**Prerequisites**

Before you begin, ensure you have the following installed on your machine:

* **Node.js** (v12 or later)
* **MySQL** (or another compatible database)
* **npm** (comes with Node.js)
* **Vue CLI** (optional, but recommended for Vue.js projects)

**Step 1: Set Up the MySQL Database**

1. **Start MySQL Server**: Ensure your MySQL server is running.
2. **Create a Database**:

Open your MySQL command line or a GUI tool (like MySQL Workbench) and execute the following commands to create a database and table for your tasks:

CREATE DATABASE task\_management;

USE task\_management;

CREATE TABLE tasks (

id INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

completed BOOLEAN NOT NULL DEFAULT false

);

**Step 2: Set Up the Backend (Node.js)**

1. **Create a Backend Directory**:

mkdir my-backend

cd my-backend

npm init -y

npm install express mysql2 axios cors body-parser

1. **Create server.js**:

Create a file named server.js and add the following code:

const express = require('express');

const mysql = require('mysql2');

const axios = require('axios');

const cors = require('cors');

const bodyParser = require('body-parser');

const app = express();

const port = 3000;

app.use(cors());

app.use(bodyParser.json());

const db = mysql.createConnection({

host: 'localhost',

user: 'your\_username', // Replace with your MySQL username

password: 'your\_password', // Replace with your MySQL password

database: 'task\_management', // Your database name

});

db.connect((err) => {

if (err) throw err;

console.log('Connected to MySQL Database.');

});

const EXTERNAL\_API\_URL = 'https://67911187af8442fd7378e736.mockapi.io/tasks';

app.get('/tasks', async (req, res) => {

try {

const [localTasks] = await db.promise().query('SELECT \* FROM tasks');

const { data: externalTasks } = await axios.get(EXTERNAL\_API\_URL);

const allTasks = [...localTasks, ...externalTasks];

res.json(allTasks);

} catch (err) {

res.status(500).send(err);

}

});

app.post('/tasks', async (req, res) => {

const task = req.body;

try {

const [result] = await db.promise().query('INSERT INTO tasks SET ?', task);

await axios.post(EXTERNAL\_API\_URL, { title: task.title, completed: task.completed });

res.json({ id: result.insertId, ...task });

} catch (err) {

res.status(500).send(err);

}

});

app.delete('/tasks/:id', async (req, res) => {

const { id } = req.params;

try {

await db.promise().query('DELETE FROM tasks WHERE id = ?', id);

await axios.delete(`${EXTERNAL\_API\_URL}/${id}`);

res.sendStatus(204);

} catch (err) {

res.status(500).send(err);

}

});

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

1. **Run the Backend**:

In the terminal, navigate to the backend directory and start the server:

node server.js

**Step 3: Set Up the Frontend (Vue.js)**

1. **Create the Vue.js Project**:

If you have Vue CLI installed, create a new Vue.js project:

vue create my-frontend

cd my-frontend

Choose the default preset or customize it as needed.

1. **Install Axios**:

Inside the frontend directory, install Axios for making HTTP requests:

npm install axios

1. **Create API Utility**:

Create a new directory for utilities and a file for the API:

**File: src/utils/api.js**

import axios from 'axios';

const API\_URL = 'http://localhost:3000/tasks';

export const fetchTasks = async () => {

const response = await axios.get(API\_URL);

return response.data;

};

export const addTask = async (task) => {

const response = await axios.post(API\_URL, task);

return response.data;

};

export const deleteTask = async (id) => {

await axios.delete(`${API\_URL}/${id}`);

};

1. **Create Components**:

Create components for task management:

**TaskForm.vue**

<template>

<form @submit.prevent="addTask">

<input type="text" v-model="newTask.title" placeholder="Task Title" required />

<select v-model="newTask.completed">

<option :value="true">Completed</option>

<option :value="false">Not Completed</option>

</select>

<button type="submit">Add Task</button>

</form>

</template>

<script>

export default {

data() {

return {

newTask: {

title: '',

completed: false,

},

};

},

methods: {

async addTask() {

this.$emit('task-added', { ...this.newTask });

this.resetForm();

},

resetForm() {

this.newTask.title = '';

this.newTask.completed = false;

},

},

};

</script>

**TaskList.vue**

<template>

<div>

<h2>Tasks</h2>

<table>

<thead>

<tr>

<th>Title</th>

<th>Completed</th>

<th>ID</th>

<th>Action</th>

</tr>

</thead>

<tbody>

<tr v-for="task in tasks" :key="task.id">

<td>{{ task.title }}</td>

<td>{{ task.completed ? 'Yes' : 'No' }}</td>

<td>{{ task.id }}</td>

<td><button @click="removeTask(task.id)">Delete</button></td>

</tr>

<tr v-if="tasks.length === 0">

<td colspan="4">No tasks available.</td>

</tr>

</tbody>

</table>

</div>

</template>

<script>

export default {

props: {

tasks: Array,

removeTask: Function,

},

};

</script>

**App.vue**

<template>

<div>

<h1>Task Management</h1>

<TaskForm @task-added="addTask" />

<TaskList :tasks="tasks" :removeTask="removeTask" />

</div>

</template>

<script>

import TaskForm from './components/TaskForm.vue';

import TaskList from './components/TaskList.vue';

import { fetchTasks, addTask as apiAddTask, deleteTask as apiDeleteTask } from './utils/api';

export default {

components: {

TaskForm,

TaskList,

},

data() {

return {

tasks: [],

};

},

created() {

this.loadTasks();

},

methods: {

async loadTasks() {

this.tasks = await fetchTasks();

},

async addTask(newTask) {

const addedTask = await apiAddTask(newTask);

this.tasks.push(addedTask);

},

async removeTask(id) {

await apiDeleteTask(id);

this.tasks = this.tasks.filter(task => task.id !== id);

},

},

};

</script>

1. **Run the Frontend**:

In another terminal, navigate to the frontend directory and start the Vue.js application:

npm run serve

**Step 4: Test the Application**

1. **Open Browser**:

Open your web browser and navigate to http://localhost:8080 (or the port specified by Vue.js).

1. **Add Tasks**:

Use the form to add tasks. After submitting, check both the frontend and the MySQL database to ensure tasks are added correctly.

1. **Delete Tasks**:

Test deleting tasks and ensure the changes are reflected in both the frontend and the external API.

**Conclusion**

You have now set up and tested a Vue.js task management application that interacts with a local MySQL database and an external API. Ensure you have error handling and validations in place for a smoother user experience. If you have any questions or run into issues, feel free to ask!