**ระบบจัดการร้านขายดอกไม้**

**Part 1**

**ส่วนของ Font End**

1. ส่วนของการเข้าระบบของผู้ใช้งาน (Login User)

2. ส่วนของการลงทะเบียนใช้งาน (Register User)

3. ส่วนของการออกจากระบบของผู้ใช้งาน (Logout User)

4. การทำ Routing

5. การสร้าง API

6. ส่วนของข้อมูลร้านขายดอกไม้

7. ส่วนของเพิ่มข้อมูลร้านขายดอกไม้

8. ส่วนของปรับปรุงข้อมูลร้านขายดอกไม้

9. ส่วนของลบข้อมูลร้านขายดอกไม้

10. การทำ Sorting

11. ส่วนอื่น ๆ ที่เกี่ยวข้อง

**ส่วนของ Back End**

1. JSON SERVER API

2. POSTMAN TEST API

3. APIs

3.1 List

3.2 Add

3.3 Update

3.4 Delete

3.5 Authentic User

**ส่วนของ Technology ที่ใช้**

1. Node.js

2. NPM

3. AXIOS

4. Vue.js 3

5. JSON

**Part 2**

การติดตั้งโปรแกรม/ซอฟต์แวร์/เครื่องมือในการพัฒนา/ชิ้นงาน/โครงงาน

1. ติดตั้ง Node.js และ NPM (node 14.x.x & npm 6.x.x)

nvm install 14

nvm use 14

2. ติดตั้ง Vue.js (Install vue cli)

npm install –g @vue/cli

3. สร้างชิ้นงาน/โครงงาน (Create Project)

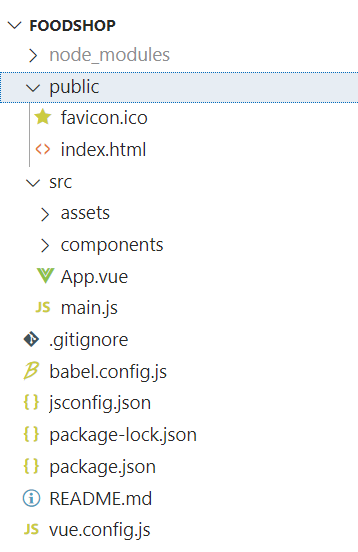
vue create foodshop

4. สั่งงานชิ้นงาน/โครงงาน (Run Project)

npm run serve

**Part 3**

1. ทำความเข้าใจเกี่ยวกับโครงสร้างไฟล์และโฟลเดอร์ชิ้นงาน/โครงงาน



2. ทำความเข้าใจเกี่ยวกับโค้ดของภายในไฟล์

|  |
| --- |
| **App.vue** |
| <template>  <img alt="Vue logo" src="./assets/logo.png">  <HelloWorld msg="Welcome to Your Vue.js App"/>  </template>  <script>  import HelloWorld from './components/HelloWorld.vue'  export default {  name: 'App',  components: {  HelloWorld  }  }  </script>  <style>  #app {  font-family: Avenir, Helvetica, Arial, sans-serif;  -webkit-font-smoothing: antialiased;  -moz-osx-font-smoothing: grayscale;  text-align: center;  color: #2c3e50;  margin-top: 60px;  }  </style> |

3. ทำการลบไฟล์ที่ไม่เกี่ยวข้องโครงงาน/ชิ้นงาน

HelloWorld.vue

**Part 4**

1. สร้างคอมโพเนนต์สำหรับการลงทะเบียนใช้งานในระบบ (SignUp.vue) ในคอมโพเนนต์ App (App.vue)

|  |
| --- |
| **SignUp.vue** |
| <template>  <img alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <h1>ลงทะเบียนใช้งาน</h1>  </template>  <script>  export default{  name: 'SignUp'  }  </script>  <style>  </style> |

2. เพิ่มคอมโพเนนต์ลงทะเบียนใช้งาน (SignUp.vue)

|  |
| --- |
| **App.vue** |
| <template>  <SignUp />  </template>  <script>  import SignUp from './components/SignUp.vue';  export default {  name: 'App',  components: {  SignUp  }  }  </script>  <style>  #app {  font-family: Avenir, Helvetica, Arial, sans-serif;  -webkit-font-smoothing: antialiased;  -moz-osx-font-smoothing: grayscale;  text-align: center;  color: #2c3e50;  margin-top: 60px;  }  </style> |

