**Implementing CRUD operations using JSON server**

**Aim :** To implement CRUD operation for contact details (contacts object in JSON server)

**Step – 1**

* Create a Vue-cli project with router concept
* Include the following options;’
  + (babel, router,vuex,linter/formatter)

Commands

* npm install –g @vue/cli or npm install –g vue-cli
* vue create <contact-manager>

**Step – 2**

* Install bootstrap, font-awesome, axios (Search in google – fontawesome npm)

Command

* npm i bootstrap @fortawesome/fontawesome-free axios

Check the version of fontawesome installed in our application – package.json

* Select and copy the CDN library ( installed version 6.1.1) and keep it in
* **public > index.html**

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css" integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />

* Configure bootstrap and fontawesome **import them in main.js file**

/\* Font Awesome Icons \*/

import "../node\_modules/@fortawesome/fontawesome-free/css/all.css";

/\* Bootstrap CSS and JS \*/

import "../node\_modules/bootstrap/dist/css/bootstrap.css";

import "../node\_modules/bootstrap/dist/js/bootstrap.bundle";

* **Create global style sheet**
  + Create **style.css in src----- > src > style.css**
* @import url('https://fonts.googleapis.com/css2?family=Roboto:wght@300&display=swap');
* body{
* font-family: 'Roboto', sans-serif !important;
* background-color: #f0fdf0 !important;
* }
* .contact-img{
* width: 150px;
* border-radius:50%;
* }
* Import the style.css in **main.js** file
  + import "./style.css";

**Step – 3**

* Create the following **vue files in src > views folder**
* ContactManager.vue
* AddContact.vue
* ViewContact.vue
* EditContact
* PageNotFound
* Create and manage routes in router file
  + src > router > index.js

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

import ContactManagerVue from '@/views/ContactManager.vue'

import AddContactVue from '@/views/AddContact.vue'

import EditContactVue from '@/views/EditContact.vue'

import ViewContactVue from '@/views/ViewContact.vue'

import PageNotFoundVue from '@/views/PageNotFound.vue'

const routes = [

{

path: '/',

name: 'home',

redirect : "/contacts",

component: HomeView

},

{

path: '/contacts',

name: 'ContactManager',

component: ContactManagerVue

},

{

path: '/contacts/add',

name: 'AddContact',

component: AddContactVue

},

{

path: '/contacts/edit/:contactId',

name: 'EditContact',

component: EditContactVue

},

{

path: '/contacts/view/:contactId',

name: 'ViewContact',

component: ViewContactVue

},

{

// path: '/\*',

path: '/:pathMatch(.\*)\*',

name: 'PageNotFound',

component: PageNotFoundVue

}

]

* **Set the routes in App.vue**
* <router-view /> in template
* Check the routes in the browser

**Step – 4**

* **Create the following components files under > src > components folder**.
* NavBar.vue (Keep navbar )
* Spinner.vue (To specify loading time delay)

**NavBar.vue**

<template>

<nav class="navbar navbar-dark bg-success navbar-expand-sm">

<div class="container">

<router-link to="/" class="navbar-brand">

<i class="fa fa-mobile"></i>

Contact Manager

</router-link>

</div>

</nav>

</template>

<script>

export default{

name: 'NavBar'

}

</script> <style scoped> </style>

**Spinner.vue**

* Search and the spinner (loading) image in > src > assets file

<template>

<div class="container">

<div class="row">

<div class="col">

<img src="../assets/Spinner-loading-image.gif" />

</div>

</div>

</div>

</template>

<script>

export default{

name: 'SpinnerVue'

}

</script>

<style scoped>

</style>

**Step – 5**

**Creation installation and creation of JSON server and its data**

* Create a folder ‘server’ in the application
* Go to the server folder in the terminal
* Type the command
  + ….. /server > **npm init –yes** (Creates default package for json)
  + …../server **> npm install json-server** (installs json-server)
* **Create db.json file under server folder** . It contains “contacts” and “groups” object details

{

"contacts": [

{

"id": "aaaaa",

"name": "Arockia Seelan",

"company": "Infosis",

"email": "seelan@gmail.com",

"title": "Project Lead",

"mobile": "9894205895",

"photo": "https://th.bing.com/th/id/R.6b0022312d41080436c52da571d5c697?rik=ejx13G9ZroRrcg&riu=http%3a%2f%2fpluspng.com%2fimg-png%2fuser-png-icon-young-user-icon-2400.png&ehk=NNF6zZUBr0n5i%2fx0Bh3AMRDRDrzslPXB0ANabkkPyv0%3d&risl=&pid=ImgRaw&r=0",

