# LAB 4

❖ **CONTENT**

o Enable Restful API exchange in backend with "cors" package

o Initialize new VueJS project as frontend

o Consume Restful APIs from backend with "axios" package

❖ **INTRODUCTION**

o CORS: Cross-Origin Resource Sharing. It is a security feature implemented in web browsers to control how web applications running at one origin can interact with resources from a different origin.

o VueJS is a progressive JavaScript framework used for building user interfaces and single-page applications. Vue.js is known for its simplicity, flexibility, and fine-grained reactivity.

o Some alternatives to VueS: ReactJS, AngularJS, SvelteJS

o Some ways to consume Restful API with VueJS:

    o *Axios: a promise-based HTTP client*

    o Fetch API: a browser built-in web API

o System architecture diagram:
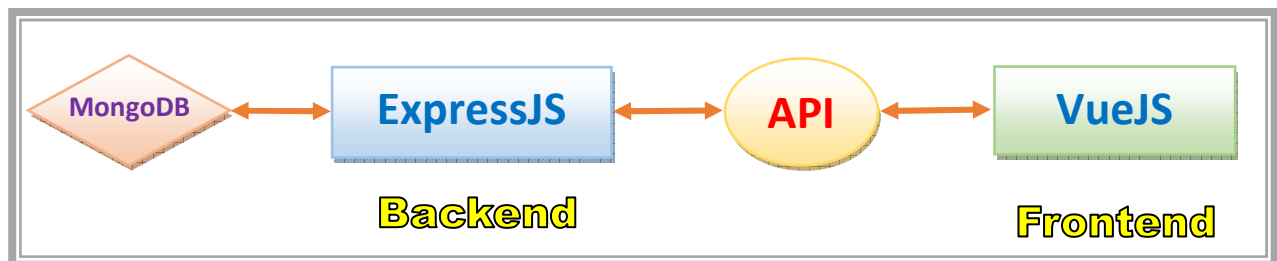
❖ INSTRUCTION

1. Continue with previous project and set it as backend project
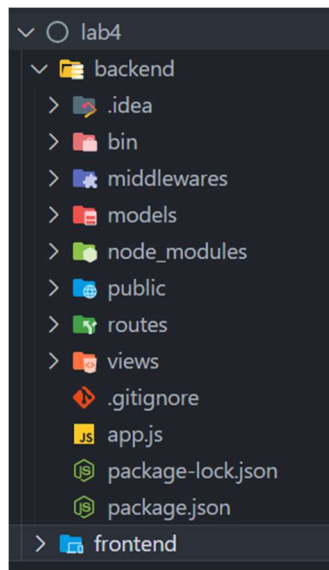

Figure 1 - Project structure

2. Install new package: CORS


```
npm install cors
```

Figure 2 - Install new package

3. Enable CORS in backend side for Restful API exchange


```
//import "cors" library
var cors = require('cors');

//usage 1: enable CORS requests for all domains
app.use(cors());

//usage 2: enable CORS requests for single route
app.get('/', cors(), (req, res) => {
  //codes go here
})

//usage 3: enable CORS requests for single domain
var corsOptions = {
  origin: 'https://longndt.com',
  optionsSuccessStatus: 200
}
app.use(cors(corsOptions));
app.get('/', cors(corsOptions), (req, res) => {
  //codes go here
})
```

*Figure 3 - Config CORS in backend (select 1 usage only) (**app.js**)*

4. Start the backend server and keep it running along with frontend server.

⇨ ***Do not stop it***, otherwise, 2 servers can not exchange data

Note: Default web server address for **ExpressJS** : **http://localhost:3000**

```
\lab4\backend>npm start
```

Figure 4 - Start the backend web server

5. Develop frontend side with VueJS framework.

   o Initialize new VueJS project

   npm create vue@latest *project_name*

   o Install axios package to consume API

   npm install axios

   o Start frontend web server

   npm run dev

Note: Default web server address for **VueJS** : **http://localhost:5173**

```
npm create vue@latest frontend
cd frontend
npm install axios
npm run dev
```

Figure 5 - Initialize VueJS project and install library in a new terminal

```
√ Add TypeScript? ... No / Yes
√ Add JSX Support? ... No / Yes
√ Add Vue Router for Single Page Application development? ... No / Yes
√ Add Pinia for state management? ... No / Yes
√ Add Vitest for Unit Testing? ... No / Yes
√ Add an End-to-End Testing Solution? » No
√ Add ESLint for code quality? ... No / Yes
```

Figure 6 - Config parameter when initializing VueJS project

6. Update the default view page src/App.vue to fetch data from backend

```
//import "axios" library to consume API from backend
import axios from "axios";
```

Figure 7 - import "axios" library

```
//declare the backend API url
var backendAPI = "http://localhost:3000/api/product";
```

Figure 8 - declare backend API

```
export default {
  data() {
    return {
      data: null,
    };
  },
  mounted() {
    this.fetchProducts();
  },
  methods: {
    fetchProducts() {
      axios
        .get(backendAPI)
        .then((response) => {
          this.data = response.data;
        })
        .catch((err) => {
          console.log("Error loading product list: " + err);
        });
    },
```

Figure 9 - fetch data with method axios.get()

```
deleteProduct(id) {
  const confirmed = confirm("Are you sure to delete this product?");
  if (confirm) {
    axios
      .delete(backendAPI + "/delete/" + id)
      .then(() => {
        this.fetchProducts();
      })
      .catch((err) => {
        console.error("Error deleting product:" + err);
      });
  }
},
```

Figure 10 - delete data with method axios.delete()

LONGNDT

```
<template>
  <div class="container">
    <table>
      <thead>
        <tr>
          <th colspan="5"><h3>Product List</h3></th>
        </tr>
        <tr>
          <th>Name</th>
          <th>Price</th>
          <th>Image</th>
          <th>Menu</th>
        </tr>
      </thead>
      <tbody>
        <tr v-for="product in data" :key="product._id">
          <td>{{ product.name }}</td>
          <td>${{ product.price }}</td>
          <td>
            <img :src="product.image" width="100" height="100" />
          </td>
          <td>
            <button @click="deleteProduct(product._id)" class="btn orange">
              Delete
            </button>
          </td>
        </tr>
      </tbody>
    </table>
  </div>
</template>
```

Figure 11 – Render the web template to display content

```
<title>Product Management System</title>
<!-- Compiled and minified CSS -->
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">

<!-- Compiled and minified JavaScript -->
<script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
</head>
```

*Figure 12 - Import Materialize CSS framework to file **src/index.html***

Product List

Name	Price	Image	Menu
dell xps	$1800		DELETE
galaxy s25 ultra	$1111		DELETE
macbook pro	$2000		DELETE
iphone 16 pro	$1234.56		DELETE

Figure 13 - Result of fetching data from backend