3. เพิ่มฟิลด์ในฟอร์มแบบต่าง ๆ

|  |
| --- |
| **SignUp.vue** |
| <template>  <img alt="FlowerShop Logo" src="../assets/ FlowerShop.png">  <div>  <input type="text" placeholder="ชื่อ-สกุล">  <input type="email" placeholder="อีเมล์">  <input type="password" placeholder="รหัสผ่าน">  <button>ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  export default{  name: 'SignUp'  }  </script>  <style>  </style> |

4. เพิ่มสไตล์ชีตในคอมโพเนนต์ SignUp

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล">  <input type="email" placeholder="อีเมล์">  <input type="password" placeholder="รหัสผ่าน">  <button>ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  export default{  name: 'SignUp'  }  </script>  <style>  .logo{  width: 100px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

**Part 5**

1. การติดตั้ง JSON Server

1.1 npm install -g json-server

1.2 สร้างโฟลเดอร์ dbjson เพื่อกำหนดเป็น JSON SERVER

2. สร้างไฟล์สำหรับ JSON Serve

2.1 สร้างไฟล์ db.json ในโฟลเดอร์ dbjson

{

"posts": [

{ "id": 1, "title": "json-server", "author": "typicode" }

],

"comments": [

{ "id": 1, "body": "some comment", "postId": 1 }

],

"profile": { "name": "typicode" }

}

2.2 ทำการสั่ง JSON SERVER ทำงาน

json-server --watch db.json

2.3 เปิดดูข้อมูลของ JSON SERVER

localhost:3000

3. สร้าง Dummy API

แก้ไขไฟล์ db.json

{

"user": [

{

"user": "teera",

"email": "teera@test.com",

"password": "teera@test",

},

]

}

4. ทดสอบ API ด้วยโปรแกรม Postman

4.1 เปิดโปรแกรม Postman ทดสอบส่งข้อมูลลงใน db.json

4.1.1 แก้ไขไฟล์ db.json ลบข้อมูลภายใน

{

user:[

]

