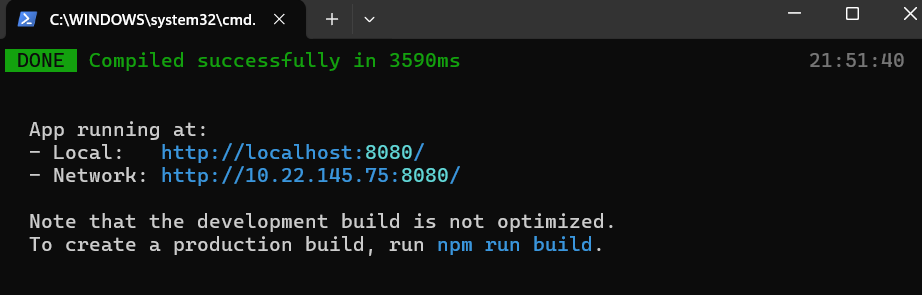
***Parte 2: Questões Práticas (10 questões)***

**1.** Instale o Node.js em sua máquina e, utilizando o terminal, verifique a versão instalada. Envie um print do comando node -v.

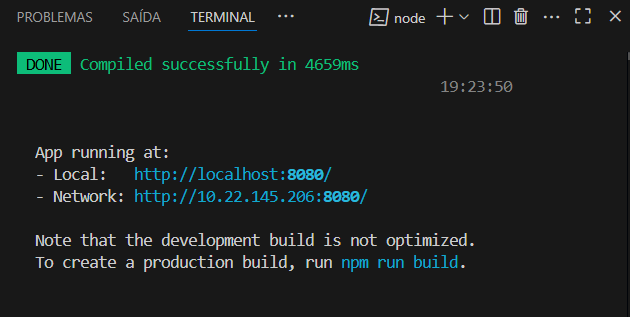


**2**. Crie um novo projeto Vue.js utilizando o Vue CLI. Certifique-se de que o servidor de desenvolvimento seja iniciado com sucesso.



**3.** Utilizando o npm, instale a biblioteca Axios em um projeto Vue.js e configure-a para realizar uma requisição GET para https://jsonplaceholder.typicode.com/posts. Exiba os dados retornados no console.



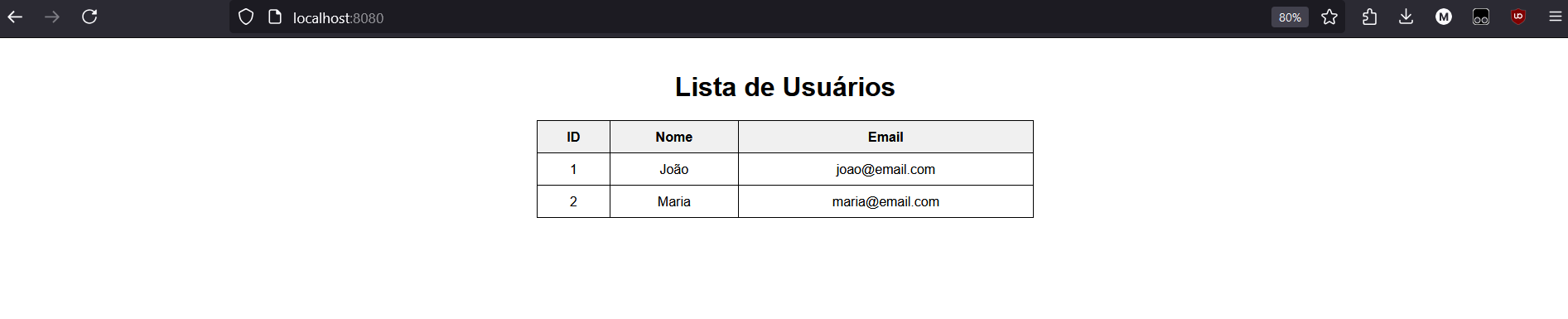


**4.** Implemente uma API RESTful com Node.js e Express que possua os seguintes endpoints: **○** GET /usuarios: Retorna uma lista de usuários.

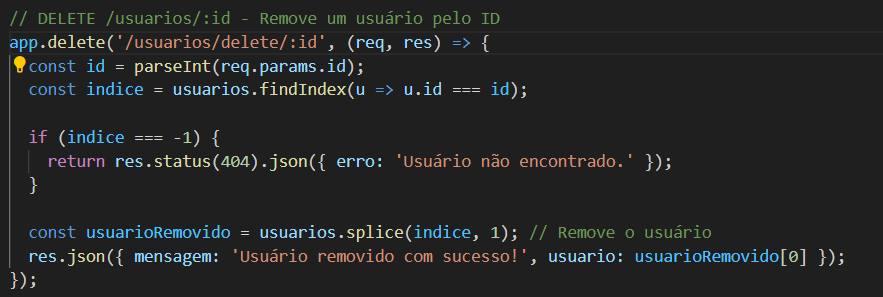
**○** POST /usuarios: Adiciona um novo usuário.



**5.** No Vue.js, crie uma interface que consuma os dados da API criada no exercício anterior e exiba a lista de usuários em uma tabela.



**6.** Adapte a API do exercício 4 para incluir um endpoint DELETE /usuarios/:id, que permita remover um usuário pelo ID.



**7.** Implemente uma página no Vue.js que permita adicionar e remover usuários, conectando-se à API do exercício anterior.

