,
    ]);
}
```

Store (Tambah Data)

Function ini melakukan proses tambah dan menyimpan data pada *database* dengan cara memvalidasi *input request* terlebih dahulu kemudian disimpan menggunakan perintah *Contact::create([])* dan mengembalikan hasil dalam bentuk JSON.

```
/* Store (Add)
public function store(Request $request){
    // Request Validate
    $request->validate([
        'name' => 'required|string',
        'email_address' => 'required|string|email|unique:contacts',
        'phone_number' => 'required',
        'address' => 'required'
    ]);

    $contact = Contact::create([
        'name' => $request->name,
        'email_address' => $request->email_address,
        'phone_number' => $request->phone_number,
        'address' => $request->address,
    ]);

    return response([
        'success' => true,
        'message' => 'Contact Profile Saved Successfully'
    ]);
}
```

Edit (Ubah Data)

Function ini bertugas melakukan perubahan data berdasarkan *id* yang dipanggil

```
/* Edit
public function edit(Request $request, $id){
    // Request Validate
    $request->validate([
        'name' => 'required|string',
        'email_address' => 'required|string|email|unique:contacts,email_address,'.$id,
        'phone_number' => 'required',
        'address' => 'required'
    ]);

    // data request
    $data = [
        'name' => $request->name,
        'email_address' => $request->email_address,
        'phone_number' => $request->phone_number,
        'address' => $request->address,
    ];

    $person = Contact::find($id);
    $person->update($data);

    return response([
        'success' => true,
        'message' => 'Edit Successfull',
        'data' => $data
    ]);
}
```

Delete (Hapus Data)

Function ini bertugas untuk melakukan penghapusan data *contact* berdasarkan *id* yang dipilih untuk dihapus.

```
/* Delete
public function delete($id){
    $person = Contact::find($id)->delete();
    return response([
        'success' => true,
        'message' => 'Delete Person Successfully'
    ]);
}
```

Routes

Dikarenakan aplikasi adanya komunikasi data antara *backend* dan *frontend* yang berbeda. Sehingga perlu adanya penghubung antara *Laravel* dan *Vue*. Buka file *routes/web.php*, tulis perintah seperti dibawah ini.

```
<?php

use Illuminate\Support\Facades\Route;

/*
|-----
| Web Routes
|-----
|
| Here is where you can register web routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "web" middleware group. Make something great!
|
*/

Route::get('/{any}', function () {
    return view('app');
})->where('any', '.*');
```

Kemudian, definisikan *route* pada file *routes/api.php* sebagai fungsi API sehingga *Vue Js* dapat mengakses data pada fungsi-fungsi *Laravel*.

```
<?php


use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;
use App\Http\Controllers\ContactProfilesController;

/*
|-----
| API Routes
|-----
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "api" middleware group. Make something great!
|
*/

Route::get('/', [ContactProfilesController::class, 'index']);
Route::get('/show/{id}', [ContactProfilesController::class, 'show']);
Route::post('/store', [ContactProfilesController::class, 'store']);
Route::post('/edit/{id}', [ContactProfilesController::class, 'edit']);
Route::delete('/delete/{id}', [ContactProfilesController::class, 'delete']);
```



Install Laravel Vue UI

Langkah ini menjelaskan untuk melakukan instalasi tampilan *Vue* pada projek *Laravel*. Buka terminal, ketikkan perintah seperti dibawah ini.




```
composer require laravel/ui
```

Jika telah selesai, lakukan perintah selanjutnya seperti dibawah ini.




```
php artisan ui vue
```

Selanjutnya, diperlukan *dependencies* *vue-router* (interaksi antar halaman) dan *vue-axios* (untuk mengakses API pada *Laravel*).




```
npm install vue-router vue-axios
```

Kemudian, pada aplikasi ini menggunakan *dependencies* tambahan untuk *icons* dan *notification*.



```
npm install primeicons @kyvg/vue3-notification
```

Apabila sudah berhasil dan selesai semua, selanjutnya mencompiler assets pada *Vue*.



```
npm run dev
```

Pengaturan *Vue* Pada *Laravel*

Untuk melakukan pengaturan *vue* pada aplikasi *laravel*, kita harus membuat sebuah *file* yang menjembatani tampilan/layout *vue* dapat bekerja pada aplikasi *laravel*. Tambahkan sebuah file baru pada ***resources/views/app.blade.php***.