}

4.1.2 สร้าง Environment โดยกำหนดเป็น localhost:3000/user

4.1.3 เลือก Method เป็น POST

4.1.4 เลือก Body เป็น raw

4.1.5 เขียนข้อมูลในรูปแบบ json

{

"user": "teera",

"email": "teera@test.com",

"password": "teera@test",

}

4.1.6 กดปุ่ม Send

4.1.7 ตรวจสอบข้อมูลในไฟล์ db.json

{

"user": [

{

"user": "teera",

"email": "teera@test.com",

"password": "teera@test",

id": 1

},

{

"user": "peera",

"email": "peera@test.com",

"password": "peera@test",

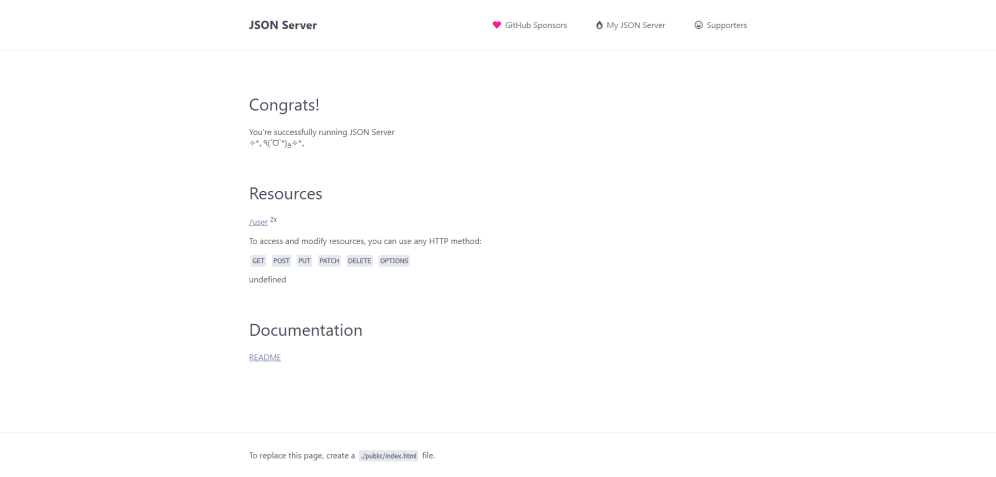
"id": 2

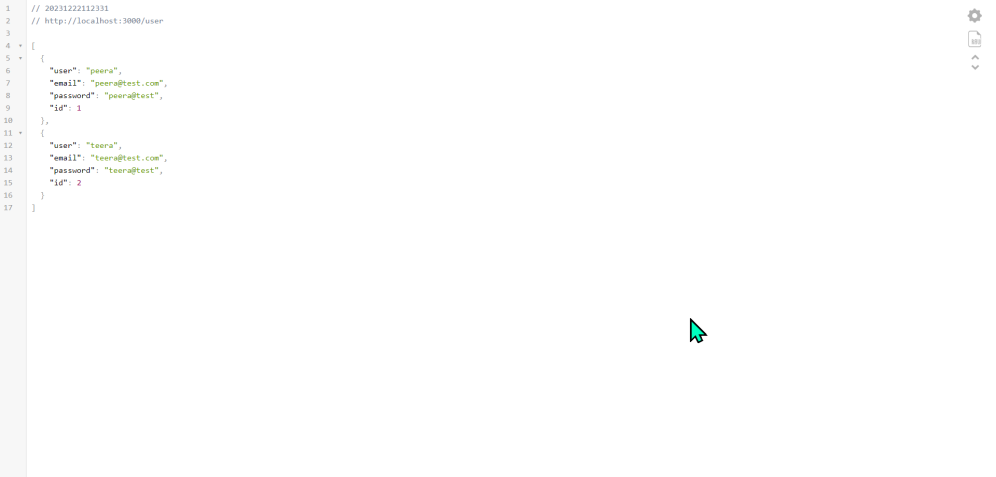
}

]

}

4.1.8 ตรวจสอบข้อมูลผ่าน localhost:3000 เลือก Resources ในส่วนของ user





**Part 6**

1. สร้าง API สำหรับคอมโพเนนต์ SignUp User

1.1 แก้ไขไฟล์ db.json

{

"users": [

]

}

1.2 เปิดโปรแกรม Postman ทดสอบส่งข้อมูลลงใน db.json

1.2.1 สร้าง Environment โดยกำหนดเป็น localhost:3000/user

1.2.2 เลือก Method เป็น POST

1.2.3 เลือก Body เป็น raw

1.2.4 เขียนข้อมูลในรูปแบบ json

{

"user": "teera",

"email": "teera@test.com",

"password": "teera@test",

}

1.2.5 กดปุ่ม Send

1.2.6 ตรวจสอบข้อมูลในไฟล์ db.json

{

"user": [

{

"user": "teera",

"email": "teera@test.com",

"password": "teera@test",

id": 1

},

]

}

2. ทำการเรียก/ดึง (Get) ข้อมูลจากฟิลด์ที่สร้างใน SignUp.vue

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="name">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button>ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  export default{  name: 'SignUp',  data(){  return{  name: '',  email: '',  password: ''  }  }  }  </script>  <style>  .logo{  width: 100px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

3. เรียกฟังก์ชั่นจากการคลิกปุ่ม (Button Click)

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="name">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button v-on:click="signup">ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  export default{  name: 'SignUp',  data(){  return{  name: '',  email: '',  password: ''  }  },  methods:{  signup(){  console.warn("ลงทะเบียนใช้งาน", this.fullname, this.email, this.password)  }  }  }  </script>  <style>  .logo{  width: 100px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

4. ติดตั้งแพ็คเกจ Axios สำหรับ API

npm install axios

5. ทำการเรียก API

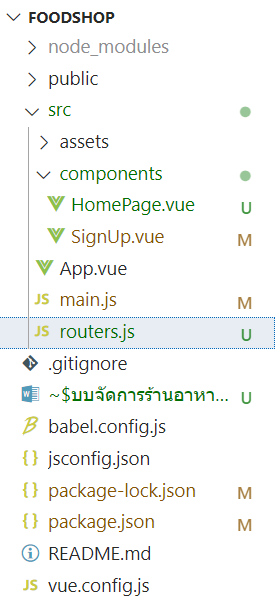
|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="name">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button v-on:click="signup">ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  import axios from 'axios'  export default{  name: 'SignUp',  data(){  return{  name: '',  email: '',  password: ''  }  },  methods:{  async signup(){  let result = await axios.post("http://localhost:3000/users",{  fullname: this.fullname,  email: this.email,  password: this.password  });  console.warn(result);  if(result.status == 201){  alert("ลงทะเบียนใช้งานเรียบร้อยแล้ว");  }else{  alert("ไม่สามารถลงทะเบียนใช้งานได้");  }  }  }  }  </script>  <style>  .logo{  width: 100px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