"groupId": "2"

},

{

"id": "bbbbbb",

"name": "Michael Churchill",

"company": "Wipro",

"email": "churchill@gmail.com",

"title": "System Engineer",

"mobile": "9894205895",

"photo": "https://i.pinimg.com/736x/5f/40/6a/5f406ab25e8942cbe0da6485afd26b71.jpg",

"groupId": "3"

},

{

"id": "cccccccc",

"name": "Jeyam Christin",

"company": "TCS",

"email": "jeyam@gmail.com",

"title": "HR Manager",

"mobile": "9909895895",

"photo": "https://cdn4.iconfinder.com/data/icons/people-avatar-2-1/128/60-512.png",

"groupId": "4"

},

{

"id": "dddddddd",

"name": "Joseph Daniel",

"company": "CGVAK-Covai",

"email": "daniel@gmail.com",

"title": "DB admin",

"mobile": "9894209090",

"photo": "https://th.bing.com/th/id/OIP.08QlgS6XlzNXeGDcCqqUVQHaHa?w=192&h=192&c=7&r=0&o=5&pid=1.7",

"groupId": "2"

},

{

"name": "Helena",

"company": "TCS",

"email": "helena@gmail.com",

"title": "Engineer",

"mobile": "9893328756",

"photo": "https://www.vippng.com/png/detail/88-881947\_free-icons-png-user-female-icon-png.png",

"groupId": "1",

"id": "UfiBJU9"

}

],

"groups": [

{

"id": "1",

"name": "Colleague"

},

{

"id": "2",

"name": "Family"

},

{

"id": "3",

"name": "Friends"

},

{

"id": "4",

"name": "Service"

}

]

}

* **Go to > server > package.json and include the following script**

"scripts": {

"start": "json-server --watch db.json -p 9000"

},

* To run json server
* …../server> npm start

(JSON server is started in the localhost 9000)

**http://localhost:9000/**

**Step – 6**

**List of CRUD services in the application**

* To view all contacts 🡪 Use GET
  + <http://localhost:9000/contacts>
* To view single contacts 🡪 Use GET
  + <http://localhost:9000/contacts>/**:**contactId
* To create a contact 🡪 Use POST
* <http://localhost:9000/contacts>
* To update a contact 🡪 Use PUT
* <http://localhost:9000/contacts>/**:**contactId
* To delete a contact 🡪 Use DELETE
* <http://localhost:9000/contacts>/:contactId

**Creation of services ---- (CRUD operations using Axios)**

* **Create a folder name services under src**
  + > src > services
* Create a java script file ContactServices.js under > src > services folder

(Use axios for the CRUD operations )

**Code in ContactServices.js**

import axios from "axios";

export class ContactService{

static serverURL = `http://localhost:9000`;

static getAllContacts(){

let dataURL = `${this.serverURL}/contacts`;

return axios.get(dataURL);

}

static getContact(contactId){

let dataURL = `${this.serverURL}/contacts/${contactId}`

return axios.get(dataURL);

}

static createContact(contact){

let dataURL = `${this.serverURL}/contacts/`;

return axios.post(dataURL, contact);

}

static updateContact(contact, contactId){

let dataURL= `${this.serverURL}/contacts/${contactId}`;

return axios.put(dataURL,contact);

}

static deleteContact(contactId){

let dataURL= `${this.serverURL}/contacts/${contactId}`;

return axios.delete(dataURL);

}

static getAllGroups(){

let dataURL = `${this.serverURL}/groups/`;

return axios.get(dataURL);

}

static getGroup(contact){

let groupId = contact.groupId;