```
<!doctype html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta name="csrf-token" value="{{ csrf_token() }}" />
  <title>Vue JS CRUD Operations in Laravel</title>

  <link href="https://fonts.googleapis.com/css2?family=Poppins&display=swap"
rel="stylesheet">
  <!-- Scripts -->
  @vite(['resources/sass/app.scss', 'resources/js/app.js'])
</head>
<body>
  <div id="app"></div>
</body>
</html>
```

Langkah selanjutnya, buka file ***resources/js/app.js*** tambahkan perintah seperti dibawah ini. Langkah ini untuk mendefinisikan *dependencies* yang digunakan, melakukan *register components vue* dan melakukan *rendering* agar *layout vue* bisa dibaca pada aplikasi *laravel*.

```
// DEPENDENCIES
import "./bootstrap";
import { createApp } from "vue";
import App from "./App.vue";
import axios from "axios";
import VueAxios from "vue-axios";

// Notifications
import Notifications from "@kyvg/vue3-notification";

// Primeicons
import "primeicons/primeicons.css";

// Set up router (SPA)
import { createRouter, createWebHistory } from "vue-router";
import { routes } from "./routes";
const router = createRouter({
  history: createWebHistory(),
  routes,
});

// Use dependencies
const app = createApp(App);
app.use(router);
app.use(VueAxios, axios);
app.use(Notifications);

// Rendering app.blade.php
app.mount("#app");
```

Kemudian, tambahkan file **routes.js** pada **resources/js** yang berfungsi sebagai *route* (interaksi antar halaman) pada *vue*.

```
export const routes = [
  {
    path: "/",
    name: "home",
    component: () => import("./Pages/Home.vue"),
  },
  {
    path: "/create",
    name: "create",
    component: () => import("./Pages/Create.vue"),
  },
  {
    path: "/edit/:id",
    name: "edit",
    component: () => import("./Pages/Edit.vue"),
  },
  {
    path: "/show/:id",
    name: "show",
    component: () => import("./Pages/Show.vue"),
  },
];
```

Tambahkan file baru **App.vue** pada **resources/js**. File ini dijalankan pertama kali saat proses *rendering*. Tag *router-view* digunakan sebagai wadah/components yang menampung semua tampilan pada *vue*.

```
<template>
  <div class="wrapper">
    <div class="container">
      <nav class="navbar navbar-light bg-light">
        <router-link :to="{ name: 'home' }" class="navbar-brand mb-0">
          >Navbar</router-link>
        </nav>

        <!-- Notifications -->
        <notifications />

        <!-- ROUTER VIEW -->
        <router-view> </router-view>
      </div>
    </div>
  </template>

  <style>
  body {
    font-family: "Poppins";
  }

  .title-header {
    font-size: 18px;
  }

  .card {
    border-radius: 9px;
    border: 1px solid rgba(0, 0, 0, 0.204);
    box-shadow: 0px 2px 8px 1px rgba(0, 0, 0, 0.1);
  }
  </style>
```

CRUD VUE

Pada langkah ini, menjelaskan membuat tampilan untuk proses CRUD pada *vue* serta melakukan pengambilan data dengan mengakses API yang telah dibuat pada *backend (laravel)*.

Buat folder baru **Pages** pada **resources/js** dan tambahkan file :

- Home.vue
- Create.vue
- Edit.vue
- Show.vue

Tambahkan *code* pada **resources/js/Pages/Home.vue**. File ini nanti kita akan mengakses semua data pada *database* serta melakukan penghapusan salah satu data dari *database*

```
<template>
<div class="container">
  <div class="row">
    <div class="col-12">
      <div class="card">
        <div class="card-body">
          <!-- Header -->
          <section>
            <span class="title-header">Dashboard</span>
          </section>
          <section>
            <!-- Tambah -->
            <router-link
              :to="{ name: 'create' }"
              class="btn btn-sm btn-success my-3"
            >
              Tambah Data
            </router-link>

            <!-- Table -->
            <table class="table table-hover">
              <thead>
                <tr>
                  <th scope="col">Id</th>
                  <th scope="col">Name</th>
                  <th scope="col">Email</th>
                  <th scope="col">Phone Number</th>
                  <th scope="col">Address</th>
                  <th scope="col">Action</th>
                </tr>
              </thead>
              <tbody>
                <tr v-for="(contact, index) in contacts" :key="index">
                  <th scope="row">
                    {{ contact.id }}
                  </th>
                  <th scope="row">
                    {{ contact.name }}
                  </th>
                  <th scope="row">
                    {{ contact.email_address }}
                  </th>
                  <th scope="row">
                    {{ contact.phone_number }}
                  </th>
                  <th scope="row">
                    {{ contact.address }}
                  </th>
                  <th>
                    <div class="btn-group btn-group-toggle">
                      <router-link
                        :to="{
                          name: 'edit',
                          params: { id: contact.id },
                        }"
                        class="btn btn-success btn-sm"
                        data-toggle="tooltip"
                        data-placement="top"
                        title="Update Keluhan"
                      >
                        <i class="pi pi-pencil" style="color: white;"></i>
                      </router-link>
                      <router-link
                        :to="{
                          name: 'show',
                          params: { id: contact.id },
                        }"
                        class="btn btn-primary btn-sm"
                        data-toggle="tooltip"
                        data-placement="top"
                        title="Lihat Keluhan"
                      >
                        <i class="pi pi-eye" style="color: white;"></i>
                      </router-link>
                      <button
                        class="btn btn-danger btn-sm"
                        data-toggle="tooltip"
                        data-placement="top"
                        title="Hapus Keluhan"
                        @click="deleteData(contact.id)"
                      >
                        <i class="pi pi-trash" style="color: white;"></i>
                      </button>
                    </div>
                  </th>
                </tr>
              </tbody>
            </table>
          </section>
        </div>
      </div>
    </div>
  </div>
</div>
</template>
```