6. เก็บข้อมูลที่ป้อนลงใน LocalStorage

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/FlowerShop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="name">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button v-on:click="signup">ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  import axios from 'axios'  export default{  name: 'SignUp',  data(){  return{  name: '',  email: '',  password: ''  }  },  methods:{  async signup(){  let result = await axios.post("http://localhost:3000/users",{  fullname: this.fullname,  email: this.email,  password: this.password  });  console.warn(result);  if(result.status == 201){  alert("ลงทะเบียนใช้งานเรียบร้อยแล้ว");  localStorage.setItem("user-data", JSON.stringify(result.data));  }else{  alert("ไม่สามารถลงทะเบียนใช้งานได้");  }  }  }  }  </script>  <style>  .logo{  width: 100px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

**Part 7**

1. สร้างคอมโพเนนต์ Home



|  |
| --- |
| **HomePage.vue** |
| <template>  <h1>  สวัสดี ลูกค้าผู้ใช้งาน ยินดีต้อนรับในระบบของเรา  </h1>  </template>  <script>  export default{  name: 'Home'  }  </script>  <style>  </style> |

2. ติดตั้งแพ็คเกจ vue routing

npm install vue-router@next

3. สร้างไฟล์ Routing (routers.js) สำหรับ Home และ SignUp ในโฟลเดอร์ src

แก้ไขไฟล์ main.js

|  |
| --- |
| **main.js** |
| import { createApp } from 'vue'  import App from './App.vue'  import router from './routers'  createApp(App).use(router).mount('#app') |

4. ย้ายคอมโพเนนต์ออกจาก App.vue

|  |
| --- |
| **App.vue** |
| <template>  <router-view />  </template>  <script>  export default {  name: 'App',  }  </script>  <style>  #app {  font-family: Avenir, Helvetica, Arial, sans-serif;  -webkit-font-smoothing: antialiased;  -moz-osx-font-smoothing: grayscale;  text-align: center;  color: #2c3e50;  margin-top: 60px;  }  </style> |

5. กำหนด route ไปยังหน้า Home และ SignUp

|  |
| --- |
| **routers.js** |
| import Home from './components/Home.vue'  import SignUp from './components/SignUp.vue'  import { createRouter, createWebHistory } from 'vue-router'  const router = createRouter({  history: createWebHistory(),  routes: [  { path: '/', component: Home, name: 'Home' },  { path: '/sign-up', component: SignUp, name: 'SignUp' }  ],  });  export default router; |

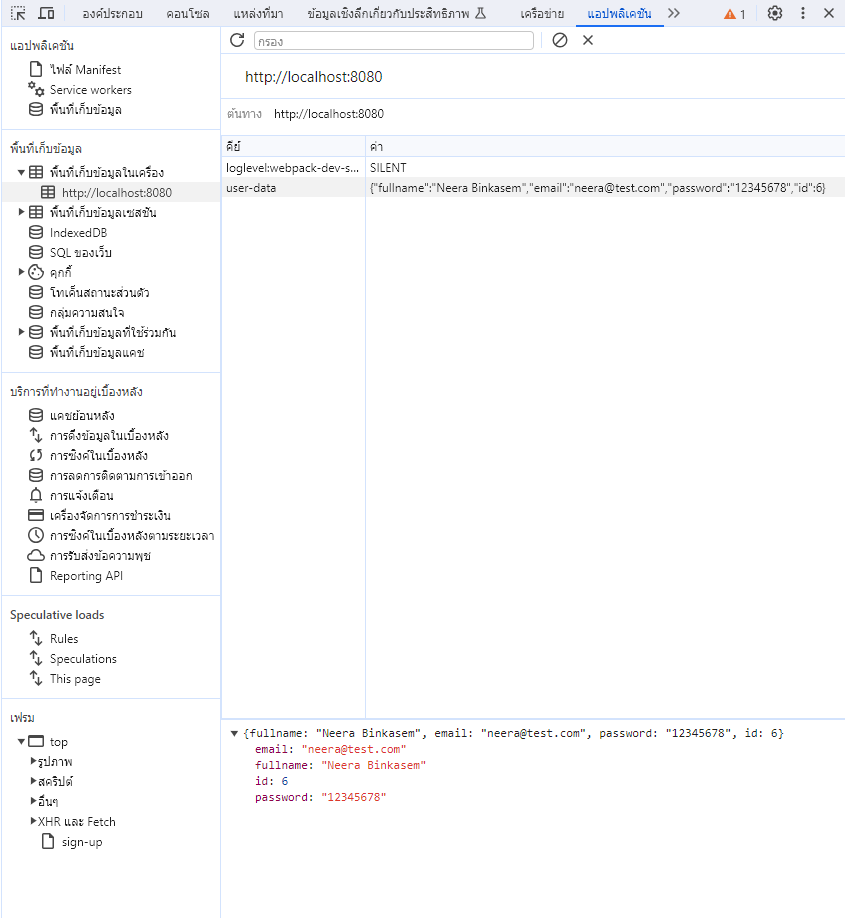
6. กำหนดการทำงานของส่งข้อมูลลงทะเบียนเก็บใน JSON-SERVER หลังจากนั้นส่งไปหน้าที่ต้องการ