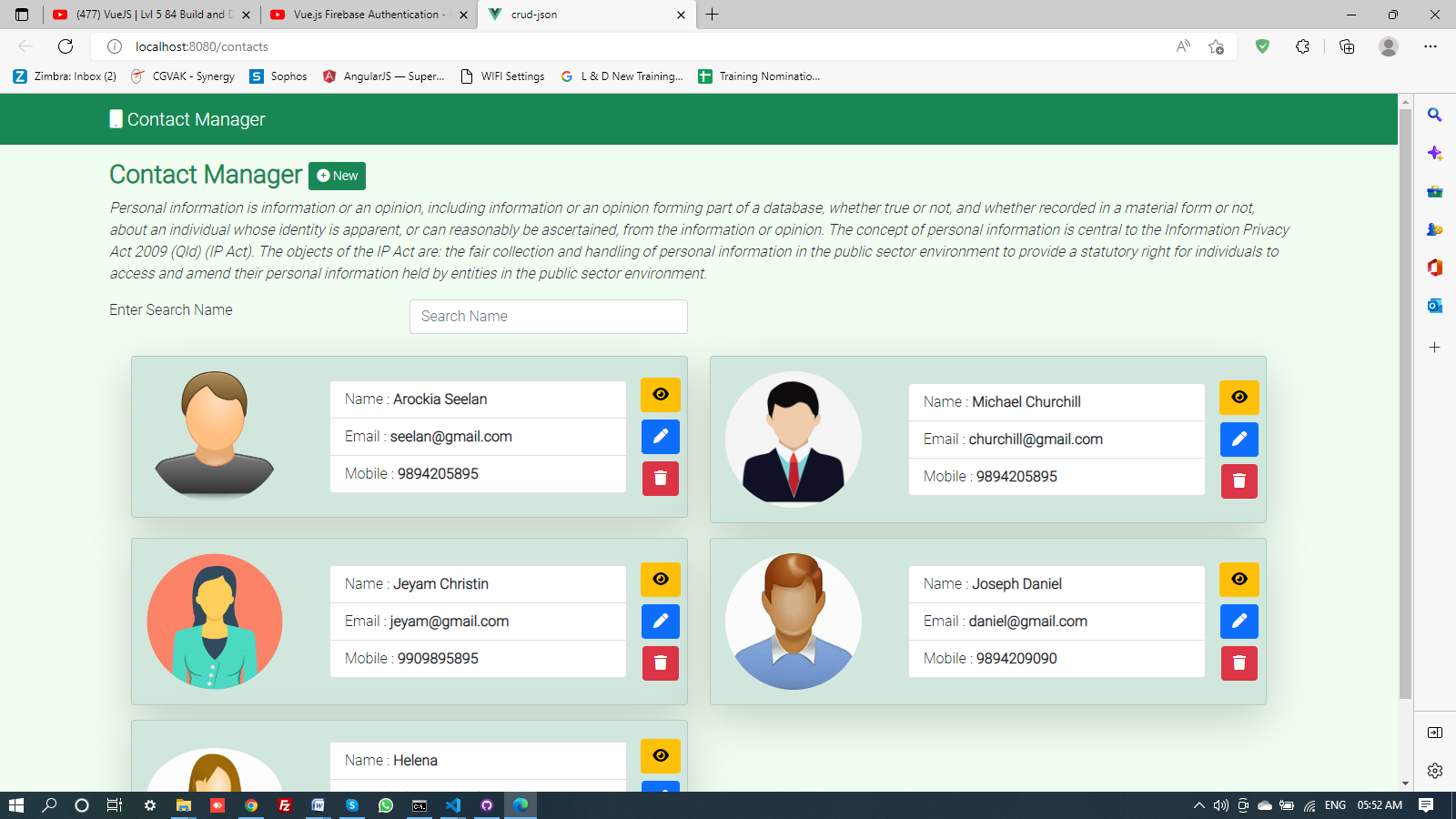
let dataURL = `${this.serverURL}/groups/${groupId}`;

return axios.get(dataURL);

