# router/index.js

import { createRouter, createWebHistory } from 'vue-router'

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

import AboutView from '../views/AboutView.vue'

import LoginView from '@/views/LoginView.vue'

import AccessDenied from '@/views/AccessDenied.vue'

const routes = [

  {

    path: '/',

    name: 'Home',

    component: HomeView

  },

  {

    path: '/about',

    name: 'About',

    component: AboutView,

    meta: { requiresAuth: true }

  },

  {

    path: '/login',

    name: 'Login',

    component: LoginView

  },

  {

    path: '/access-denied',

    name: 'AccessDenied',

    component: AccessDenied

  }

]

const router = createRouter({

  history: createWebHistory(),

  routes

})

router.beforeEach((to, from, next) => {

  const isAuthenticated = localStorage.getItem('isAuthenticated');

  if (to.meta?.requiresAuth && !isAuthenticated) {

    next('/access-denied');

  } else {

    next();

  }

});

export default router

# src/App.vue

<script setup>

// import LoginForm from './components/LoginForm.vue'

// import Json from './components/JSON.vue'

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

import BHeader from './components/BHeader.vue'

</script>

<template>

  <div class="main-container">

    <header>

      <BHeader />

    </header>

    <main class="main-box">

      <RouterView></RouterView>

    </main>

  </div>

  <!-- <Json /> -->

  <!-- <LoginForm /> -->

  <!-- <HomeView /> -->

</template>

# src/views/HomeView.vue

<template>

  <div class="container mt-5">

    <div class="row">

      <div class="col-md-8 offset-md-2">

        <h1 class="text-center">User Information Form</h1>

        <form @submit.prevent="submitForm">

          <div class="row mb-3">

            <!-- username -->

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

              <label for="username" class="form-label">Username</label>

              <input

                type="text"

                class="form-control"

                id="username"

                v-model="formData.username"

                @blur="() => validateName(true)"

                @input="() => validateName(false)"

              />

              <div v-if="errors.username" class="text-danger">{{ errors.username }}</div>

            </div>

            <!-- gender -->

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

              <label for="gender" class="form-label">Gender</label>

              <select

                id="gender"

                class="form-select"

                v-model="formData.gender"

                @change="validateGender"

              >

                <option value="male">Male</option>

                <option value="female">Female</option>

                <option value="other">Other</option>

              </select>

              <div v-if="errors.gender" class="text-danger">{{ errors.gender }}</div>

            </div>

          </div>

          <div class="row mb-3">

            <!-- password -->

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

              <label for="password" class="form-label">Password</label>

              <input

                type="password"

                class="form-control"

                id="password"

                v-model="formData.password"

                @blur="

                  () => {

                    validatePassword(true)

                  }

                "

                @input="

                  () => {

                    validatePassword(false)

                  }

                "

              />

              <div v-if="errors.password" class="text-danger">{{ errors.password }}</div>

            </div>

            <!-- confirm password -->

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

              <label for="confirm-password" class="form-label">Confirm Password</label>

              <input

                type="password"

                class="form-control"

                id="confirm-password"

                v-model="formData.confirmPassword"

                @blur="

                  () => {

                    validateConfirmPassword(true)

                  }

                "

              />

              <div v-if="errors.confirmPassword" class="text-danger">

                {{ errors.confirmPassword }}

              </div>

            </div>

          </div>

          <!-- isAustralian -->

          <div class="mb-3">

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

              <div class="form-check">

                <input

                  type="radio"

                  class="form-check-input"

                  id="isAustralian"

                  name="isAustralian"

                  v-model="formData.isAustralian"

                  value="Yes"

                  @change="validateResident"

                />

                <label class="form-check-label" for="isAustralian"> Australian Resident</label>

              </div>

              <div class="form-check">

                <input

                  type="radio"

                  class="form-check-input"

                  id="notAustralian"

                  name="isAustralian"

                  v-model="formData.isAustralian"

                  value="No"

                  @change="validateResident"

                />

                <label class="form-check-label" for="notAustralian">Not Australian Resident</label>

              </div>

              <div v-if="errors.resident" class="text-danger">{{ errors.resident }}</div>

            </div>

          </div>

          <!-- reason -->

          <div class="mb-3">

            <label for="reason" class="form-label">Reason for joining</label>

            <textarea

              class="form-control"

              id="reason"

              rows="3"

              v-model="formData.reason"

              @change="validateReason"

            ></textarea>

            <div v-if="errors.reason" :class="errClass">{{ errors.reason }}</div>

          </div>

          <!-- Suburb -->

          <div class="mb-3">

            <label for="reason" class="form-label">Suburb</label>

            <input type="text" class="form-control" id="suburb" :value="formData.suburb" />

          </div>

          <!-- submit or celar button -->

          <div class="text-center">

            <button type="submit" class="btn btn-primary me-2">Submit</button>

            <button type="button" class="btn btn-secondary" @click="clearForm">Clear</button>

          </div>

        </form>

      </div>

    </div>

    <!-- <div class="row mt-5" v-if="submittedCards.length">

        <div class="d-flex flex-wrap justify-content-start">

          <div

            v-for="(card, index) in submittedCards"