(Redirect Page) คือ หน้าแรก (Home.vue)

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/flowershop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="fullname">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button v-on:click="signup">ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  import axios from 'axios'  export default{  name: 'SignUp',  data(){  return{  fullname: '',  email: '',  password: ''  }  },  methods:{  async signup(){  let result = await axios.post("http://localhost:3000/users",{  fullname: this.fullname,  email: this.email,  password: this.password  });  if(result.status == 201){  localStorage.setItem("user-data", JSON.stringify(result.data))  this.$router.push({  name: 'Home'  })  }else{  alert("ไม่สามารถลงทะเบียนใช้งานได้");  }  }  }  }  </script>  <style>  .logo{  width: 200px;  padding-bottom: 20px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

**Part 8**

1. ตรวจสอบข้อมูลลการลงทะเบียนใช้งาน และหน้าแรก



2. เก็บข้อมูลของการลงทะเบียน แล้วตรวจสอบ หากมีข้อมูลแล้วให้ส่งไปยังหน้าที่ต้องการ

3. ทดสอบการทำงานของระบบ

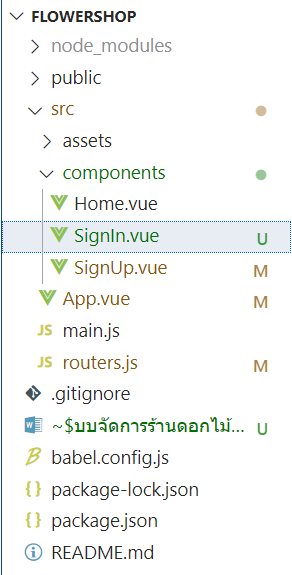
4. สร้างฟังก์ชั่น mouted ในไฟล์ Home.vue และ SignUp.vue

|  |
| --- |
| **Home.vue** |
| <template>  <h1>สวัสดีลูกค้า</h1>  <h2>ยินดีต้อนรับในการใช้งานระบบของเรา</h2>  </template>  <script>  export default{  name: 'Home',  mounted(){  let user = localStorage.getItem("user-data")  if(!user){  this.$router.push({  name: 'SignUp'  })  }  }  }  </script>  <style>  </style> |

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/flowershop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="fullname">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button v-on:click="signup">ลงทะเบียนใช้งาน</button>  </div>  </template>  <script>  import axios from 'axios'  export default{  name: 'SignUp',  data(){  return{  fullname: '',  email: '',  password: ''  }  },  methods:{  async signup(){  let result = await axios.post("http://localhost:3000/users",{  fullname: this.fullname,  email: this.email,  password: this.password  });  if(result.status == 201){  // alert("ลงทะเบียนใช้งานเรียบร้อยแล้ว");  localStorage.setItem("user-data", JSON.stringify(result.data))  this.$router.push({  name: 'Home'  })  }else{  alert("ไม่สามารถลงทะเบียนใช้งานได้");  }  },  },  mounted(){  let user = localStorage.getItem("user-data")  if(user){  this.$router.push({  name: 'Home'  })  }  }  }  </script>  <style>  .logo{  width: 200px;  padding-bottom: 20px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