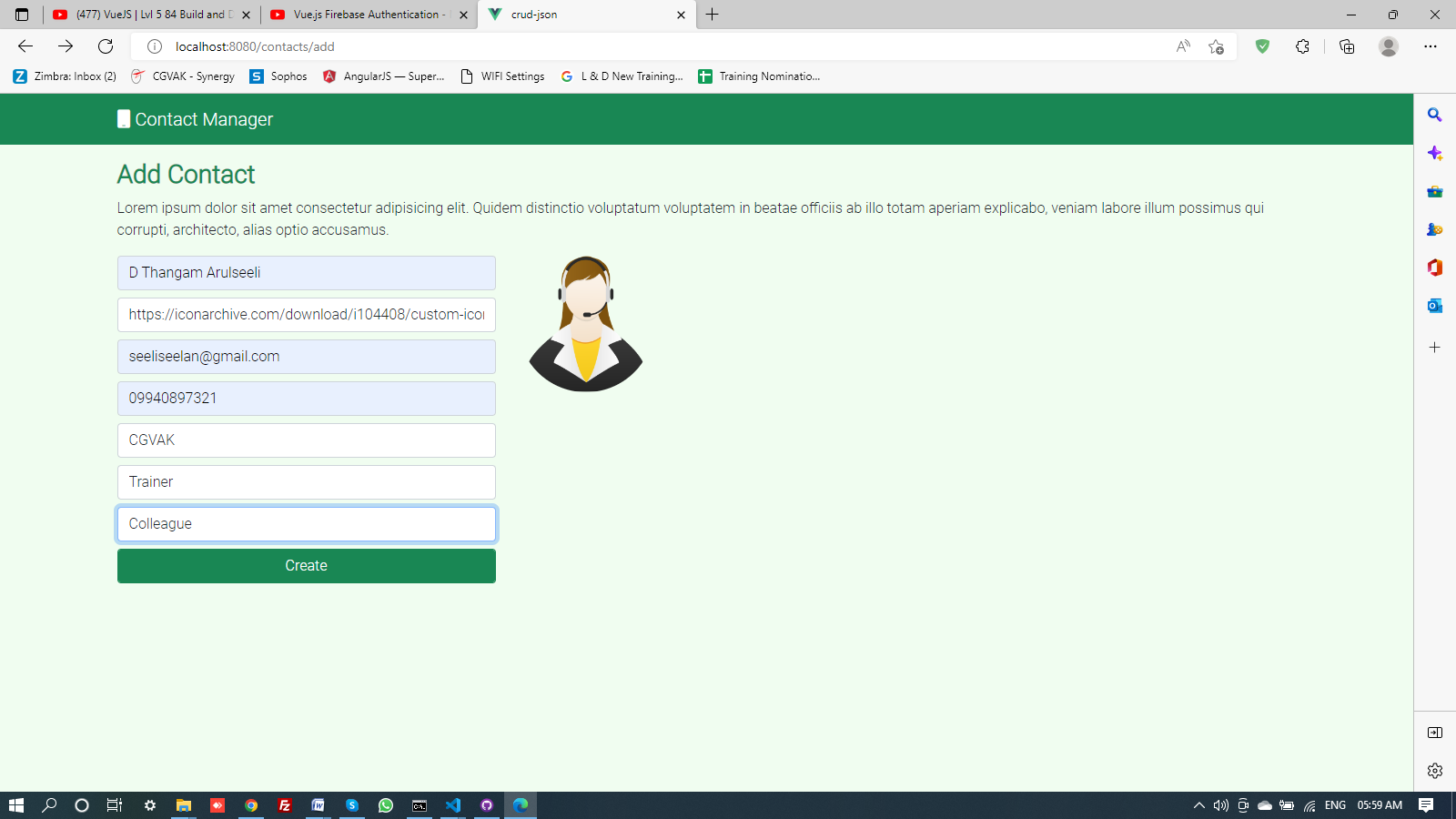
} }

**Layouts**

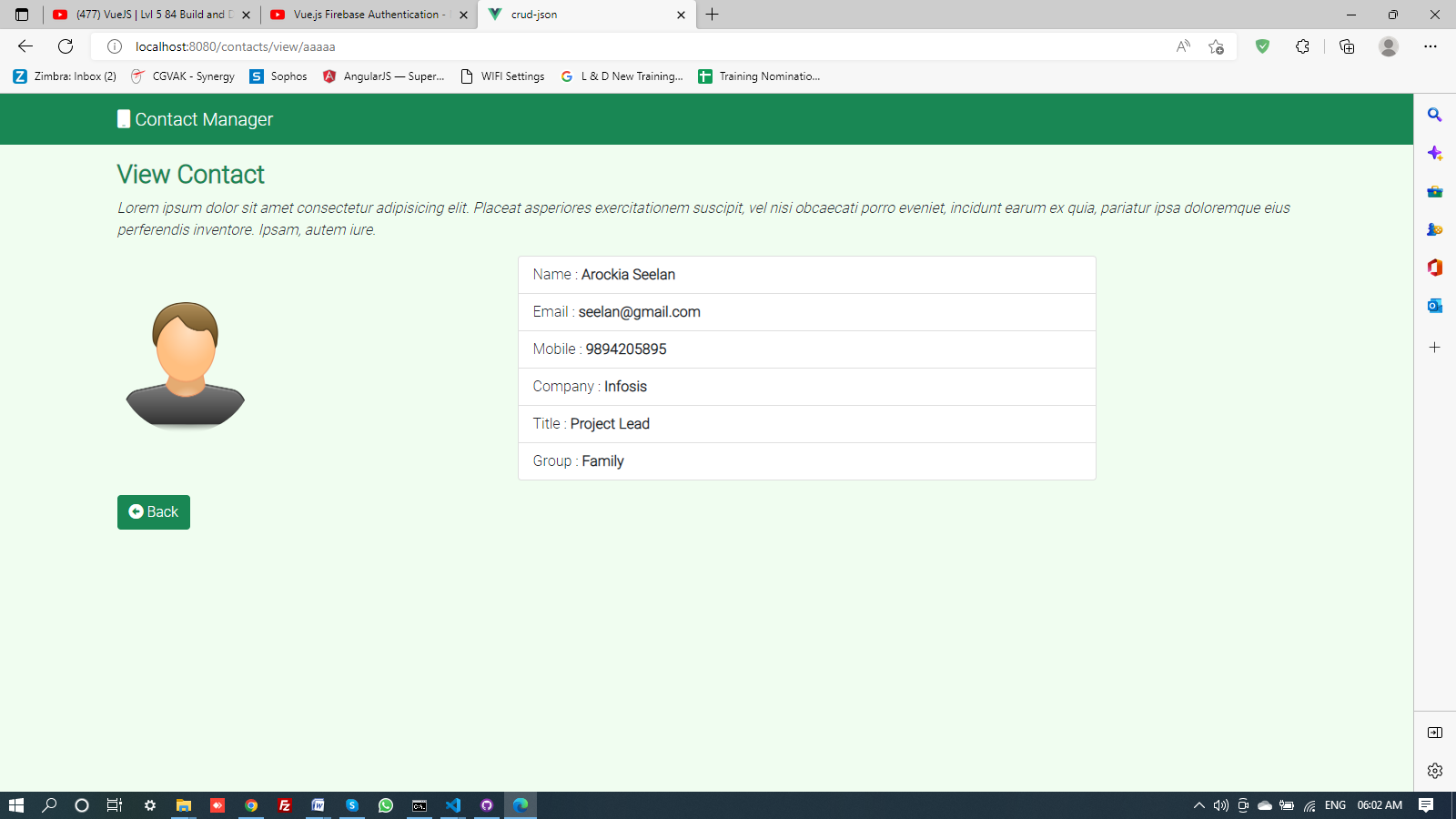
**ContactManager** (Initial Page) ---- ContactManager.vue