            :key="index"

            class="card m-2"

            style="width: 18rem"

          >

            <div class="card-header">User Information</div>

            <ul class="list-group list-group-flush">

              <li class="list-group-item">Username:{{ card.username }}</li>

              <li class="list-group-item">Password:{{ card.password }}</li>

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

                Australian Resident:{{ card.isAustralian ? 'Yes' : 'No' }}

              </li>

              <li class="list-group-item">Gender:{{ card.gender }}</li>

              <li class="list-group-item">Reason:{{ card.reason }}</li>

            </ul>

          </div>

        </div>

      </div> -->

    <div class="card">

      <DataTable :value="submittedCards" tableStyle="min-width: 50rem">

        <Column field="username" header="Username"></Column>

        <Column field="password" header="Password"></Column>

        <Column field="isAustralian" header="Australian Resident"></Column>

        <Column field="gender" header="Gender"></Column>

        <Column field="reason" header="Reason"></Column>

      </DataTable>

    </div>

  </div>

</template>

<script setup>

import { ref } from 'vue'

import DataTable from 'primevue/datatable'

import Column from 'primevue/column'

const formData = ref({

  username: '',

  password: '',

  confirmPassword: '',

  isAustralian: '',

  reason: '',

  gender: '',

  suburb: 'Clayton'

})

const submittedCards = ref([])

const submitForm = () => {

  validateName(true)

  validatePassword(true)

  validateResident()

  validateGender()

  validateReason()

  validateConfirmPassword(true)

  if (

    !errors.value.username &&

    !errors.value.password &&

    !errors.value.resident &&

    !errors.value.gender &&

    !errors.value.reason &&

    !errors.value.confirmPassword

  ) {

    submittedCards.value.push({ ...formData.value })

    clearForm()

  }

}

const clearForm = () => {

  formData.value = {

    username: '',

    password: '',

    isAustralian: false,

    reason: '',

    gender: ''

  }

}

const errors = ref({

  username: null,

  password: null,

  confirmPassword: null,

  resident: null,

  gender: null,

  reason: null

})

const validateName = (blur) => {

  if (formData.value.username.length < 3) {

    if (blur) errors.value.username = 'Name must be at least 3 characters'

  } else {

    errors.value.username = null

  }

}

const validatePassword = (blur) => {

  const password = formData.value.password

  const minLength = 8

  const hasUppercase = /[A-Z]/.test(password)

  const hasLowercase = /[a-z]/.test(password)

  const hasNumber = /\d/.test(password)

  const hasSpecialChar = /[!@#$%^&\*(),.?":{}|<>]/.test(password)

  if (password.length < minLength) {

    if (blur) {

      errors.value.password = `Password must be at least ${minLength} characters long.`

    }

  } else if (!hasUppercase) {

    if (blur) {

      errors.value.password = 'Password must contain at least one uppercase letter.'

    }

  } else if (!hasLowercase) {

    if (blur) {

      errors.value.password = 'Password must contain at least one lowercase letter.'

    }

  } else if (!hasNumber) {

    if (blur) {

      errors.value.password = 'Password must contain at least one number.'

    }

  } else if (!hasSpecialChar) {

    errors.value.password = 'Password must contain at least one special character.'

  } else {

    errors.value.password = null

  }

}

const validateConfirmPassword = (blur) => {

  if (!formData.value.confirmPassword) {

    errors.value.confirmPassword = 'Please confirm the password'

  } else if (formData.value.confirmPassword !== formData.value.password) {

    if (blur) {

      errors.value.confirmPassword = 'Password is not match.'

    }

  } else {

    errors.value.confirmPassword = null

  }

}

const validateResident = () => {

  // console.log(formData.value.isAustralian, !formData.value.isAustralian)

  if (!formData.value.isAustralian) {

    errors.value.resident = 'Please select one of the option'

  } else {

    errors.value.resident = null

  }

}

const validateGender = () => {

  if (!formData.value.gender) {

    errors.value.gender = 'Please select one of the option.'

  } else {

    errors.value.gender = null

  }

}

const errClass = ref('text-danger')

const validateReason = () => {

  if (!formData.value.reason) {

    errors.value.reason = 'Please input the reason.'

    errClass.value = 'text-danger'

  } else if (formData.value.reason.includes('friend')) {

    errors.value.reason = 'Great to have a friend'

    errClass.value = 'text-success'

  } else {

    errors.value.reason = null

  }

}

</script>

<style scoped>

.card {

  border: 1px solid #ccc;

  border-radius: 10px;

  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

}

.card-header {

  background-color: #275fda;

  color: white;

  padding: 10px;

  border-radius: 10px 10px 0 0;

}

.list-group-item {

  padding: 10px;

}

</style>

# Screenshot of App in Localhost (browser) open on Home page

A screenshot of a computer

Description automatically generated

# Screenshot of App in Localhost (browser) open on About page

A screen shot of a computer

Description automatically generated

A screenshot of a login form

Description automatically generated

A screenshot of a computer

Description automatically generated