**Part 9**

1. สร้างคอมโพเนนต์สำหรับหน้าในการเข้าระบบ (SignIn)



2. กำหนดเส้นทาง(Route) ไปยังหน้าในการเข้าระบบ (SignIn)

|  |
| --- |
| **routers.js** |
| import Home from './components/Home.vue'  import SignUp from './components/SignUp.vue'  import SignIn from './components/SignIn.vue'  import { createRouter, createWebHistory } from 'vue-router'  const router = createRouter({  history: createWebHistory(),  routes: [  { path: '/', component: Home, name: 'Home' },  { path: '/sign-up', component: SignUp, name: 'SignUp' },  { path: '/sign-in', component: SignIn, name: 'SignIn' }  ],  });  export default router; |

|  |
| --- |
| **SignIn.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/flowershop.png">  <div class="signin">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button>เข้าใช้งานในระบบ</button>  <p><router-link to="/sign-up">ลงทะเบียนใช้งาน</router-link></p>  </div>  </template>  <script>  export default{  name: 'SignIn',  data(){  return{  email: '',  password: ''  }  },  }  </script>  <style>  </style> |

3. สร้างจุดเชื่อมโยง (Link) ในหน้าของการลงทะเบียน(SignUp) ไปยังหน้าในการเข้าระบบ(SignIn)

|  |
| --- |
| **SignUp.vue** |
| <template>  <img class="logo" alt="FlowerShop Logo" src="../assets/flowershop.png">  <div class="signup">  <input type="text" placeholder="ชื่อ-สกุล" v-model="fullname">  <input type="email" placeholder="อีเมล์" v-model="email">  <input type="password" placeholder="รหัสผ่าน" v-model="password">  <button>ลงทะเบียนใช้งาน</button>  <p>  <router-link to="/sign-in">ใช้งานในระบบ</router-link>  </p>  </div>  </template>  <script>  import axios from 'axios'  export default{  name: 'SignUp',  data(){  return{  fullname: '',  email: '',  password: ''  }  },  methods:{  async signup(){  let result = await axios.post("http://localhost:3000/users",{  fullname: this.fullname,  email: this.email,  password: this.password  });  if(result.status == 201){  // alert("ลงทะเบียนใช้งานเรียบร้อยแล้ว");  localStorage.setItem("user-data", JSON.stringify(result.data))  this.$router.push({  name: 'Home'  })  }else{  alert("ไม่สามารถลงทะเบียนใช้งานได้");  }  },  },  mounted(){  let user = localStorage.getItem("user-data")  if(user){  this.$router.push({  name: 'Home'  })  }  }  }  </script>  <style>  .logo{  width: 200px;  padding-bottom: 20px;  }  .signup input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

4. แก้ไขและเพิ่มรูปแบบ (Style) ของหน้าลงทะเบียนใช้งานและหน้าในการเข้าระบบ นำไปวางใน App.vue

|  |
| --- |
| **App.vue** |
| <template>  <router-view />  </template>  <script>  export default {  name: 'App',  }  </script>  <style>  #app {  font-family: Avenir, Helvetica, Arial, sans-serif;  -webkit-font-smoothing: antialiased;  -moz-osx-font-smoothing: grayscale;  text-align: center;  color: #2c3e50;  margin-top: 60px;  }  .logo{  width: 200px;  padding-bottom: 20px;  }  .signup input, .signin input{  display: block;  width: 300px;  height: 40px;  padding-left: 20px;  margin-bottom: 20px;  margin-left: auto;  margin-right: auto;  border: 1px dashed green;  }  .signup button, .signin button{  color: white;  width: 320px;  height: 40px;  border: 1px dashed green;  background-color: green;  cursor: pointer;  }  </style> |

**Part 10**

1. ดึงข้อมูลใน Property ใน Form Data

2. สร้างฟังก์ชันของการใช้งานปุ่มในการเข้าระบบ (SignIn)

3. ทำการทดสอบ API สำหรับในการเข้าระบบ (SignIn)

4. ทำการเรียกใช้งาน API สำหรับในการเข้าระบบ (SignIn)

5. สร้างการส่งต่อจากหน้าในการเข้าระบบ (SignIn) เพื่อไปยังหน้าที่กำหนด