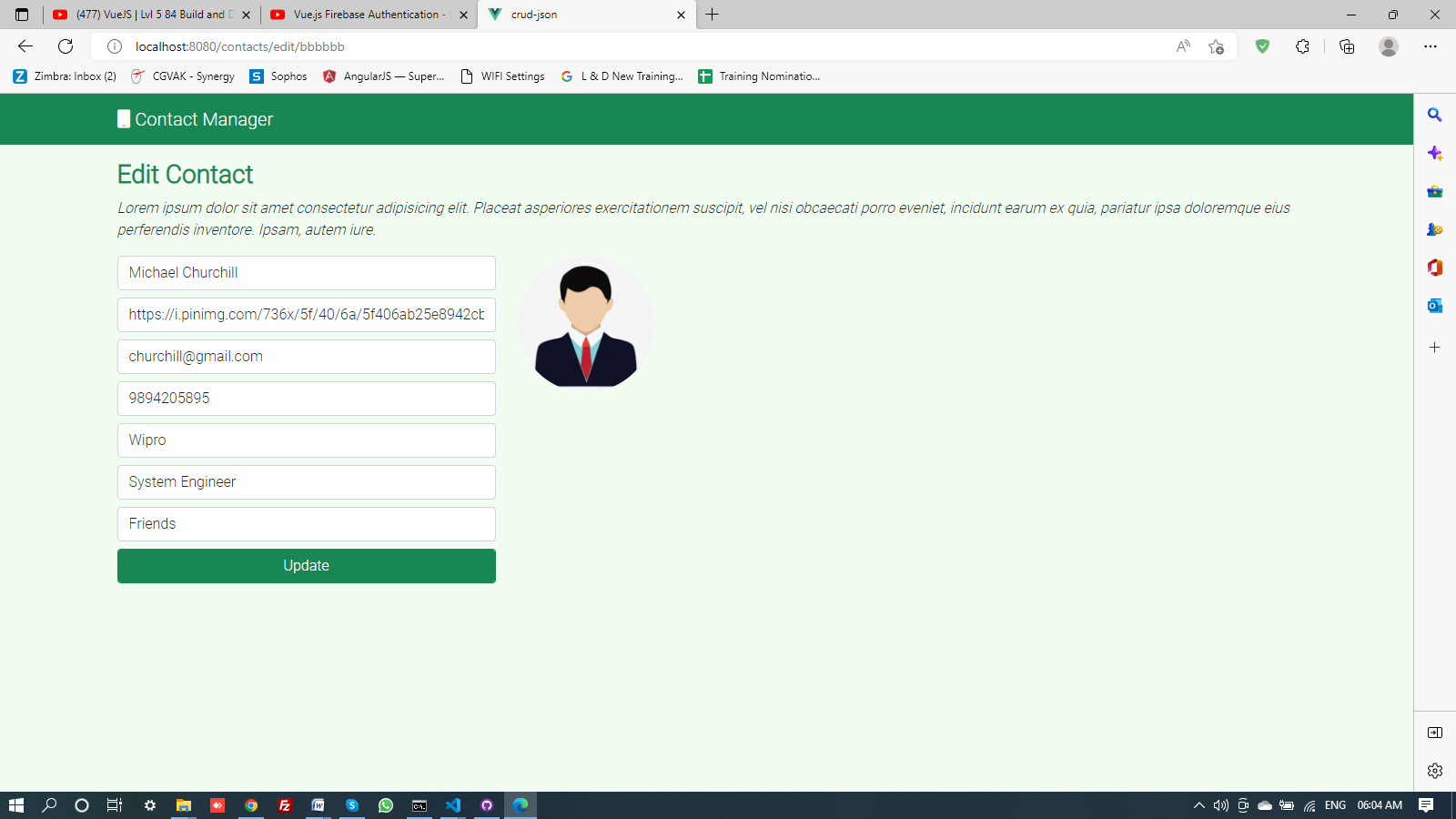
**Add New Contact Details** ----- AddContact.vue