```








<script>
export default {
  data() {
    return {
      contacts: [],
    };
  },
  created() {
    this.loadData();
  },
  methods: {
    loadData() {
      this.axios.get(`/api`).then((response) => {
        this.contacts = response.data.data;
        console.log(this.contacts);
      });
    },
    deleteData(id) {
      /** Delete Data
      this.axios
        .delete(`/api/delete/` + id)
        .then((response) => {
          this.loadData();
          this.$notify({ title: response.data.message, type: "error" });
        })
        .catch((errors) => {
          this.$notify({ title: errors.response.data.message, type: "error" });
          console.log(errors.response.status);
        });
    },
  },
};
</script>

```

Navbar

Dashboard

Tambah Data

Id	Name	Email	Phone Number	Addres	Action
1	Trihadi Putra	trihadi17@gmail.com	01212	Jalan	  
2	Tiya Wulandari	tiya@gmail.com	1212	Jalan	  
3	Muhammad Hafiz	hafiz@gmail.com	10921	Jalani	  

Selanjutnya tambahkan *code* pada *resources/js/Pages/Create.vue*. File ini akan melakukan proses penyimpanan data pada *database*.

```
<template>
  <section>
    <div class="row">
      <div class="col-12">
        <div class="card">
          <div class="card-body">
            <span class="title-header">Create</span>
            <!-- Form Tambah -->
            <section class="my-3">
              <div class="row">
                <div class="col-12">
                  <form @submit.prevent="onSubmit">
                    <div class="form-group mb-2">
                      <label for="name">Full Name</label>
                      <input
                        v-model="contact.name"
                        class="form-control"
                        placeholder="Full Name"
                      />
                      <span class="form-text text-danger" v-if="errors.name">{{
                        errors.name[0]
                      }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="email_address">Email Address</label>
                      <input
                        v-model="contact.email_address"
                        class="form-control"
                        placeholder="Email Address"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.email_address"
                        >{{ errors.email_address[0] }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="phone_number">Phone Number</label>
                      <input
                        v-model="contact.phone_number"
                        class="form-control"
                        placeholder="Phone Number"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.phone_number"
                        >{{ errors.phone_number[0] }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="address">Address</label>
                      <input
                        v-model="contact.address"
                        class="form-control"
                        placeholder="Address"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.address"
                        >{{ errors.address[0] }}</span>
                    </div>

                    <div class="form-group mt-2" style="float: right">
                      <router-link
                        :to="{ name: 'home' }"
                        class="btn btn-sm btn-secondary"
                        style="margin-right: 5px"
                        >Cancel</router-link>
                      <button class="btn btn-sm btn-success ml-2" type="submit">
                        Create
                      </button>
                    </div>
                  </form>
                </div>
              </div>
            </section>
          </div>
        </div>
      </div>
    </div>
  </section>
</template>
```

```

<script>
export default {
  data() {
    return {
      contact: {},
      errors: {
        name: "",
        email_address: "",
        phone_number: "",
        address: "",
      },
    };
  },
  methods: {
    onSubmit() {
      this.$axios
        .post(`/api/store`, this.contact)
        .then((response) => {
          this.$router.push({ name: "home" });
          this.$notify({ title: response.data.message, type: "success" });
        })
        .catch((err) => {
          this.errors.name = err.response.data.errors.name;
          this.errors.email_address = err.response.data.errors.email_address;
          this.errors.phone_number = err.response.data.errors.phone_number;
          this.errors.address = err.response.data.errors.address;
        });
    },
  },
};
</script>

```

Navbar

Create

Full Name

Email Address

The email address has already been taken.

Phone Number

The phone number field is required.

Address

The address field is required.

Cancel Create

Langkah selanjutnya, tambahkan *code* pada ***resources/js/Pages/Edit.vue***. File ini akan melakukan proses perubahan data pada *database* berdasarkan *id*.