<template>

  <div class="container">

    <h1>Lista de Usuários</h1>

    <table border="1">

      <thead>

        <tr>

          <th>ID</th>

          <th>Nome</th>

          <th>Email</th>

          <th>Ações</th>

        </tr>

      </thead>

      <tbody>

        <tr v-for="usuario in usuarios" :key="usuario.id">

          <td>{{ usuario.id }}</td>

          <td>{{ usuario.nome }}</td>

          <td>{{ usuario.email }}</td>

          <td>

            <button @click="deletarUsuario(usuario.id)">Excluir</button>

          </td>

        </tr>

      </tbody>

    </table>

  </div>

</template>

<script>

import axios from 'axios';

export default {

  data() {

    return {

      usuarios: []

    };

  },

  mounted() {

    this.buscarUsuarios();

  },

  methods: {

    // Busca todos os usuários (GET)

    buscarUsuarios() {

      axios.get('http://localhost:3000/usuarios')

        .then(response => {

          this.usuarios = response.data;

        })

        .catch(error => {

          console.error('Erro ao buscar usuários:', error);

        });

    },

    // Deleta um usuário pelo ID (DELETE)

    deletarUsuario(id) {

  console.log('Tentando deletar o usuário com ID:', id); // <-- Log inicial

  axios.delete(`http://localhost:3000/usuarios/${id}`)

    .then((response) => {

      console.log('Resposta do DELETE:', response.data); // <-- Log sucesso

      this.buscarUsuarios();

    })

    .catch(error => {

      console.error('Erro ao deletar usuário:', error.response || error.message); // <-- Log erro

    });

}

  }

};

</script>

<style>

.container {

  max-width: 600px;

  margin: 40px auto;

  font-family: Arial, sans-serif;

  text-align: center;

}

table {

  width: 100%;

  border-collapse: collapse;

  margin-top: 20px;

}

th, td {

  padding: 10px;

}

th {

  background-color: #f0f0f0;

}

button {

  background: red;

  color: white;

  border: none;

  padding: 5px 10px;

  cursor: pointer;

}

button:hover {

  background: darkred;

}

</style>

**8.** Utilizando o PHP, implemente uma API básica para listar produtos de uma tabela fictícia chamada produtos no banco de dados MySQL. Crie o endpoint GET /produtos.

<?php

$host = "127.0.0.1";

$dbname = "produtos\_api";

$username = "root";

$password = "root";

header("Content-Type: application/json; charset=UTF-8");

header("Access-Control-Allow-Origin: \*");

try {

    $conn = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);

    $conn->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

} catch (PDOException $e) {

    http\_response\_code(500);

    echo json\_encode(["erro" => "Falha na conexão com o banco: " . $e->getMessage()]);

    exit;

}

$method = $\_SERVER['REQUEST\_METHOD'];

$uri = $\_SERVER['REQUEST\_URI'];

if ($method === 'GET' && preg\_match('/\/produtos$/', $uri)) {

    try {

        $stmt = $conn->query("SELECT id, nome, descricao, preco FROM produtos");

        $produtos = $stmt->fetchAll(PDO::FETCH\_ASSOC);

        echo json\_encode($produtos, JSON\_UNESCAPED\_UNICODE | JSON\_PRETTY\_PRINT);

    } catch (PDOException $e) {

        http\_response\_code(500);

        echo json\_encode(["erro" => "Erro ao buscar produtos: " . $e->getMessage()]);

    }

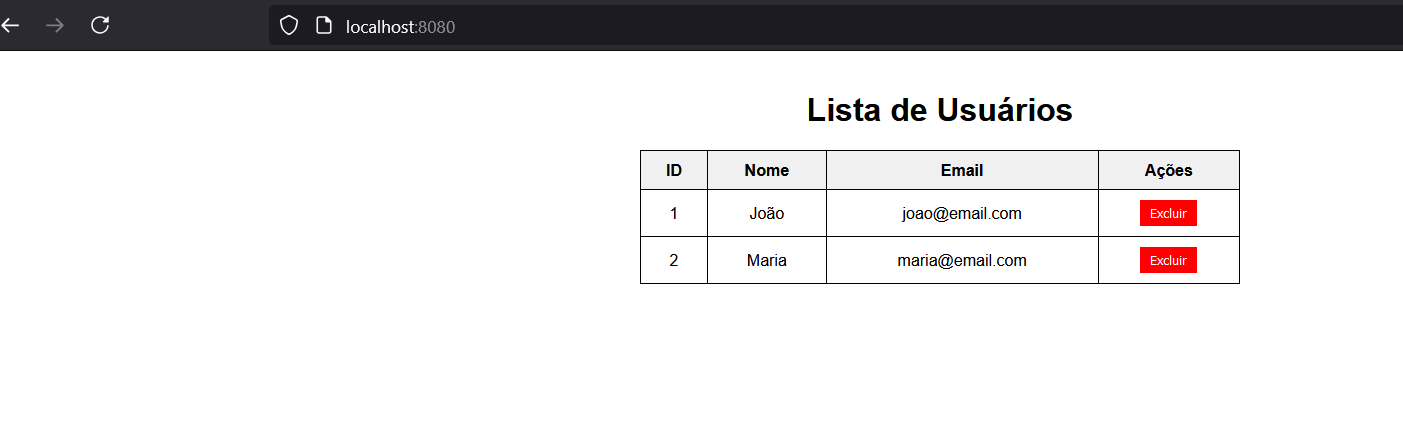
} else {

    http\_response\_code(404);

    echo json\_encode(["erro" => "Endpoint não encontrado."]);

}

**9.** Configure um projeto com Vue CLI e adicione um formulário reativo que permita o cadastro de usuários. O formulário deve conter os campos Nome, Email e Telefone.



**10.** Consuma a API pública de CEP (ex.: ViaCEP) no Vue.js. Crie uma interface onde o usuário insira o CEP e os dados retornados sejam exibidos na tela (logradouro, bairro, cidade e estado).