**View Contact Details** ----- ViewContact.vue



**Edit Contact Details** ----- ViewContact.vue



**Code in ContactManager.vue**

<template>

<div class="container mt-3">

<div class="row">

<div class="col">

<p class="h3 text-success fw-bold">

Contact Manager

<router-link to="/contacts/add" class="btn btn-success btn-sm">

<i class="fa fa-plus-circle"> </i> New

</router-link>

</p>

<p class="fst-italic"> Personal information is information or an opinion, including information or an opinion forming part of a database, whether true or not, and whether recorded in a material form or not, about an individual whose identity is apparent, or can reasonably be ascertained, from the information or opinion.

The concept of personal information is central to the Information Privacy Act 2009 (Qld) (IP Act). The objects of the IP Act are:

the fair collection and handling of personal information in the public sector environment

to provide a statutory right for individuals to access and amend their personal information held by entities in the public sector environment.

</p>

<form>

<div class="row">

<div class="col-md-6">

<div class="row">

<div class="col">

Enter Search Name

</div>

<div class="col">

<input type="text" v-model="searchName" class="form-control" placeholder="Search Name" />

<!-- <input type="submit" class="btn btn-outline-dark" /> -->

</div>

</div>

</div>

</div>

</form>

<!-- Spinner -->

<div v-if="loading">

<div class="container">

<div class="row">

<div class="col">

<Spinner />

</div>

</div>

</div>

</div>

<!-- Error Message -->

<div v-if="!loading && errorMessage">

<div class="container mt-5">

<div class="row">

<div class="col">

<p class="h3 text-danger fw-bold"> {{errorMessage}} </p>

</div>

</div>

</div>

</div>

<div class="container mt-3" v-if="contacts.length > 0">

<div class="row">

<div class="col-md-6" v-for="contact of filterContacts" :key="contact">

<div class="card my-2 list-group-item-success shadow-lg">

<div class="card-body">

<div class="row align-items-center">

<div class="col-sm-4">

<img :src="contact.photo"

alt="contact.name" class="contact-img" />

</div>

<div class="col-sm-7">

<ul class="list-group">

<li class="list-group-item">

Name : <span class="fw-bold"> {{contact.name}}</span>

</li>

<li class="list-group-item">

Email : <span class="fw-bold"> {{contact.email}}</span>

</li>

<li class="list-group-item">

Mobile : <span class="fw-bold"> {{contact.mobile}}</span>

</li>

</ul>

</div>

<div class="col-sm-1 d-flex flex-column justify-content-center align-items-center">

<!-- <router-link to="/contacts/view/:contactId" -->

<router-link :to="`/contacts/view/${contact.id}`"

class="btn btn-warning my-1">

<i class="fa fa-eye"></i>

</router-link>

<!-- <router-link to="/contacts/edit/:contactId" -->

<router-link :to="`/contacts/edit/${contact.id}`"

class="btn btn-primary my-1">

<i class="fa fa-pen"></i>

</router-link>

<button class="btn btn-danger my-1" @click="clickDeleteContact(contact.id)">

<i class="fa fa-trash"> </i>

</button>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</template>

<script>

import { ContactService } from '@/services/ContactServices';

import Spinner from '@/components/Spinner.vue';

export default{

name: "ContactManager",

data: function () {

return {

loading: false,

contacts: [],

errorMessage: null,

searchName: ''

};

},

created: async function () {

try {

this.loading = true;

// let response=await this.getAllContactsData();

let response = await ContactService.getAllContacts();

this.contacts = response.data;

this.loading = false;