```
<template>
  <section>
    <div class="row">
      <div class="col-12">
        <div class="card">
          <div class="card-body">
            <span class="title-header">Edit</span>
            <!-- Form Tambah -->
            <section class="my-3">
              <div class="row">
                <div class="col-12">
                  <form @submit.prevent="onSubmit">
                    <div class="form-group mb-2">
                      <label for="name">Full Name</label>
                      <input
                        v-model="contact.name"
                        class="form-control"
                        placeholder="Full Name"
                      />
                      <span class="form-text text-danger" v-if="errors.name">{{
                        errors.name[0]
                      }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="email_address">Email Address</label>
                      <input
                        v-model="contact.email_address"
                        class="form-control"
                        placeholder="Email Address"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.email_address"
                        >{{ errors.email_address[0] }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="phone_number">Phone Number</label>
                      <input
                        v-model="contact.phone_number"
                        class="form-control"
                        placeholder="Phone Number"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.phone_number"
                        >{{ errors.phone_number[0] }}</span>
                    </div>

                    <div class="form-group mb-2">
                      <label for="address">Address</label>
                      <input
                        v-model="contact.address"
                        class="form-control"
                        placeholder="Address"
                      />
                      <span
                        class="form-text text-danger"
                        v-if="errors.address"
                        >{{ errors.address[0] }}</span>
                    </div>

                    <div class="form-group mt-2" style="float: right">
                      <router-link
                        :to="{ name: 'home' }"
                        class="btn btn-sm btn-secondary"
                        style="margin-right: 5px"
                        >Cancel</router-link>
                      <button class="btn btn-sm btn-success ml-2" type="submit">
                        Edit
                      </button>
                    </div>
                  </form>
                </div>
              </div>
            </section>
          </div>
        </div>
      </div>
    </div>
  </section>
</template>
```



```

<script>
export default {
  data() {
    return {
      contact: {},
      errors: {
        name: "",
        email_address: "",
        phone_number: "",
        address: "",
      },
    };
  },
  created() {
    this.loadData();
  },
  methods: {
    loadData() {
      /* Get Data */
      this.$axios.get(`/api/show/${this.$route.params.id}`).then((response) => {
        this.contact = response.data.data;
      });
    },
    onSubmit() {
      this.$axios
        .post(`/api/edit/${this.$route.params.id}`, this.contact)
        .then((response) => {
          this.$router.push({ name: "home" });
          this.$notify({ title: response.data.message, type: "success" });
        })
        .catch((err) => {
          this.errors.name = err.response.data.errors.name;
          this.errors.email_address = err.response.data.errors.email_address;
          this.errors.phone_number = err.response.data.errors.phone_number;
          this.errors.address = err.response.data.errors.address;
        });
    },
  },
};
</script>

```

Navbar

Edit

Full Name

Email Address

Phone Number

Address

Lalu, tambahkan *code* pada ***resources/js/Pages/Show.vue***. File ini akan melakukan proses menampilkan data pada *database* berdasarkan *id*.

```
<template>
  <section>
    <div class="row">
      <div class="col-12">
        <div class="card">
          <div class="card-body">
            <span class="title-header">Show</span>

            <section class="my-3">
              <div class="form-group mb-2">
                <label for="name" class="col-12">Full Name</label>
                <span class="btn btn-sm btn-primary">{{ contact.name }}</span>
              </div>

              <div class="form-group mb-2">
                <label for="email_address" class="col-12">Email Address</label>
                <span class="btn btn-sm btn-primary">{{
                  contact.email_address
                }}</span>
              </div>

              <div class="form-group mb-2">
                <label for="phone_number" class="col-12">Phone Number</label>
                <span class="btn btn-sm btn-primary">{{
                  contact.phone_number
                }}</span>
              </div>

              <div class="form-group mb-2">
                <label for="address" class="col-12">Address</label>
                <span class="btn btn-sm btn-primary">{{
                  contact.address
                }}</span>
              </div>
            </section>
          </div>
        </div>
      </div>
    </div>
  </section>
</template>
```

```
<script>
export default {
  data() {
    return {
      contact: {},
    };
  },
  created() {
    this.loadData();
  },
  methods: {
    loadData() {
      /* Get Data */
      this.axios.get(`/api/show/${this.$route.params.id}`).then((response) => {
        this.contact = response.data.data;
      });
    },
  },
};
</script>
```

Navbar

Show

Full Name

Trihadi Putra

Email Address

trihadi17@gmail.com

Phone Number

01212

Address

Jalan