}

catch (error) {

this.errorMessage = error;

this.loading = false;

}

},

computed: {

filterContacts(){

return this.contacts.filter(contact => contact.name.toLowerCase().includes(this.searchName.toLowerCase()));

}

},

methods: {

// getAllContactsData : async function(){

// return await ContactService.getAllContacts();

// }

clickDeleteContact: async function(contactId){

try{

this.loading=true;

let response = await ContactService.deleteContact(contactId);

if (response){

let response = await ContactService.getAllContacts(); // gets the fresh data after deletion

this.contacts = response.data;

this.loading = false;

}

}

catch (error){

this.errorMessage=error;

this.loading=false;

}

}

},

components: { Spinner }

}

</script>

<style scoped>

</style>

**Code in AddContact.vue**

<template>

<div class="container mt-3">

<div class="row">

<div class="col">

<p class="h3 text-success fw-bold">

Add Contact

</p>

<p classs="fst-italic"> Lorem ipsum dolor sit amet consectetur adipisicing elit. Quidem

distinctio voluptatum voluptatem in beatae officiis ab illo totam aperiam explicabo,

veniam labore illum possimus qui corrupti, architecto,

alias optio accusamus.</p>

</div>

</div>

</div>

<!-- {{contact}} -->

<div class="container" mt-3>

<div class="row">

<div class="col-md-4">

<form @submit.prevent="submitCreate()">

<div class="mb-2">

<input type="text" required v-model="contact.name" class="form-control" placeholder="Name" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.photo" class="form-control" placeholder="Photo URL" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.email" class="form-control" placeholder="Email" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.mobile" class="form-control" placeholder="Mobile" />

</div>

<div class="mb-2">

<input type="text" v-model="contact.company" class="form-control" placeholder="Company" />

</div>

<div class="mb-2">

<input type="text" required v-model='contact.title' class="form-control" placeholder="Title" />

</div>

<div class="mb-2">

<select required v-model="contact.groupId" class="form-control" v-if="groups.length > 0" >

<option value="" > Select Group </option>

<option :value="group.id" v-for="group of groups" :key="group.id"> {{group.name}}</option>

</select>

</div>

<div class="mb-2">

<input type="submit" class="form-control btn btn-success" value="Create" />

</div>

</form>

</div>

<div class="col-md-4">

<img :src="contact.photo" alt="" class="contact-img" />

</div>

</div>

</div>

<!-- <pre> {{contact}}</pre> -->

</template>

<script>

import { ContactService } from '@/services/ContactServices'

export default{

name: "AddContact",

data: function(){

return{

contact: {

name: '',

company: '',

email: '',

title: '',

mobile: '',

photo: '',

groupId: ''

},

groups: []

}

},

created: async function(){

try{

let response=await ContactService.getAllGroups();

this.groups=response.data;

}

catch (error){

console.log(error);

}

},

methods:{

submitCreate: async function(){

try{

let response = await ContactService.createContact(this.contact);

if (response){

return this.$router.push('/');

}

else{

return this.$router.push(`/contacts/add`);

}

}

catch(error){

console.log(error);

}

}

}

}

</script>

<style scoped>

</style>

**Code in ViewContact.vue**

<template>

<div class="container mt-3">

<div class="row">

<div class="col">

<p class="h3 text-success fw-bold"> View Contact </p>

<p class="fst-italic"> Lorem ipsum dolor sit amet consectetur

adipisicing elit. Placeat asperiores exercitationem

suscipit, vel nisi obcaecati porro eveniet,

incidunt earum ex quia, pariatur ipsa doloremque

eius perferendis inventore. Ipsam, autem iure.

</p>

</div>

</div>

</div>

<!-- Spinner -->

<div v-if="loading">

<div class="container">

<div class="row">

<div class="col">

<Spinner />

</div>

</div>

</div>

</div>

<!-- Error Message -->

<div v-if="!loading && errorMessage">

<div class="container mt-5">

<div class="row">

<div class="col">

<p class="h3 text-danger fw-bold"> {{errorMessage}} </p>

</div>

</div>

</div>

</div>

<div class="container" v-if="!loading && isDone()">

<div class="row align-items-center">

<div class="col-md-4">

<img :src="contact.photo"

class="contact-img" />

</div>

<div class="col-md-6">

<ul class="list-group">

<li class="list-group-item">

Name : <span class="fw-bold"> {{contact.name}}</span>

</li>

<li class="list-group-item">

Email : <span class="fw-bold">{{contact.email}}</span>

</li>

<li class="list-group-item">

Mobile : <span class="fw-bold"> {{contact.mobile}}</span>

</li>

<li class="list-group-item">

Company : <span class="fw-bold"> {{contact.company}}</span>

</li>

<li class="list-group-item">

Title : <span class="fw-bold"> {{contact.title}}</span>

</li>

<li class="list-group-item">

Group : <span class="fw-bold"> {{group.name}}</span>

</li>

</ul>

</div>

</div>

<div class="row mt-3">

<div class="col">

<router-link to="/" class="btn btn-success">

<i class="fa fa-arrow-alt-circle-left"></i> Back

</router-link>

</div>

</div>

</div>

<!-- {{contact}}

<br />

{{group}} -->

</template>

<script>

import { ContactService } from '@/services/ContactServices'

import Spinner from '@/components/Spinner.vue';

//import { objectExpression } from '@babel/types';

export default{

name: "ViewContact",

components: {Spinner},

data: function () {

return {

contactId: this.$route.params.contactId,

loading: false,

contact: {},

errorMessage: null,

group: {}

};

},

created: async function () {

try {

this.loading = true;

let response = await ContactService.getContact(this.contactId);

let groupResponse = await ContactService.getGroup(response.data);

this.contact = response.data;

this.group = groupResponse.data;

this.loading = false;

}

catch (error) {

this.errorMessage = error;

this.loading = false;

}

},

methods: {

isDone: function () {

return Object.keys(this.contact).length > 0 && Object.keys(this.group).length > 0;

}

}

}

</script>

<style scoped>

</style>

**Code in EditContact.vue**

<template>

<div class="container mt-3">

<div class="row">

<div class="col">

<p class="h3 text-success fw-bold"> Edit Contact </p>

<p class="fst-italic"> Lorem ipsum dolor sit amet consectetur

adipisicing elit. Placeat asperiores exercitationem

suscipit, vel nisi obcaecati porro eveniet,

incidunt earum ex quia, pariatur ipsa doloremque

eius perferendis inventore. Ipsam, autem iure.

</p>

</div>

</div>

</div>

<!-- Spinner -->

<div v-if="loading">

<div class="container">

<div class="row">

<div class="col">

<Spinner />

</div>

</div>

</div>

</div>

<!-- Error Message -->

<div v-if="!loading && errorMessage">

<div class="container mt-5">

<div class="row">

<div class="col">

<p class="h3 text-danger fw-bold"> {{errorMessage}} </p>

</div>

</div>

</div>

</div>

<div class="container" mt-3>

<div class="row">

<div class="col-md-4">

<form @submit.prevent="updateSubmit()">

<div class="mb-2">

<input type="text" required v-model="contact.name" class="form-control" placeholder="Name" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.photo" class="form-control" placeholder="Photo URL" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.email" class="form-control" placeholder="Email" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.mobile" class="form-control" placeholder="Mobile" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.company" class="form-control" placeholder="Company" />

</div>

<div class="mb-2">

<input type="text" required v-model="contact.title" class="form-control" placeholder="Title" />

</div>

<div class="mb-2">

<select required v-model="contact.groupId" class="form-control" v-if="groups.length > 0">

<option value="" > Select Group </option>

<option :value="group.id" v-for="group of groups" :key="group.id">{{group.name}}</option>

</select>

</div>

<div class="mb-2">

<input type="submit" class="form-control btn btn-success" value="Update" />

</div>

</form>

</div>

<div class="col-md-4">

<img :src="contact.photo"

:alt="contact.name" class="contact-img" />

</div>

</div>

</div>

</template>

<script>

import Spinner from '@/components/Spinner.vue';

import { ContactService } from '@/services/ContactServices';

export default{

name: "EditContact",

components: { Spinner },

data: function(){

return{

contactId: this.$route.params.contactId,

loading: false,

contact: {},

errorMessage: null,

groups: []

}

},

created: async function(){

try{

this.loading = true;

let response = await ContactService.getContact(this.contactId);

let groupResponse = await ContactService.getAllGroups();

this.contact = response.data;

this.groups = groupResponse.data;

this.loading = false;

}

catch(error){

this.errorMessage=error;

this.loading=false;

}

},

methods: {

updateSubmit: async function(){

try{

let response = await ContactService.updateContact(this.contact, this.contactId);

if (response){

return this.$router.push('/');

}

else{

return this.$router.push(`/contacts/edit/${this.contactId}`);

}

}

catch(error){

console.log(error);

}

}

}

}

</script>

**Code in PageNotFound.vue**

<template>

<div class="container text-center mt-3">

<div class="row">

<div class="col">

<h2 class="text-danger"> 404! Page Not Found </h2>

</div>

</div>

</div>

</template>

<script>

export default{

name: "PageNotFound"

}

</script>